Table of Contents

Overview ...................................................................................................................... 1
  About Splunk IT Service Intelligence ............................................................ 1
  IT Service Intelligence concepts and features .............................................. 1

Installation ................................................................................................................. 6
  ITSI deployment planning ........................................................................... 6
  Install Splunk IT Service Intelligence ...................................................... 16
  Upgrade Splunk IT Service Intelligence .................................................... 20
  Uninstall Splunk IT Service Intelligence ................................................... 23
  Use the free Splunk IT Service Intelligence sandbox .................................. 26

Permissions ............................................................................................................. 27
  Configure users and roles in ITSI ............................................................... 27
  Add capabilities to a role in ITSI ............................................................... 35
  KV store collection permissions in ITSI ................................................... 42
  Restrict access to objects in ITSI ............................................................... 43
  Overview of service-level permissions in ITSI ........................................ 44
  Implement service-level permissions in ITSI ............................................. 49
  Create teams in ITSI ................................................................................. 52

Service Configuration ............................................................................................. 54
  Overview of creating services in ITSI ....................................................... 54
  Define entities in ITSI ................................................................................ 55
  Create a single service in ITSI ................................................................. 64
  Overview of configuring services in ITSI .................................................. 66
  Add entity rules to a service in ITSI .......................................................... 67
  Add service dependencies in ITSI ............................................................. 68
  Backfill service health scores in ITSI ......................................................... 71
  Configure a service created from a service template in ITSI ................... 72
  Clone a service in ITSI ............................................................................... 74
  Bulk import services in ITSI ...................................................................... 74
  How service health scores work in ITSI .................................................... 82

KPI Configuration ..................................................................................................... 85
  Overview of configuring KPIs in ITSI ......................................................... 85
  Add a KPI to a service in ITSI .................................................................. 87
  Set KPI importance values in ITSI ........................................................... 97
  Create KPI threshold time policies in ITSI ............................................... 98
  Apply anomaly detection to a KPI in ITSI ................................................. 102
# Table of Contents

## KPI Configuration
- Create KPI base searches in ITSI ........................................................... 106
- Schedule KPI searches to update at the same time in ITSI .................... 113

## Service Templates
- Manage services in bulk with service templates in ITSI ......................... 115
- Create a service template in ITSI ............................................................ 119
- Update a service template in ITSI ......................................................... 121
- Link a service to a service template in ITSI ............................................. 123
- Manage service templates in ITSI ........................................................... 124

## Predictive Analytics
- Predict and prevent outages with ITSI Predictive Analytics .................... 128
- Set up Predictive Analytics in ITSI ...................................................... 129
- Access Predictive Analytics in ITSI ...................................................... 130
- Determine whether your service is a good fit for modeling in ITSI ............ 131
- Train a predictive model in ITSI ............................................................ 133
- Test a predictive model in ITSI ............................................................. 140
- Create an alert for potential service degradation in ITSI ......................... 142
- Add a predictive model to a glass table in ITSI ...................................... 144
- Retrain a predictive model in ITSI ....................................................... 145
- Delete a predictive model in ITSI ......................................................... 147
- Predictive Analytics performance considerations in ITSI ...................... 147
- ITSI Predictive Analytics use case ....................................................... 149

## Notable Events
- Customize Episode Review in ITSI ..................................................... 161
- Manage notable event indexes in ITSI ................................................. 162
- Tune notable event grouping in ITSI .................................................... 166
- Ingest third-party alerts as ITSI notable events ..................................... 168
- Ingest SNMP traps in ITSI ................................................................. 173
- Use the Notable Event Actions SDK .................................................. 177
- Notable Event Actions SDK reference .............................................. 178

## Aggregation Policies
- Notable event aggregation policies overview for ITSI ......................... 197
- Use the default aggregation policy in ITSI ......................................... 197
- Create a custom aggregation policy in ITSI ......................................... 198
- Group similar events with Smart Mode in ITSI ................................... 203
# Table of Contents

## Correlation Searches
- Correlation search overview for ITSI .......................................................... 206
- Create correlation searches in ITSI ............................................................... 206
- Configure correlation searches in ITSI ......................................................... 207
- Create multi-KPI alerts in ITSI .................................................................... 213

## Maintenance
- Create maintenance windows in ITSI ....................................................... 218
- Back up and restore ITSI KV store data ...................................................... 221
- kvstore_to_json.py operations in ITSI ........................................................ 228

## Administer Splunk ITSI with configuration files
- About configuration files ............................................................................. 244
- List of ITSI configuration files .................................................................... 245

## Configuration file reference
- alert_actions.conf ................................................................................... 253
- app_common_flags.conf ........................................................................... 256
- authorize.conf ......................................................................................... 258
- commands.conf ....................................................................................... 277
- collections.conf ....................................................................................... 285
- datamodels.conf ....................................................................................... 288
- deep_dive_drilldowns.conf ....................................................................... 295
- distsearch.conf ........................................................................................ 302
- drilldownsearch_offset.conf .................................................................... 316
- fields.conf ................................................................................................ 317
- glasstable_icon_library.conf ................................................................... 321
- inputs.conf ............................................................................................... 322
- itsi_da.conf ............................................................................................ 335
- itsi_deep_dive.conf .................................................................................. 337
- itsi_glass_table.conf ................................................................................ 339
- itsi_kpi_base_search.conf ....................................................................... 340
- itsi_kpi_template.conf ............................................................................ 342
- itsi_kpi_threshold_template.conf .............................................................. 343
- itsi_module_settings.conf ....................................................................... 345
- itsi_module_viz.conf ................................................................................ 346
- itsi_notable_event_retention.conf ............................................................. 348
- itsi_notable_event_severity.conf ............................................................... 349
- itsi_notable_event_status.conf ................................................................ 351
# Table of Contents

## Configuration file reference

- itsi_service_analyzer.conf ................................................................. 352
- itsi_service_template.conf ................................................................. 354
- itsi_settings.conf ........................................................................... 356
- itsi_team.conf ................................................................................. 359
- limits.conf .......................................................................................... 361
- logging.conf ....................................................................................... 443
- macros.conf ......................................................................................... 444
- mad.conf ............................................................................................. 447
- notable_event_actions.conf ................................................................. 454
- notable_event_commonality.conf ......................................................... 459
- notable_event_correlation.conf ......................................................... 460
- props.conf ........................................................................................... 463
- restmap.conf ....................................................................................... 501
- savedsearches.conf ........................................................................... 511
- searchbnf.conf ................................................................................. 515
- service_analyzer_settings.conf ............................................................ 519
- threshold_labels.conf ........................................................................ 520
- threshold_periods.conf ..................................................................... 522
- transforms.conf .................................................................................. 523
- ui-tour.conf ......................................................................................... 549
- visualizations.conf ........................................................................... 554
- web.conf ............................................................................................... 558

## Troubleshooting

- Search IT Service Intelligence logs ......................................................... 588
- Installation and configuration considerations and issues ....................... 588
Overview

About Splunk IT Service Intelligence

Splunk IT Service Intelligence (ITSI) is a scalable IT monitoring and analytics solution that provides actionable insight into the performance and behavior of your IT operations.

As an ITSI admin, you can performing the following tasks:

- Ingest and analyze all types of IT event data.
- Create services based on your IT infrastructure for service-level agreement (SLA) monitoring.
- Add KPIs (key performance indicators) with severity-level thresholds to monitor the status of IT processes.
- Predict service health scores and prevent potential service outages.
- Set up service dependencies to identify upstream/downstream causal relationships.
- Build custom glass table visualizations of KPI search results for any service.
- Create custom deep dives to compare and analyze time-series events across IT services.
- Create multi-KPI alerts to generate notable events based on multiple KPI threshold trigger conditions.
- Apply anomaly detection to identify statistical outliers and find root cause faster.

To learn more about ITSI, see ITSI concepts and features in this manual.

For installation instructions, see Install Splunk IT Service Intelligence in this manual.

For detailed user instructions, see the Splunk IT Service Intelligence User Manual.

IT Service Intelligence concepts and features

Familiarize yourself with the following concepts and features before deploying Splunk IT Service Intelligence (ITSI).
**Key ITSI concepts**

**Service**

A service is a logical mapping of IT objects that applies to your business goals. Splunk ITSI provides for a broad definition of a service. For example, a service can be:

- An application or group of applications
- An infrastructure tier (such as web, database, or network tier)
- A business service, such as an online store, with multiple tiers
- A single process, such as one instance of an application running on a host

ITSI lets you create services that model your IT infrastructure. ITSI services contain KPIs (Key Performance Indicators), which make it possible to monitor service health, perform root cause analysis, receive alerts, and ensure that your IT operations are in compliance with business SLAs (service-level agreements).
**Entity**

An **entity** is an IT infrastructure component, such as

- A physical or virtual server
- A network device (switch, router)
- A user (AD/LDAP)
- A storage system or volume
- An operating system process
- A software application (database, web server, business app)
- An application process instance (for example, 2 instances of the same web server application is 2 separate entities)

Each entity has specific attributes and relationships to other IT processes that uniquely identify it. For example, a server that you define as an entity can have multiple IP addresses, MAC addresses, DNS names, and so on.

Data comes from applications on the entities, log files, and through API data collection processes. You can map data to entities in ITSI through aliases (field/value pairs) that you extract from Splunk searches. For example: `host=10.141.24.63`. In this way, entities link services to Splunk search results.

An entity is similar to a configuration item in the ITIL framework. However, an entity is never a service itself.

Entities are optional.

**KPI**

A **KPI** (Key Performance Indicator) is a recurring saved search that returns the value of an IT performance metric, such as CPU load percentage, memory used percentage, response time, and so on.

ITSI lets you create KPIs and add them to your services. You can then use KPI search result values inside ITSI to monitor service health, check the status of IT components, and troubleshoot trends that might indicate an issue with your IT systems.

For example, `cpu_load_percent` is a KPI that measures the CPU load percentage on a server. If your organization has a site uptime guarantee of 99.9% per month, you will need to monitor the status of this KPI (and others) to ensure that CPU performance remains within acceptable parameters.
Key ITSI features

**Service Analyzer**

The Service Analyzer provides an overview of ITSI service health scores and KPI search results that are currently trending at the highest severity levels. Use the Service Analyzer to quickly view the status of IT operations and to identify services and KPIs running outside expected norms. Click on any tile in the Service Analyzer to drill down to the deep dives for further analysis and comparison of search results over time.

For more information, see Monitor the health of your services with the ITSI Service Analyzer in the *IT Service Intelligence User Manual*.

**Glass tables**

Glass tables are custom visualizations that let you monitor KPI and service health scores. You can use glass tables to create dynamic contextual views of your IT topology or business processes and monitor them in real time. Glass tables features a drawing canvas where you can draw custom images, upload pre-existing images, and/or add icons from the Splunk icon library.

For more information, see Create a glass table in ITSI in the *Splunk IT Intelligence User Manual*.

**Deep dives**

Deep dives are an investigative tool that let you quickly identify and troubleshoot issues in your IT environment. Deep dives provide swimlane views that let you stack KPI search results over time and create contextual views showing all KPIs in a service. You can use deep dives to quickly zoom in on metric and log events, and visually correlate root cause.

For more information, see Overview of deep dives in ITSI in the *IT Service Intelligence User Manual*.

**Multi-KPI alerts**

A multi-KPI alert is an alert that is based on multiple KPI trigger conditions. When trigger conditions occur simultaneously, a correlation search generates a notable event. Multi-KPI alerts let you correlate the status of multiple KPIs, which can give you insight into system behavior, and help you to identify causal relationships that might negatively impact system performance.
For more information, see Create multi-KPI alerts in ITSI in the *IT Service Intelligence User Manual*.

**Episode Review**

ITSI provides a notable events management framework that lets you triage and analyze groups of notable events (episodes). ITSI generates notable events when a correlation search or multi-KPI alert meets specific conditions that you define. An episode is a group of events occurring as part of a larger sequence (an incident or period considered in isolation). Use Episode Review to view episode details and identify issues that might impact the performance and availability of your IT services.

Other ITSI notable events management features include a python-based, notable event action SDK, which lets you define secondary, post-notable event actions, such as add tags, add comments, view notable event activities, change owner, change status, change severity, and so on.

For more information, see Overview of Episode Review in ITSI in the *IT Service Intelligence User Manual*.

**ITSI modules**

ITSI modules provide pre-built KPIs, entity definitions, and dashboard visualizations. ITSI modules are tailored to specific IT use cases, such as monitoring and troubleshooting operating system hosts, load balancers, databases, app servers, and so on. Modules are optimized to process data that you collect using Splunk add-ons.

For more information, see ITSI Modules overview in the *IT Service Intelligence Modules Manual*. 
Installation

ITSI deployment planning

Deploy Splunk IT Service Intelligence on a configured Splunk platform installation. Review the system and hardware requirements and the search head and indexer considerations before deploying IT Service Intelligence.

Preparation for deployment

The first step in planning your ITSI deployment is to evaluate your objectives. This includes determining the numbers and types of services, KPIs, and entities that you want your ITSI deployment to contain. It is also critical to confirm that you have sufficient underlying hardware capacity to support optimal ITSI performance. You must also confirm compatibility with the Splunk Enterprise version on which you plan to deploy ITSI.

In preparation for your ITSI deployment, make sure you have the following information:

1. A list of services, KPIs, and glass table views that you want to create.
2. A list of your entities. Entities are usually hosts, but can also be users, mobile devices, and so on. Entities for hosts should include at minimum IP address, host name, and designated role (for example, web, db, app server).
3. Verify existing hardware performance. Verify performance using this search query:

   \[
   \text{index=\_introspection sourcetype=splunk\_resource\_usage component=Hostwide earliest=-5m | timechart avg(data.cpu\_user\_pct) by host}
   \]

   If it takes more than 2-5 seconds for the search query to complete, check performance in the Job Inspector to investigate the issue. This might indicate your current hardware is insufficient or badly configured. Or you might have a high latency dispatch requiring architecture changes.
Operating system requirement

For a list of supported operating systems, browsers, and file systems, see System Requirements in the Splunk Enterprise Installation Manual.

Splunk Enterprise system requirement

Splunk IT Service Intelligence requires a 64-bit OS install on all search heads and indexers. For the list of supported operating systems, browsers, and file systems, see System requirements for use of Splunk Enterprise on-premises in the Splunk Enterprise Installation Manual.

Use this table to determine the compatibility of the IT Service Intelligence 4.1.x versions and Splunk platform versions.

<table>
<thead>
<tr>
<th>Splunk IT Service Intelligence version</th>
<th>Splunk platform version</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.0</td>
<td>7.0.x - 7.2.x</td>
</tr>
</tbody>
</table>

ITSI license requirement

ITSI requires a separate ITSI license in addition to your Splunk Enterprise license. Your Splunk representative will provide you with an appropriate ITSI license at time of purchase. For ITSI license installation instructions, see Install a license in the Splunk Enterprise Installation and Configuration Manual.

Java requirements

ITSI 4.1.x requires Java 7 or Java 8 to run anomaly detection and event management features. Java 8 is required for Windows installations. ITSI supports OpenJDK and Oracle JDK 7 and 8. Java installation is required on search heads only, not indexers or forwarders.

Available deployment architectures

You can deploy Splunk IT Service Intelligence in a single instance deployment or a distributed search deployment. Splunk IT Service Intelligence is also available in Splunk Cloud. Before you deploy Splunk IT Service Intelligence on premises, familiarize yourself with the components of a Splunk platform deployment. See Components of a Splunk Enterprise deployment in the Capacity Planning Manual.
**Single instance deployments**

For a simple and small deployment, install ITSI on a single Splunk platform instance. A single instance functions as both a search head and an indexer. Use forwarders to collect your data and send it to the single instance for parsing, storing, and searching.

You can use a single instance deployment for a lab or test environment, or a small system with one or two users running concurrent searches. For instructions on installing ITSI on a single Splunk Enterprise instance, see Install ITSI on a single instance.

**Distributed deployments**

You can deploy ITSI across any distributed architecture supported by Splunk Enterprise. This includes all types of deployment topologies, from small departmental deployments using a single instance for both indexer and search head, to large enterprise deployments using several search heads, dozens of indexers, and hundreds of forwarders. See Types of distributed deployments in the *Distributed Deployment Manual*.

For instructions on installing ITSI in a distributed environment, see Install ITSI in a distributed environment.

In a distributed deployment, ITSI supports both search head and indexer clustering options. For details on search head clustering architecture, see Search head clustering architecture in the *Distributed Search Manual*. For details on indexer cluster architecture, see The basics of indexer cluster architecture in the *Managing Indexers and Clusters of Indexers Manual*.

**Cloud deployments**

Splunk IT Service Intelligence is available as a service in Splunk Cloud. The Splunk Cloud deployment architecture varies based on data and search load. Splunk Cloud customers work with Splunk Support to set up, manage, and maintain their cloud infrastructure. For information on Splunk Cloud managed deployments, see the Types of Splunk Cloud deployment in the Splunk Cloud *User Manual*.

**IT Service Intelligence search head considerations**

IT Service Intelligence does not require a dedicated search head. However, note that ITSI is not supported on the same search head as Splunk Enterprise
Security. For scalability beyond about 200 discrete KPIs, a search head cluster is recommended.

Real-time searches cannot be disabled on the search head, otherwise ITSI notable event grouping will stop working.

**CPU and memory**

CPU core count and RAM are critical factors in search head performance. Use the maximum number of CPU cores and RAM available for your system.

**Virtual machines**

When running a search head on a virtual machine, make sure to allocate all available CPU and RAM to the search head.

**Forward search head data to indexers**

ITSI runs KPI searches on the search head and by default stores data in the local itsi_summary index. It is considered a best practice to forward all internal data from search heads to indexers. There are two basic search head configuration scenarios for forwarding data to indexers:

1. **Non-clustered search heads**: Configure search heads to forward data to indexers.
2. **Clustered search heads**: In this scenario, you must configure outputs.conf to forward data from search heads to indexers. Then use the deployer to push the configuration file to cluster members.

For detailed instructions on how to configure search heads to forward data to indexers, see Forward search head data to indexers in the Distributed Search manual.

**Search head clustering**

A search head cluster multiplies the maximum search concurrency of the Splunk environment by the number of search heads in the cluster. To manage the increase in search load when implementing a search head cluster, add additional indexers or allocate additional cores to indexers. For a complete list of requirements, see System requirements and other deployment considerations for search head clusters in the Splunk Enterprise Distributed Search Manual.
For instructions on deploying ITSI in a search head cluster environment, see Install ITSI on a search head cluster in this manual.

**Configure multiple ITSI deployments to use the same indexing layer**

You can deploy separate non-clustered ITSI search heads for different purposes that forward data to the same indexers. For example, one search head could be used for production and a second search head could be used for testing. You can also deploy separate search head clusters that use the same indexer cluster. Prior to ITSI version 4.0.0, this architecture was not supported.

As of ITSI version 4.0.0, the data from each ITSI search head or search head cluster can be stored in separate indexes. For example, you could rename the itsi_summary index on the production ITSI search head instance to itsi_summary_prod and rename the itsi_summary index on the test ITSI search head instance to itsi_summary_test, and likewise for the other ITSI indexes. After completing the steps in this procedure, searches will point to the desired index for each separate deployment.

Rename the anomaly_detection index before enabling anomaly detection for KPIs. If the name for the anomaly_detection index is changed after anomaly detection has been enabled for any KPIs, the index will not be updated in anomaly detection searches. If anomaly detection was enabled prior to renaming the anomaly_detection index, disable both trending and cohesive anomaly detection for all the KPIs that use it, then enable it again.

To create custom ITSI indexes for multiple ITSI deployments, do the following:

1. On each search head, change the default ITSI index names to the new index names you want to use for the data from that search head in the following files:
   - itsi/local/alert_actions.conf
   - itsi/local/savedsearches.conf
   - SA-ITOA/local/macros.conf
   - SA-ITOA/local/alert_actions.conf
   - SA-ITOA/local/itsi_rules_engine.properties

2. Create a local version of commands.conf in SA-ITOA/local/commands.conf. Add the following lines to point to the local version of itsi_rules_engine.properties:

```plaintext
[itsirulesengine]
command.arg.3=-DitsiRulesEngine.configurationFile=../local/itsi_rules_engine.properties
```
3. On each Splunk indexer, add a new index for each renamed ITSI index in Settings > indexes or in SA-IndexCreation/local/indexes.conf.
4. Restart Splunk software to put the changes into effect.
5. On each search head, do the following to check that searches are pointing to the correct indexes:
   ♦ Navigate to Splunk > Data Inputs > HTTP Event Collector. You should see the renamed index names for the five ITSI event management tokens which have the following source types:
     itsi_notable:event, itsi_notable:archive, itsi_notable:audit, itsi_notable:group.
   ♦ Check the Event Analytics Audit dashboard to make sure the searches run as expected as these searches use macros.
   ♦ Try replacing macro searches with the name of the renamed index. For example:

     `itsi_event_management_index_with_close_events` | stats count AS events

     should return the same events as:

     index="<new name for itsi_tracked_alerts>" | stats count AS events

   ♦ Make sure that the data is displaying as expected in service analyzers, deep dives, glass tables, and Episode Review.
   ♦ Verify that ITSI users can access the new indexes.

Indexer considerations

In a large ITSI deployment, indexers must be able to process thousands of queries per minute. A proper ratio of search heads to indexers can help handle this load. For help determining your indexer requirements, consult your Splunk Professional Services or support representative.

CPU and memory

CPU core count and RAM are critical factors in indexer performance. Make sure to scale up indexer CPU cores as appropriate to handle the number of concurrent searches driven by ITSI. In addition, make sure to install as much RAM as possible on your indexer machines. We recommend 64GB or more of RAM per indexer.
**Indexer clustering**

ITSI supports both single and multi-site **Indexer clusters**. For more information, see Indexer cluster deployment overview in *Managing Indexers and Clusters of Indexers*.

**Performance considerations**

ITSI works by way of KPI collection via searches against information stored within the Splunk Enterprise environment. ITSI production deployments might require additional hardware, depending on several factors, including the existing unused capacity of the environment, the number of concurrent KPI searches, the version of Splunk Enterprise in production, and other performance considerations specific to each deployment.

To determine when to scale your Splunk Enterprise deployment, see Performance Checklist in the *Capacity Planning Manual*.

**Planning your hardware requirements**

ITSI performance depends on the ability to perform multiple fast, concurrent searches. Performance results depend on both search optimization and the capacity of your deployment to run multiple concurrent searches.

When planning your ITSI hardware requirements, consider these ITSI-specific factors that impact performance:

- Average KPI run time
- Frequency of KPIs (1, 5 or 15 minute)
- Number of entities that are being referenced per KPI

Also consider the following Splunk Enterprise factors that might impact performance:

- Number of concurrent users. See How concurrent users affect Splunk Enterprise performance in the *Capacity Planning Manual*.

**ITSI capacity planning**

ITSI capacity planning is governed by several variables. The three key variables in determining how many indexers and search heads you need are average KPI
run time, the frequency of KPIs (1, 5, or 15 minute), and the number of entities being referenced per KPI. These can vary significantly in real-world deployments and you should contact your Splunk sales representative for specific ITSI capacity planning recommendations based on your environment.

There are several other variables to consider that impact the number of indexers and search heads you need, including the number of cores on those machines, the total amount of data being indexed, the total number of concurrent users, and so on.

**Indexer and search head sizing examples**

The following examples show roughly the number of indexers and search heads required to run the specified number of KPIs. These numbers are for example purposes only and will vary based on your environment.

The following variables are fixed for each of the proceeding examples.

- Only 5 minute KPIs
- 12 cores per search head and indexer
- Environment dedicated to ITSI alone
- Splunk Enterprise version 6.6 or later
- Use of "entity" refers to entities stored in the KV store and in the examples is a per-KPI measure, not the total number of entities in the system. If simple entity splits are done for KPIs and are not based on entities in a KV store, but extracted fields in Splunk searches, they need not be considered entities.
- 1 indexer required per 100GB indexed

**Example Set 1: Average run time per KPI = 10 seconds**

Example A: 0 Entities per KPI, 100 GB indexed per day

<table>
<thead>
<tr>
<th>KPIs</th>
<th>Indexers</th>
<th>Search Heads</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1 indexer</td>
<td>1 search head</td>
</tr>
<tr>
<td>500</td>
<td>2 indexers</td>
<td>1 search head</td>
</tr>
<tr>
<td>1000</td>
<td>3 indexers</td>
<td>2 search heads</td>
</tr>
</tbody>
</table>

Rough capacity plan:

~ (Per 500 KPIs 1+ search head, 1+ indexer) + 1 Indexer.
Example B: 50 entities per KPI, 500 GB indexed per day

<table>
<thead>
<tr>
<th>KPIs</th>
<th>Indexers</th>
<th>Search Heads</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>5 indexers</td>
<td>1 search head</td>
</tr>
<tr>
<td>500</td>
<td>5 indexers</td>
<td>2 search heads</td>
</tr>
<tr>
<td>1000</td>
<td>5 indexers</td>
<td>3 search heads</td>
</tr>
</tbody>
</table>

Rough capacity plan:

~ (Per 333 KPIs 1+ search head)

**Example Set 2: Average run time per KPI = 5 seconds**

Example A: 0 entities per KPI, 100 GB indexed per day

<table>
<thead>
<tr>
<th>KPIs</th>
<th>Indexers</th>
<th>Search Heads</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1 indexer</td>
<td>1 search head</td>
</tr>
<tr>
<td>500</td>
<td>1 indexer</td>
<td>1 search head</td>
</tr>
<tr>
<td>1000</td>
<td>2 indexers</td>
<td>2 search heads</td>
</tr>
</tbody>
</table>

Rough capacity plan:

~ (Per 950 KPIs 1+ search head), (Per 730 KPIs 1+ indexer)

Example B: 50 entities per KPI, 500 GB indexed per day

<table>
<thead>
<tr>
<th>KPIs</th>
<th>Indexers</th>
<th>Search Heads</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>5 indexers</td>
<td>1 search head</td>
</tr>
<tr>
<td>500</td>
<td>5 indexers</td>
<td>1 search head</td>
</tr>
<tr>
<td>1000</td>
<td>5 indexers</td>
<td>3 search heads</td>
</tr>
</tbody>
</table>

Rough capacity plan:

~ (Per 333 KPIs 1+ search head)

It is important to distinguish between the number of KPIs and the number of KPI searches. When using KPI base searches, these two can be dramatically different, and it is the number of actual search jobs that matters.
KV store size limits

Splunk IT Service Intelligence requires the KV store. You can backup and restore ITSI KV store data using the Backup/Restore UI, or using the `kvstore_to_json` python script. For more information, see Backup and restore ITSI configuration data in this manual.

The limit of a single batch save to a KV store collection is 50MB. As a result, if one KPI base search is in use by multiple services, and the total size of your services exceeds 50MB, ITSI generates an error. Additionally, if the number of objects (services, KPIs, etc.), exceeds the KV store memory limits, services might be lost during a backup or migration. To avoid these issues, check the total amount of data that your services contain, and, if necessary, increase the KV store size limit in `limits.conf`, as shown below:

1. Use the Backup/Restore UI or the `kvstore_to_json.py` script to create a backup of your system.
2. If the size of `itsi_services___service___0.json` exceeds 50 MB, increase the KV store size limit, as shown in steps 3 and 4.
3. Add the following stanza to `/local/limits.conf`:

   `[kvstore]
   # The maximum size, in megabytes, of a batch save query.
   max_size_per_batch_save_mb = 50

4. Increase the value of `max_size_per_batch_save_mb` to a higher value.
5. Additionally, if you have more than 1,000 kpi and services, add the following stanza to `/local/limits.conf`:

   `[kvstore]
   # The maximum size, in megabytes, of the result that will be returned for a single query to a collection.
   max_size_per_result_mb = 100

6. Increase the value of `max_size_per_result_mb` to roughly 50MB per 1,000 KPIs.

Search macros in ITSI

ITSI uses search macros to simplify and consolidate lengthy KPI searches. You can view a complete list of search macros used in ITSI, including macro definitions and usage details in macros.conf. For more information on search macros, see Use search macros in the Knowledge Manager Manual.
HTTP event collector

ITSI uses HTTP Event Collector (HEC) for event management. HEC runs as a separate app called splunk_httpinput and stores its input configuration in $SPLUNK_HOME/etc/apps/splunk_httpinput/local.

HEC requires port 8088 be open for local traffic. No additional HEC configuration is required.

For more information on HTTP Event Collector, see Set up and use HTTP Event Collector in the Splunk Enterprise Getting Data In manual.

ITSI compatibility with other apps

Do not install ITSI and Splunk Enterprise Security on the same search head or search head cluster. With the exception of Enterprise Security, ITSI can be deployed on Splunk Enterprise instances with other Splunk apps.

For a comprehensive evaluation of your environment, consult your Splunk Professional Services or support representative.

Install Splunk IT Service Intelligence

You can install Splunk IT Service Intelligence (ITSI) on Splunk Enterprise in both single-instance and distributed deployment environments. For an overview of these Splunk Enterprise environments, see Deployment architectures in this manual. Splunk Cloud customers must work with Splunk Support to coordinate access to the IT Service Intelligence search head.

Before you install, review the latest deployment requirements, including Splunk Enterprise version requirements and ITSI license requirements. See Splunk Enterprise version requirement in this manual.

Before you install Splunk IT Service Intelligence

Perform the following tasks before you install Splunk IT Service Intelligence.

1. Install required Java components

ITSI requires Java 7 or Java 8 to run anomaly detection and notable event management features. Java 8 is required for Windows installations. Java 9 and
10 are not currently supported. You can install Java prior to or after installing ITSI.

Install Java 8 on all search heads running ITSI. On RHEL and Ubuntu Linux, you can install the vendor packages: java-1.8.0-openjdk on RHEL Linux and openjdk-8-jdk on Ubuntu Linux. Alternatively, you can download and install the latest version of Oracle Java 8 (JRE or JDK).

If the JAVA_HOME environment variable is set correctly to the base of the Java installation, or the ?java? executable (or ?java.exe? in Windows) can be found using the PATH environment variable, no additional action is required. This is typically the case if you install the vendor Java packages in Linux or OS X.

If you install Java to a custom location (for example, when you install Oracle Java from oracle.com) and neither PATH nor JAVA_HOME is set to the Java installation, you must set JAVA_HOME in $SPLUNK_HOME/etc/splunk-launch.conf. For example:

```
JAVA_HOME=/opt/jdk1.8.0_74.jdk
```

2. **(Optional) Install required Predictive Analytics apps**

To use ITSI’s Predictive Analytics capabilities, install the Splunk Machine Learning Toolkit (MLTK) and share the machine learning macros with all apps so ITSI can access them. These applications are only required if you plan to use Predictive Analytics to predict service health scores. For more information, see Set up Predictive Analytics in ITSI.

1. Install the Python for Scientific Computing add-on version 1.3 or later for your operating system:
   - Mac
   - Linux 64-bit
   - Linux 32-bit
   - Windows 64-bit
2. Install the latest version of the Splunk MLTK. Follow the steps in Install the Splunk Machine Learning Toolkit in the MLTK User Guide.
3. Give the MLTK app Global permissions:
   1. In ITSI, click **App: IT Service Intelligence > Manage Apps.**
   2. In the filter bar, enter Splunk Machine Learning Toolkit.
   3. Click **Permissions.**
   4. Ensure that All apps is selected.
   5. Click **Save.**
Install ITSI on a single instance

Install ITSI on a single Splunk Enterprise instance. In a single-instance deployment, a single Splunk Enterprise instance serves as both search head and indexer.

You must install ITSI by extracting the ITSI installation package. ITSI does not support installation using the app manager in Splunk Web or using the `splunk install app` command at the command line.

1. Log in to splunk.com with your Splunk.com user name and password.
2. Download the latest Splunk IT Service Intelligence product.
3. Stop `splunk`. For example:

   ```bash
cd $SPLUNK_HOME/bin
./splunk stop
```
4. Extract the ITSI installation package into `$SPLUNK_HOME/etc/apps`. For example:

   ```bash
tar -xvf splunk-it-service-intelligence_<latest_version>.spl -C $SPLUNK_HOME/etc/apps
```

   On Windows, rename the file extension from .spl to .tgz first and use a third-party utility like 7-Zip to perform the extraction.
5. Start `splunk`. For example:

   ```bash
cd $SPLUNK_HOME/bin
./splunk start
```

Install ITSI in a distributed environment

You can install ITSI in any distributed Splunk Enterprise environment. For more information on distributed Splunk Enterprise environments, see Distributed deployment in this manual.

Where to install ITSI

<table>
<thead>
<tr>
<th>Splunk instance type</th>
<th>Supported</th>
<th>Required</th>
<th>Actions required / Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search Heads</td>
<td>Yes</td>
<td>Yes</td>
<td>Install ITSI to all search heads where ITSI is used as described in Install ITSI on a single instance. Search heads must be running Splunk Enterprise 7.0.x</td>
</tr>
<tr>
<td>Splunk instance type</td>
<td>Supported</td>
<td>Required</td>
<td>Actions required / Comments</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------</td>
<td>----------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Indexers</td>
<td>Yes</td>
<td>Yes</td>
<td><strong>SA-IndexCreation</strong> is required on all indexers. For non-clustered distributed environments, copy <strong>SA-IndexCreation</strong> to <code>$SPLUNK_HOME/etc/apps/</code> on individual indexers. Indexers must be running Splunk Enterprise 7.0.x -7.2.x.</td>
</tr>
<tr>
<td>License master</td>
<td>Yes</td>
<td>Yes</td>
<td>Install <strong>SA-ITSI-Licensechecker</strong> and <strong>SA-UserAccess</strong> on any license master in a distributed or search head cluster environment. If a search head in your environment is also a license master, the license master components are installed when you install ITSI on the search heads. <strong>Note:</strong> <strong>SA-ITOA</strong> is no longer required as of ITSI version 3.1.x.</td>
</tr>
<tr>
<td>Heavy Forwarders</td>
<td>Yes</td>
<td>No</td>
<td>ITSI does not contain a data collection component.</td>
</tr>
<tr>
<td>Universal Forwarders</td>
<td>Yes</td>
<td>No</td>
<td>ITSI does not contain a data collection component.</td>
</tr>
</tbody>
</table>

### Distributed deployment feature compatibility

This table describes the compatibility of ITSI with Splunk distributed deployment features.

<table>
<thead>
<tr>
<th>Distributed deployment feature</th>
<th>Supported</th>
<th>Actions required / Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search Head Clusters</td>
<td>Yes</td>
<td>Use the <strong>deployer</strong> to distribute ITSI to search head cluster members. Search heads must be running Splunk Enterprise 7.0.x -7.2.x.</td>
</tr>
<tr>
<td>Indexer Clusters</td>
<td>Yes</td>
<td>Use the configuration bundle method to replicate <strong>SA-IndexCreation</strong> across all peer nodes. On the master node, place a copy of <strong>SA-IndexCreation</strong> in <code>$SPLUNK_HOME/etc/master-apps/</code>.</td>
</tr>
</tbody>
</table>
## Distributed deployment feature

<table>
<thead>
<tr>
<th>Distributed deployment feature</th>
<th>Supported</th>
<th>Actions required / Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deployment Server</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For information about updating search head cluster members, see Use the deployer to distribute apps and configuration updates in the *Distributed Search* manual.

For information about updating peers in an indexer cluster, see Manage app deployment across all peers in the *Managing Indexers and Clusters of Indexers* manual.

### Configure search heads and cluster members to forward data to indexers

In a distributed environment, configure search heads to forward data. ITSI runs KPI searches on search heads and by default stores data in the local `itsi_summary` index. It is considered a best practice to forward data from search heads to indexers.

**See also**

- Forward search head data to indexers
- Forward data from search head cluster members

## Upgrade Splunk IT Service Intelligence

This topic describes how to upgrade Splunk IT Service Intelligence on an on-premises deployment from version 2.6.x or later to version 4.0.x. Splunk Cloud customers work with Splunk Support to coordinate upgrades to IT Service Intelligence.

### Before you upgrade IT Service Intelligence

1. Back up your ITSI KV store data and store those backup files in a secure location. See Backup and restore ITSI data in this manual.
2. (Optional) If you plan to use ITSI’s Predictive Analytics capabilities, install the Python for Scientific Computing add-on and the latest version of the Splunk Machine Learning Toolkit. For more information, see Set up Predictive Analytics in ITSI.
3. Make sure the Splunk admin role inherits from the itoa_admin role. The default settings for admin role inheritance for ITSI are contained in authorize.conf. Problems can occur when these settings have been modified in a local version of the file.

   1. Use the CLI `btool` command and look at the line `importRoles` to make sure `itoa_admin`, `itoa_analyst`, and `itoa_user` are listed. For example: `./splunk btool authorize list role_admin ?debug`.

   2. To add the itoa roles, click Settings > Access Controls > Roles > admin. Under Inheritance, add `itoa_admin`, `itoa_analyst` and `itoa_user` to the selected roles if necessary.

4. Check KV store size limits. The limit of a single batch save to a KV store collection is 50 MB. Check the total amount of data that your services contain, and, if necessary, increase the KV store size limit in limits.conf.

Upgrade Splunk IT Service Intelligence

Upgrade your on-premise ITSI deployment the same way you initially installed ITSI, by extracting the installation package for the new version and copying the necessary components to each instance in your ITSI deployment. See Splunk Enterprise Deployments in this manual.

You must upgrade ITSI by extracting the ITSI installation package. ITSI does not support installation using the app manager in Splunk Web or using the `splunk install app` command at the command line.

The first time you start Splunk Enterprise after installing the new files, a migration script runs to migrate existing ITSI knowledge objects to the new version.

**Upgrade search heads or a single-instance**

On each search head, or on a single-instance deployment, download the `splunk-it-service-intelligence_<latest_version>.spl` install package and extract it into `$SPLUNK_HOME/etc/apps`. See Install ITSI on a single instance for details.

**Upgrade indexers**

For non-clustered distributed environments, copy `SA-IndexCreation` to `$SPLUNK_HOME/etc/apps` on each indexer in your deployment.

If you have an indexer cluster, use the configuration bundle method to replicate `SA-IndexCreation` across all peer nodes. On the master node, place a copy of `SA-IndexCreation` in `$SPLUNK_HOME/etc/master-apps/`. For information about
updating peers in an indexer cluster, see Manage app deployment across all peers in the *Managing Indexers and Clusters of Indexers* manual.

**Upgrade ITSI on a search head cluster**

To upgrade ITSI on a search head cluster, use the **deployer** to distribute the new version of ITSI to search head cluster members (the same way ITSI was deployed on the search head cluster initially). The migration script kicks off on the captain. The upgrade then propagates to all other cluster members.

For information about updating search head cluster members, see Use the deployer to distribute apps and configuration updates in the *Distributed Search* manual.

**Upgrade ITSI license components**

When you upgrade Splunk IT Service Intelligence, you must also upgrade `SA-ITSI-Licensechecker` and `SA-UserAccess` on any **license master** in a distributed or search head cluster environment.

If one of the search heads in your environment is also a license master, the license master components are upgraded when you upgrade ITSI on the search heads.

**Post migration tasks**

After the ITSI migration completes, do the following:

1. In Splunk Web, click **Help > About** to verify that upgrade to ITSI version 4.1.x was successful.
2. Clear the browser cache of the browser you use to access Splunk Web to make sure that you access a fresh version of Splunk Web after upgrading. If you do not clear the browser cache, some pages might fail to load.
3. To initiate the fix for ITSI-1868 concerning entity rules, you need to trigger the service-entity rule change handler. To do this, run the `kvstore_to_json mode 4` option which will regenerate your KPI search schedules.
4. If you have a dedicated license master, remove `SA-ITOA` from the license master since ITSI no longer requires it as of version 3.1.x.
5. Remove unnecessary XML files from the ITSI OS Module. The files have been removed or renamed as of ITSI 4.0.0. Remove the following files from `$SPLUNK_HOME/etc/apps/DA-ITSI-OS/default/data/ui/panels`:
   - `cpu_memory_usage.xml`
   - `memory_free_percent.xml`
Uninstall Splunk IT Service Intelligence

This page describes how to safely remove an existing Splunk IT Service Intelligence (ITSI) deployment. Once you complete these steps you can perform a clean reinstallation of ITSI. See Install Splunk IT Service Intelligence in this manual.

High level steps:

1. Clean the KV store.
2. Delete all ITSI entries in collections.conf.
3. Remove all Splunk apps installed with ITSI.
4. Remove all ITSI-specific indexes.

These steps will permanently delete all data associated with your ITSI deployment. Do not perform these steps unless you are certain that you want to permanently delete your ITSI deployment. If you are uncertain how to proceed, contact Splunk support for guidance.

Step 1: Clean the KV store

Clean the KV store on a standalone search head or license master

On all search heads (and license master, if applicable) clean the KV store. There are two ways to clean the KV store:

Use Splunk CLI:

```bash
$SPLUNK_HOME/bin/splunk clean kvstore -app SA-ITOA
```

OR

Run a curl request to delete each individual SA-ITOA collection. For example:

```bash
$curl ?k ?u admin:changeme ?X DELETE
```
A complete listing of all SA-ITOA collections is available in $SPLUNK_HOME/etc/apps/SA-ITOA/default/metadata/default.meta.

**Clean the KV store in a search head cluster**

To clean the KV store in a search head cluster environment, you can run one of the above options to clean the KV store on a single cluster member. The cluster replicates this action and cleans the KV store on each cluster member. See Configuration methods that trigger replication in the *Distributed Search* manual.

**Step 2: Delete all ITSI entries in collections.conf**

On all search heads, delete all ITSI entries in collections.conf.

1. Edit $SPLUNK_HOME/etc/system/local/collections.conf.
2. Delete all entries whose stanza name starts with itsi_*.

**Step 3: Remove all Splunk apps installed with ITSI**

Remove all Splunk apps and add-ons installed with the current or previous versions of ITSI.

Do not remove SA-ThreatIntelligence, SA-Ticketing, SA-Utils or Splunk_SA_CIM if they are in use by another app (such as Splunk Enterprise Security or Splunk App for VMware). If these add-ons are in use, you can safely leave them on your server. If you remove them, any dependent apps will not function as expected.

**Remove apps from standalone or non-clustered distributed environments**

1. Stop Splunk.
2. On all search heads and indexers where ITSI and/or dependent apps and add-ons are installed, delete all items installed by the ITSI installation package. For example:

   ```
   cd $SPLUNK_HOME/etc/apps
   rm -rf itsi DA-ITSI-APPSERVER DA-ITSI-DATABASE DA-ITSI-EUEM
   DA-ITSI-LB DA-ITSI-OS DA-ITSI-STORAGE DA-ITSI-VIRTUALIZATION
   DA-ITSI-WEBSERVER itsi SA-IndexCreation SA-ITOA SA-ITSI-ATAD
   SA-ITSI-CustomModuleViz SA-ITSI-Licensechecker SA-ITSI-MetricAD
   SA-UserAccess
   ```

   **Note:** In most cases, SA-IndexCreation is the only ITSI-related item that you must remove from indexers.

For a complete listing of apps and add-ons installed by the ITSI installation package, see About the ITSI install package in this manual.

Note that you should also remove any ITSI modules that have been installed independently from ITSI, such as the Splunk ITSI Module for Application Performance Monitoring for example.

**Remove apps from clusters**

To delete an app from a search head cluster, you must remove it from the configuration bundle on the deployer. When you next push the bundle, each cluster member deletes the app from its own file system. For more information, see Where to place the configuration bundle on the deployer in the *Distributed Search* manual.

To delete an app from an indexer cluster, you must remove the app from the deployment location on the cluster master. For more information, see Update common peer configurations and apps in *Managing Indexers and Clusters of Indexers*.

**Step 4: Remove all ITSI-specific indexes**

Remove all ITSI-specific indexes that SA-IndexCreation places in
$SPLUNK_HOME/var/lib/splunk, including:

anomaly_detection
itsi_grouped_alerts
itsi_notable_archive
itsi_notable_audit
itsi_summary
itsi_tracked_alerts
snmptrapd

For example:

```
cd $SPLUNK_HOME/var/lib/splunk
rm -rf itsi_* anomaly_detection
```

Do not remove any indexes that are currently in use by Splunk Enterprise Security or other Splunk apps, including notable and risk indexes.
ITSI does not currently provide an automatic way to clean up the contents for a distributed deployment. To clean a distributed deployment you must perform these cleaning steps on individual search heads and indexers.

**Use the free ITSI sandbox**

The free 7-day online IT Service Intelligence (ITSI) sandbox available at https://www.splunk.com/itsi shares data about your sandbox usage with Splunk Inc. (“Splunk”). Your consent to Splunk’s collection of this data is a condition to using the free online ITSI sandbox; you cannot opt out. If you do not want Splunk to collect the information described below, please do not use the service. This information is important to help Splunk understand which use cases our customers are most interested in.

**What usage data is collected?**

In addition to contact information, including email address, collected as part of the registration process, Splunk collects the following data from the free online ITSI sandbox:

- login data such as time, user, host, number of logins
- page view data

**How is the data used?**

The data collected from the free online ITSI sandbox is used for various purposes including:

- Reporting on aggregated usage data to gain an understanding of how the product is used.
- Sending email communications. For example, Splunk sends you an email in the middle of the 7-day trial to give you suggestions based on your usage patterns.
Permissions

Configure users and roles in ITSI

Splunk IT Service Intelligence uses the access control system integrated with the Splunk platform. The Splunk platform authorization allows you to add users, assign users to roles, and assign those roles custom capabilities to provide granular, role-based access control for your organization.

ITSI roles

Splunk IT Service Intelligence (ITSI) provides four special roles with predefined capabilities:

**itoa_user**
- Assign this role to users who need basic read access to ITSI. This role can read services, KPIs, and entities as well as glass tables, service analyzers, and deep dives built by others and can create private glass tables, service analyzers, and deep dives. This role can read notable events and create and delete Episode Review custom views.

**itoa_analyst**
- Assign this role to knowledge managers in your organization who will create glass tables, deep dives, and service analyzers and work with notable events in Episode Review.

**itoa_team_admin**
- Create team admin roles that inherit from this role. Team admins can create and administer services for ITSI teams to which they are assigned read/write access. This role can also create and manage notable event aggregation policies.

**itoa_admin**
- Assign this role to ITSI administrators. Users possessing this role can create teams for team administrators to administer as well as create services, service templates, entities, KPI base searches, and KPI threshold templates in the Global team. This role is required to assign permissions to objects such as glass tables to other ITSI roles. Note that users with the Splunk admin role also have the itoa_admin role.

Splunk Enterprise administrators (users with the admin role) can assign users to these roles to grant an appropriate level of access to specific ITSI functions. The role to which you assign a user depends on the specific tasks the user performs.
inside ITSI. Splunk Cloud administrators (users with the role, sc_admin) need to request Splunk support to assign users to the ITSI roles.

You can also create custom roles. If your organization is planning to use service-level permissions (teams), you need to create custom roles that inherit from the provided ITSI roles. See Create custom roles for teams for information.

**ITSI role capabilities**

The following table summarizes ITSI roles, inheritance, and capabilities. ITSI roles inherit from lesser ITSI roles and thus inherit the capabilities of the lesser roles.

<table>
<thead>
<tr>
<th>Role</th>
<th>Inherits from role</th>
<th>Added capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>itoa_user</td>
<td>user, user_ad_user*</td>
<td>• read services, KPIs, and entities</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read service templates</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read KPI base searches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read KPI threshold templates</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read glass tables</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read the default Service Analyzer (homeview)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read deep dives</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete deep dives context (drilldown from Service Analyzer or notable events)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read correlation search</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete notable event management state</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read notable events</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read notable event actions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read team objects</td>
</tr>
<tr>
<td>Role</td>
<td>Inherits from role</td>
<td>Added capabilities</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>itoa_analyst</td>
<td>itoa_user, user, power, user_ad_user*</td>
<td>All capabilities of itoa_user plus the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete glass tables</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete deep dives</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete saved service analyzers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete notable events</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• change notable event status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/execute notable event actions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read notable event aggregation policies</td>
</tr>
<tr>
<td>itoa_team_admin</td>
<td>itoa_analyst, user, power, metric_ad_admin*</td>
<td>All capabilities of itoa_analyst plus the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• configure permissions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete services, KPIs, and entities</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete KPI base searches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete KPI threshold templates</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete correlation search</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete maintenance windows</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read/write/delete modules</td>
</tr>
</tbody>
</table>
### Role Inheritance and Added Capabilities

<table>
<thead>
<tr>
<th>Role</th>
<th>Inherits from role</th>
<th>Added capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>itoa_admin</td>
<td>itoa_team_admin, user, power, metric_ad_admin*</td>
<td>All capabilities of itoa_team_admin plus the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- read/write/delete notable event aggregation policies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- write/delete team objects</td>
</tr>
<tr>
<td>admin</td>
<td>itoa_admin, itoa_analyst, itoa_user, user, power</td>
<td>All</td>
</tr>
</tbody>
</table>

*The user_ad_user and metric_ad_admin roles are inherited by ITSI roles for the purposes of using anomaly detection in ITSI. Do not assign these roles to users separately.

ITSI role capabilities apply only to shared objects. Users assigned to the itoa_user role can create and manage private service analyzers, glass tables, and deep dives.

If you have the itoa_admin or itoa_team_admin role (or the capabilities of these roles) you still need write access to the Global team to write and delete global objects (service templates, entities, KPI templates, base searches, and threshold templates).

### ITSI indexes

All ITSI-specific roles require access to the following ITSI indexes:
• anomaly_detection
• itsi_grouped_alerts
• itsi_notable_archive
• itsi_notable_audit
• itsi_summary
• itsi_tracked_alerts
• snmptrapd (optional, used only if SNMP traps are collected)

Enable/disable ITSI capabilities for a role

You can enable/disable object capabilities for ITSI roles in $SPLUNK_HOME/etc/apps/itsi/local/authorize.conf.

1. Copy the authorize.conf file from the itsi/default directory to the itsi/local directory. For example:

   cd $SPLUNK_HOME/etc/apps/itsi/default
   cp authorize.conf ../local

2. Edit local/authorize.conf to enable or disable the appropriate capabilities for ITSI-specific roles. To disable a capability in authorize.conf replace the word "enabled" with "disabled" or delete the capability from the file.

   For example, the following shows a portion of authorize.conf with
   read_itsi_glass_table = disabled for role_itoa_user:

   ## ITOA User
   ## The ITOA user role inherits user role
   ## This allows users assigned to the itoa_user role to perform all capabilities of a Splunk user
   ## The itoa_user role can also perform RT search
   [role_itoa_user]
   importRoles = user;user_ad_user

   ## Core dependent capabilities
   list_storage_passwords = enabled
   rtsearch = enabled

   # For event management
   edit_token_http = enabled

   ## ITSI specific/controlled capabilities

   # Glass Table
   read_itsi_glass_table = disabled
Create custom roles for teams

If you decide to create teams in ITSI to segment your service-level data, you need to create custom roles that inherit from the standard ITSI roles to enable you to assign permissions to specific roles that correspond to specific teams. See Service-level permissions for information about service-level permissions and teams.

Create a role in the Splunk platform for each ITSI team admin and configure the roles to inherit from the itoa_team_admin role in order to obtain the appropriate capabilities. Then assign users to each team admin role you created.

For example, the Splunk admin creates an itoa_finance_admin role that inherits from the itoa_team_admin role for the administrator of the Finance team. The Splunk admin then assigns the Finance team administrator to the itoa_finance_admin role.

Likewise, create custom roles for the ITSI analysts and users in each team. This allows you to differentiate when assigning permissions to teams. For example, create an itoa_finance_analyst role that inherits from the itoa_analyst role for the analysts in the Finance department and an itoa_finance_user role that inherits from the itoa_user role for the users in the Finance department. The ITSI admin can then assign permissions to the Finance team for the itoa_finance_analyst and itoa_finance_user roles without allowing access to analysts and users from other departments.

You must configure the itoa_admin role to inherit from the custom roles you create otherwise the itoa_admin role will not be able to assign permissions to the custom roles. Alternatively, use the admin role to assign permissions.

Splunk Cloud administrators need to request Splunk Support to create the custom roles for teams.

For information about creating custom roles, see About configuring role-based user access in the Securing Splunk Enterprise manual.
Create other custom roles

If you create a new role that does not inherit from one of the standard ITSI roles, you need to do four things to ensure the custom role has the appropriate level of access in ITSI:

1. Assign the role proper capabilities.
2. Make sure the role has access to the ITSI indexes.
3. Assign the role proper view level access.
4. Assign the role KV store collection level access.

For example, in order to assign a new role write permissions to a deep dive, that new role must first be assigned the write_deep_dives capability. The new role must also have write access to the saved_deep_dives_lister view, and write access to the itsi_pages collection.

Set permissions to ITSI views

ITSI includes default entries in itsi/metadata/default.meta that determine access for ITSI roles to specific ITSI views. By default, only itoa_admin has read/write access to all ITSI views.

Set permissions to ITSI views in Splunk Web

1. In Splunk Web, go to Settings > All configurations.
2. Set the App Context to IT Service Intelligence (itsi). Set the Owner to Any.
3. Select the check box to Show only objects created in this app context. This narrows down the page view to ITSI objects only.
4. In the Sharing column, click Permissions for the specific view.
5. Select the check boxes to grant read and/or write permissions for ITSI roles. Click Save.

This updates the access permissions to ITSI views for ITSI roles in $SPLUNK_HOME/etc/apps/itsi/metadata/local.meta.

Set permissions to ITSI views from the command line

1. Create a local.meta file in the itsi/metadata/ directory.

   cd $SPLUNK_HOME/etc/apps/itsi/metadata
cp default.meta local.meta

2. Edit itsi/metadata/local.meta.
3. Set access for specific roles in local.meta. For example:
Set permissions to KV store collections

SA-ITOA includes default entries in `metadata/default.meta` that determine access to KV store collections for ITSI roles. For a list of default permissions to KV store collections for ITSI roles, see KV store collection permissions. By default, only `itoa_admin` has read/write/delete access to all ITSI KV store collections.

Set permissions to KV store collections in Splunk Web

1. In Splunk Web, go to **Settings > All configurations**.
2. Set the **App Context** to SA-ITOA. Set **Owner** to Any.
3. Select the check box to **Show only objects created in this app context**. This narrows down the page view to SA-ITOA objects only.
4. In the **Sharing** column, click **Permissions** for the specific collection.
5. Select the check boxes to grant read and/or write permissions to the various collections for ITSI roles. Click **Save**.

This updates KV store access permissions for the specific ITSI roles in `$SPLUNK_HOME/etc/apps/SA-ITOA/metadata/local.meta`.

Set permissions to KV store collections from the command line

1. Create a `local.meta` file in the `SA-ITOA/metadata/` directory.

   ```bash
   $ cd $SPLUNK_HOME/etc/apps/SA-ITOA/metadata
   $ cp default.meta local.meta
   
   2. Edit `SA-ITOA/metadata/local.meta`
   
   3. Set access for specific roles in `local.meta`. For example:

   ```
   [collections/itsi_services]
   access = read : [ itoa_admin, itoa_analyst, itoa_user ], write: [ itoa_admin ]
   ```

Using service-level permissions in conjunction with other access controls

See Overview of service-level permissions in ITSI for detailed information about service-level permissions and teams.
In addition to the other access control methods provided in ITSI, teams provide a more granular level of access control. Teams let you restrict read/write access to services and the KPIs associated with services within ITSI views such as glass tables, deep dives, and service analyzers.

For example, a user might have permission to view a particular glass table, but if a KPI in that glass table belongs to a service in a team for which the user does not have read permission, the KPI is not displayed. Only the data related to services for which the user has read access are displayed on the glass table.

To prevent users from being confronted with widgets they cannot view in glass tables or lanes they cannot view in deep dives, keep in mind the intended audience when creating a shared glass table or deep dive and create these visualizations for a particular team.

For example, if you are creating a glass table for the Finance team, create a shared glass table that only includes services and KPIs in the Finance team or Global team and assign read/write permissions for the glass table to the Finance team roles. Then users from other teams won't try to access the glass table and get frustrated when they can't view all of the information.

**Add capabilities to a role in ITSI**

When you create a user in ITSI you assign that user to one role. See Configure users and roles in ITSI for more information.

Each role contains a set of capabilities. You can add or edit capabilities for new, existing, and default roles. For example, you might give a role the capability to create a shared glass table or delete a KPI base search.

For a list of ITSI roles, inheritance, and capabilities, see ITSI role capabilities.

**List of capabilities**

This list shows the capabilities that you can add to any role.

Capabilities are subject to change. For the most up-to-date list of capabilities, see `$SPLUNK_HOME/etc/apps/SA-ITOA/default/authorize.conf`. For information about the capabilities assigned to ITSI roles, see Configure access controls.

A write capability implies create and update. Delete is its own capability.
<table>
<thead>
<tr>
<th>SA-ITOA Object type</th>
<th>Capability name</th>
<th>Capability description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBAC Permissions</td>
<td>configure_perms</td>
<td>Ability to configure Role Based Access Control on shared service analyzers, deep dives, glass tables, correlation searches, and notable event aggregation policies.</td>
</tr>
<tr>
<td>Configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service/KPIs/Entity</td>
<td>read_itsi_service</td>
<td>* Ability to read service-based information in Service Analyzer. * Ability to pull in service based information on a glass table or deep dive. * Listing of services and entities in their lister pages.</td>
</tr>
<tr>
<td></td>
<td>write_itsi_service</td>
<td>* Ability to create a service. * Ability to create a KPI. * Ability to create an entity. * Ability to bulk import entities/services via CSV file or via search and set dependencies.</td>
</tr>
<tr>
<td></td>
<td>delete_itsi_services</td>
<td>Ability to delete a service/KPI/entity.</td>
</tr>
<tr>
<td>Service Templates</td>
<td>read_itsi_base_service_template</td>
<td>Ability to view a service template.</td>
</tr>
<tr>
<td>SA-ITOA Object type</td>
<td>Capability name</td>
<td>Capability description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------</td>
<td>------------------------</td>
</tr>
<tr>
<td></td>
<td><code>write_itsi_base_service_template</code></td>
<td>Ability to create a service template.</td>
</tr>
<tr>
<td></td>
<td><code>delete_itsi_base_service_template</code></td>
<td>Ability to delete a service template.</td>
</tr>
<tr>
<td>KPIs Temporary (KPIs with time policies enabled)</td>
<td><code>read_itsi_temporary_kpi</code></td>
<td>Ability to read a KPI with time policy.</td>
</tr>
<tr>
<td></td>
<td><code>write_itsi_temporary_kpi</code></td>
<td>Ability to create a KPI with time policy.</td>
</tr>
<tr>
<td></td>
<td><code>delete_itsi_temporary_kpi</code></td>
<td>Ability to delete a KPI with time policy.</td>
</tr>
<tr>
<td>KPI Base Searches</td>
<td><code>read_itsi_kpi_base_search</code></td>
<td>Ability to read a KPI base search.</td>
</tr>
<tr>
<td></td>
<td><code>write_itsi_kpi_base_search</code></td>
<td>Ability to write a KPI base search.</td>
</tr>
<tr>
<td></td>
<td><code>delete_itsi_kpi_base_search</code></td>
<td>Ability to delete a KPI base search.</td>
</tr>
<tr>
<td>KPI Threshold Templates</td>
<td><code>read_itsi_kpi_threshold_template</code></td>
<td>Ability to read KPI threshold template type objects.</td>
</tr>
<tr>
<td></td>
<td><code>write_itsi_kpi_threshold_template</code></td>
<td>Ability to write a custom KPI threshold template.</td>
</tr>
<tr>
<td></td>
<td><code>delete_itsi_kpi_threshold_template</code></td>
<td>Ability to delete a KPI threshold template.</td>
</tr>
<tr>
<td>Backup/Restore</td>
<td><code>read_itsi_backup_restore</code></td>
<td>Ability to read backup/restore page.</td>
</tr>
<tr>
<td></td>
<td><code>write_itsi_backup_restore</code></td>
<td>Ability to create a backup/restore job.</td>
</tr>
<tr>
<td></td>
<td><code>delete_itsi_backup_restore</code></td>
<td>Ability to delete a backup/restore job.</td>
</tr>
<tr>
<td>Glass Table</td>
<td><code>read_itsi_glass_table</code></td>
<td>Ability to view</td>
</tr>
<tr>
<td>SA-ITOA Object type</td>
<td>Capability name</td>
<td>Capability description</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td></td>
<td>write_itsi_glass_table</td>
<td>Ability to create a shared glass table.</td>
</tr>
<tr>
<td></td>
<td>delete_itsi_glass_table</td>
<td>Ability to delete a shared glass table.</td>
</tr>
<tr>
<td></td>
<td>interact_with_itsi_glass_table</td>
<td>Ability to drilldown and interact with glass tables.</td>
</tr>
<tr>
<td>Deep Dive</td>
<td>read_itsi_deep_dive</td>
<td>Ability to view a shared deep dives.</td>
</tr>
<tr>
<td></td>
<td>write_itsi_deep_dive</td>
<td>* Ability to create a shared deep dive. * Ability to create a shared deep dive as a clone from a private deep dive.</td>
</tr>
<tr>
<td></td>
<td>delete_itsi_deep_dive</td>
<td>Ability to delete a shared deep dive.</td>
</tr>
<tr>
<td></td>
<td>interact_with_itsi_deep_dives</td>
<td>Ability to drilldown and interact with deep dives.</td>
</tr>
<tr>
<td></td>
<td>read_itsi_deep_dive_context</td>
<td>Ability to drill down to an automatically generated (unnamed) deep dive object.</td>
</tr>
<tr>
<td></td>
<td>write_itsi_deep_dive_context</td>
<td>Ability to drill down to an automatically generated (unnamed) deep dive object for the first time.</td>
</tr>
</tbody>
</table>
|                     | delete_itsi_deep_dive_context    | Ability to delete an automatically generated (unnamed) deep dive.
<table>
<thead>
<tr>
<th>SA-ITOA Object type</th>
<th>Capability name</th>
<th>Capability description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>inter</td>
<td>_with</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service Analyzer</td>
<td>read_itsi_homeview</td>
<td>Ability to read service analyzer type object. Triggered on opening the Service Analyzer page (or the ITSI app).</td>
</tr>
<tr>
<td></td>
<td>write_itsi_homeview</td>
<td>Ability to write a service analyzer type object. Triggered on opening the Service Analyzer page (or the ITSI app) for the first time.</td>
</tr>
<tr>
<td></td>
<td>delete_itsi_homeview</td>
<td>Ability to delete a service analyzer type object. Never triggered.</td>
</tr>
<tr>
<td></td>
<td>interact_with_itsi_homeview</td>
<td>Ability to drilldown and interact with service analyzer.</td>
</tr>
<tr>
<td>Correlation Search</td>
<td>read_itsi_correlation_search</td>
<td>Ability to read correlation searches.</td>
</tr>
<tr>
<td></td>
<td>write_itsi_correlation_search</td>
<td>Ability to write a correlation search.</td>
</tr>
<tr>
<td></td>
<td>delete_itsi_correlation_search</td>
<td>Ability to delete a correlation search.</td>
</tr>
<tr>
<td></td>
<td>interact_with_itsi_correlation_search</td>
<td>Ability to interact with a correlation search.</td>
</tr>
<tr>
<td>SA-ITOA Object type</td>
<td>Capability name</td>
<td>Capability description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Event Management State</td>
<td>read_itsi_event_management_state</td>
<td>Ability to read Episode Review dashboards.</td>
</tr>
<tr>
<td></td>
<td>write_itsi_event_management_state</td>
<td>Ability to save an Episode Review dashboard.</td>
</tr>
<tr>
<td></td>
<td>delete_itsi_event_management_state</td>
<td>Ability to delete an Episode Review dashboard.</td>
</tr>
<tr>
<td>Notable Event</td>
<td>read-notable_event</td>
<td>Ability to read a notable event.</td>
</tr>
<tr>
<td></td>
<td>write-notable_event</td>
<td>Ability to modify a notable event on index. Requires delete_by_keyword and edit_token_http capabilities to be enabled.</td>
</tr>
<tr>
<td></td>
<td>delete-notable_event</td>
<td>Ability to delete an episode.</td>
</tr>
<tr>
<td>Notable Event Aggregation Policy</td>
<td>read_itsi_notable_event_aggregation_policy</td>
<td>Ability to read a notable event aggregation policy.</td>
</tr>
<tr>
<td></td>
<td>write_itsi_notable_event_aggregation_policy</td>
<td>Ability to write a notable event aggregation policy.</td>
</tr>
<tr>
<td></td>
<td>delete_itsi_notable_event_aggregation_policy</td>
<td>Ability to delete a notable event aggregation policy.</td>
</tr>
<tr>
<td></td>
<td>edit_default_itsi_notable_aggregation_policy</td>
<td>Ability to edit the default notable event aggregation policy.</td>
</tr>
<tr>
<td></td>
<td>interact_with_itsi_notable_aggregation_policy</td>
<td>Ability to interact with notable event aggregation policy.</td>
</tr>
<tr>
<td>SA-ITOA Object type</td>
<td>Capability name</td>
<td>Capability description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Episode actions</td>
<td>read-notable_event_action</td>
<td>Ability to read an episode action.</td>
</tr>
<tr>
<td></td>
<td>execute-notable_event_action</td>
<td>Ability to run an episode action.</td>
</tr>
<tr>
<td>Maintenance services</td>
<td>read-maintenance_calendar</td>
<td>Ability to read a maintenance calendar.</td>
</tr>
<tr>
<td></td>
<td>write-maintenance_calendar</td>
<td>Ability to write a maintenance calendar.</td>
</tr>
<tr>
<td></td>
<td>delete-maintenance_calendar</td>
<td>Ability to delete a maintenance calendar.</td>
</tr>
<tr>
<td>ITSI Module interface</td>
<td>read-module_interface</td>
<td>Ability to view the modules on the ITSI Modules lister page and read KPIs provided by modules when creating services.</td>
</tr>
<tr>
<td></td>
<td>write-module_interface</td>
<td>Ability to create an ITSI module and edit KPIs provided by modules.</td>
</tr>
<tr>
<td></td>
<td>delete-module_interface</td>
<td>Ability to delete an ITSI module and delete KPIs provided by modules.</td>
</tr>
<tr>
<td>CSV Import mod input</td>
<td>edit_modinput_itsi_csv_import</td>
<td>Ability to save the modular input for CSV import.</td>
</tr>
<tr>
<td>Teams</td>
<td>read_itsi_team</td>
<td>Ability to read the objects in a team.</td>
</tr>
<tr>
<td></td>
<td>write_itsi_team</td>
<td></td>
</tr>
<tr>
<td>SA-ITO A Object type</td>
<td>Capability name</td>
<td>Capability description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------</td>
<td>------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ability to create or update the objects in a team.</td>
</tr>
<tr>
<td></td>
<td>delete_itsi_team</td>
<td>Ability to delete the objects in a team.</td>
</tr>
<tr>
<td></td>
<td>bulk_import_service_or_entity</td>
<td>Ability to create services or entities using bulk import.</td>
</tr>
</tbody>
</table>

A role which has a "service" capability has analogous capabilities for the "KPI" and "entity" type objects.

**KV store collection permissions in ITSI**

The table shows default permissions to KV store collections for ITSI roles. By default, only itoa_admin has read/write/delete access to all ITSI KV store collections. SA-ITO A includes default entries in `metadata/default.meta` that determine access to KV store collections for ITSI roles. See Set permissions to KV store collections for information.

<table>
<thead>
<tr>
<th>Collection name</th>
<th>itoa_admin</th>
<th>itoa_team_admin</th>
<th>itoa_analyst</th>
</tr>
</thead>
<tbody>
<tr>
<td>itsi_backfill</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_backup_restore_queue</td>
<td>read/write/delete</td>
<td>read</td>
<td>-</td>
</tr>
<tr>
<td>itsi_base_service_template</td>
<td>read/write/delete</td>
<td>read</td>
<td>read</td>
</tr>
<tr>
<td>itsi_correlation_search</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_entity_relationships</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_entity_relationship_rules</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_event_management</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
</tr>
<tr>
<td>itsi_migration</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_notable_event_aggregation_policy</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_notable_event_comment</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
</tr>
<tr>
<td>itsi_notable_event_tag</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
</tr>
<tr>
<td>itsi_notable_event_ticketing</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
</tr>
<tr>
<td>Collection name</td>
<td>itoa_admin</td>
<td>itoa_team_admin</td>
<td>itoa_analyst</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------</td>
<td>-----------------</td>
<td>--------------</td>
</tr>
<tr>
<td>itsi_pages</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
</tr>
<tr>
<td>itsi_refresh_queue</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_services</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_service_analyzer</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
</tr>
<tr>
<td>itsi_team</td>
<td>read/write/delete</td>
<td>read</td>
<td>read</td>
</tr>
<tr>
<td>itsi_temp_batch_claimed_action_queue</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_temporary_storage</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itsi_user_realnames</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>maintenance_calendar</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>operative_maintenance_log</td>
<td>read/write/delete</td>
<td>read/write/delete</td>
<td>read</td>
</tr>
<tr>
<td>itoa_entity_exchange_entities*</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>itoa_entity_exchange_entity_hash*</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>itoa_entity_exchange_metadata*</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

* These are entity exchange collections which are used for integrating entities from the Splunk App for Infrastructure with ITSI. Only the Splunk admin role has access to these collections.

**Restrict access to objects in ITSI**

ITSI lets you set read/write permissions for the following types of ITSI objects:

- Service analyzers
- Glass tables
- Deep dives
- Episode Reviews
- Correlation searches
- Multi-KPI alerts
- Notable event aggregation policies

These permissions determine which user roles have read or write access to specific objects that have been created, such as a shared service analyzer view, a shared glass table, or a correlation search.

These permissions apply to shared objects (as opposed to private ones). Private objects can be created by a user for use only by that user. To set permissions to
a private service analyzer, glass table, or deep dive, clone and save the object with **Shared in App** permissions. The other object types (Episode Reviews, correlation searches, multi-KPI alerts, and notable event aggregation policies) are shared by default.

**Prerequisites**

- To set permissions for ITSI roles, your role must be assigned the `configure_perms` capability. By default, the `itoa_admin` and `itoa_team_admin` roles have the `configure_perms` capability.
- Before you can set permissions to ITSI objects for a role, the role must have the proper capabilities assigned. For more information, see ITSI role capabilities in this manual.

**Steps**

1. On the lister page for the object type (service analyzer, glass table, deep dive, Episode Review, correlation search, or notable event aggregation policy), click **Edit** under the Actions column for the object and select **Edit Permissions**.
2. Assign read/write permissions to ITSI roles for the object.
3. Click **Save**.

**Overview of service-level permissions in ITSI**

As an ITSI administrator, you can set read/write permissions on services so that certain user roles cannot see or modify certain services. This enables you to restrict service-level information in ITSI to only the departments or organizations that need access to the information. The use of service-level permissions also enables an ITSI administrator to empower domain experts in different areas within the organization to create and monitor the services in ITSI that pertain to their department or area.

If your organization does not need to restrict service visibility to specific areas within your organization, you do not need to use service-level permissions.

To restrict service-level information in ITSI, you create teams and assign permissions to the teams by role. When you create a service, you specify the team it belongs to. A service can belong to only one team. Users without read access to a team cannot see data from the services within that team in:

- Glass tables
• Service analyzers
• Deep dives
• Correlation searches
• Multi-KPI alerts
• Episode Review

Users with read-only access to a team can see data from services within that team in glass tables, service analyzers, deep dives, correlation searches, multi-KPI alerts, and Episode Review, but cannot edit or delete any of the service-level data. Note that users with restricted access to service-level information may still have the ability to edit other information in a glass table, deep dive or other visualization if they have write permissions for that glass table, deep dive, or other object.

Service-level permissions provide presentation layer security only and not data level security. It is still possible for a user with access to the Splunk search bar to look up ITSI summary index data.

**How service-level permissions differ from other access controls in ITSI**

ITSI is delivered with ITSI-specific roles that are assigned capabilities that control access to different features in ITSI. You can change the capabilities assigned to roles as needed. See List of capabilities for more information.

ITSI also enables you to set read/write permissions for the following ITSI objects:

• Service analyzers
• Glass tables
• Deep dives
• Correlation searches
• Multi-KPI alerts
• Notable event aggregation policies

These permissions determine which user roles have read or write access to specific objects that have been created, such as a shared service analyzer view, a shared glass table, or a correlation search. See Set permissions to ITSI objects for more information.

Teams provide another more granular level of access controls. Teams allow you to restrict read/write access to the underlying objects, such as KPI searches, within these ITSI visualizations. If teams have been created in your organization, a user may have permission to view a particular glass table, but if a KPI in that
glass table belongs to a service in a team for which the user does not have read permission, the KPI will not be displayed. Only the data related to services for which the user has read access are displayed on the glass table.

**Global team**

The following objects are contained within the default Global team:

- Service templates
- Entities
- KPI templates (provided by modules)
- KPI base searches
- KPI threshold templates

These objects can only be created in the Global team. The Global team can also contain services. The Global team cannot be deleted.

The itoa_admin role has read/write permissions to the Global team. All other roles have read permissions. The read permissions allow the objects in the Global team, such as entities, to be used by services in other teams. Only the itoa_admin role can change the permissions on the Global team.

If you do not have a need to restrict service visibility to specific teams in your organization, create all services in the Global team.

**Private teams**

If you want to segregate your ITSI data into separate teams for data privacy or administration purposes, create a private team for each department or area in your organization that requires a separate view of service data. All private teams can view and use the objects in the Global team. This enables the team administrator to create services in the private team that use service templates, entities, KPI base searches, KPI templates, and KPI threshold templates in the Global team. Team admins can also create services in a private team that have a dependency on a service in the Global team. A private team can contain services...
with a dependency on other services within the same team but a service in a private team cannot depend on a service in another private team.

**Example of using private teams to segregate data in ITSI**

As an example, to keep financial data private, you can create a team for the Finance department and assign the services that pertain to the Finance department to this team. Likewise, you can create a team for the Sales department and assign services pertaining to the Sales department to this team.

A Finance admin role can be created to administer the services for the Finance department and a Sales admin role can be created to administer the services for the Sales department. The Finance admin does not have access to the services in the Sales team and the Sales admin does not have access to the services in the Finance team. Both the Finance admin and the Sales admin can read entities, KPI templates, KPI base searches, KPI threshold templates, and service templates in the Global team and use them for the services in their respective teams, but they cannot modify or delete these objects. The department admins can also create services within their respective teams that are dependent on services in the Global team.

Watch the following video for an example of how you can use private teams to restrict service visibility in ITSI.

Video link: "https://www.youtube.com/watch?v=TAylkWoyUf4?rel=0"

**ITSI role for team admins**

The role itoa_team_admin is delivered with ITSI and provides the ability for departmental or area admins to manage services for a team. This role has all of the capabilities of the itoa_admin role with the exception that the itoa_team_admin cannot perform backup/restores, perform bulk imports of entities and services, or create service templates.
The itoa_team_admin role cannot create new teams. Only the itoa_admin role can create teams. The itoa_admin role has read/write access to all private teams that are created as well as to the Global team.

Create custom roles in the Splunk platform for each ITSI departmental or area admin that will manage a team in ITSI. These roles must inherit from the itoa_team_admin role in order to obtain the appropriate capabilities. For more information about this role's capabilities, see Configure ITSI access controls.

For example, the Splunk administrator (using the admin role) can create an itoa_finance_admin role for the admin of the Financial department and an itoa_sales_admin role for the admin of the Sales department. Then the ITSI admin (using the itoa_admin role) can assign read/write permissions to the itoa_finance_admin for the Finance team and read/write permissions to the itoa_sales_admin role for the Sales team. As a result, the itoa_finance_admin has the ability to create services in the Finance team and the itoa_sales_admin has the ability to create services in the Sales team.

Example of leveraging common services in the Global team

The Global team can contain common services shared across all departments. In this scenario, the ITSI administrator (using the itoa_admin role) configures the ITSI deployment and creates services in the Global team that are common and used across departments. Each department in turn gets a view of ITSI services targeted at the specific department.

The users in a department cannot view the services in another department. Each department leverages common dependencies from the basic services. Each department admin (with roles inherited from the itoa_team_admin role) creates ITSI services for their department with a dependency on the basic services as necessary. The itoa_admin provides support to the admins of the dependent
services. The itoa_admin role has full access to view and change the basic services in the Global team as well as Financial, Sales, and Engineering services as needed.

**Upgrading from a previous release of ITSI**

A default Global team is delivered with ITSI version 3.x. All services, entities, KPI base searches, KPI templates, and KPI threshold templates created prior to version 3.0 belong to the Global team by default. Users possessing the itoa_admin role have full read/write access to the Global team. All other user roles have read only access to the Global team. A user with the itoa_admin role can create private teams and move services from the Global team to the private teams as desired.

**Implement service-level permissions in ITSI**

This topic is an overview of the steps to implement service-level permissions in ITSI.

Read ITSI Service-level permissions and decide if you have a need to implement service-level permissions for your organization.

If you do not need to implement service-level permissions, there is no need to create roles based on the itoa_team_admin role that is delivered for the purpose of administering teams. You do not need to create any additional teams. Create all services, entities, and KPI base searches in the default Global team. If you upgrade from a previous version, all services and other service-related objects already exist in the Global team by default.

**Prerequisite**

Determine the teams you need to create in ITSI. You can create teams for technology areas or for different departments within your organization. Create a team for every area that needs a separate view of ITSI service-level data or that needs to be administered independently within ITSI.

**Implement service-level permissions**

1. Create team admin roles to administer each team and assign users to those roles.
2. Create the teams and assign read/write permissions to the team admin roles you created.
3. Create entities and common services in the Global team.
4. Team admins create the services they need in their assigned teams.

Create roles to administer your teams

After you determine the teams you are going to create in ITSI, create roles to administer the services in each team.

Create a role in the Splunk platform for each ITSI team admin and configure the roles to inherit from the itoa_team_admin role in order to obtain the appropriate capabilities. Then assign users to each team admin role you created.

For example, the Splunk admin creates an itoa_finance_admin role that inherits from the itoa_team_admin role for the administrator of the Finance team. The Splunk admin then assigns the Finance team administrator to the itoa_finance_admin role.

Likewise, create custom roles for the ITSI analysts and users in each team. This allows you to differentiate when assigning permissions to teams. For example, create an itoa_finance_analyst role that inherits from the itoa_analyst role for the analysts in the Finance department. Create an itoa_finance_user role that inherits from the itoa_user role for the users in the Finance department. You can then assign permissions to the Finance team for the itoa_finance_analyst and itoa_finance_user roles without allowing access to analysts and users from other departments.

You must configure the itoa_admin role to inherit from the custom roles you create. Otherwise, the itoa_admin role cannot assign permissions to the custom roles. Alternatively, use the admin role to assign permissions.

Splunk Cloud administrators (using the sc_admin role) need to request Splunk Support to create the custom roles needed for teams.

For information about the itoa_team_admin role's capabilities, see Configure ITSI access controls. For information about creating custom roles, see About configuring role-based user access in the Securing Splunk Enterprise manual.

Create teams

After you create team admin roles and assign users to the roles, you're ready to create the teams. Only users with the itoa_admin role can create a team.
When you create a team, assign read/write access to the team admin role you created for the team. For example, if you create a team called Finance, assign read/write permissions to the itoa_finance_admin role. To create a team, Select Configure > Teams > New Team from the top menu bar in ITSI. See Create teams in this manual for more information.

**Create entities and common services in the Global team**

The ITSI administrator creates all entities in the Global team. The ITSI administrator also creates any common services or services that other teams may have a dependency on in the Global team. See Create ITSI Services for information.

By default, only the itoa_admin role has read/write access to the Global team. A user with the itoa_admin role can give write access to the Global team to other ITSI roles. If you want team admins to be able to create and modify objects in the Global team, assign these roles write permission on the Global team.

**Upgrading from a previous version of ITSI**

If you have upgraded ITSI from a previous version that did not contain service-level permissions, all of the existing services and service related objects like entities and KPI base searches are contained in the Global team by default. After creating private teams, you can move any existing services from the Global team to other teams as necessary. See Move a service to another team for information.

**Create services as a team admin**

After creating teams, the team admins that are assigned read/write permissions can create services within their teams. When creating a service, a team admin can assign it to any team for which they have read/write access. ITSI admins (itoa_admin role) can also create services in private teams.

Team admins can access all of the KPI base searches, KPI templates, and entities in the Global team when creating services in their private teams. Team admins can also create dependencies on services in the Global team or within the same team. Service dependencies cannot be created between services in different private teams. See Create ITSI Services for more information.

Team admins (users with a role that inherits from the itoa_team_admin role) cannot do bulk imports of services and entities.
Create teams in ITSI

Create teams if you want to group services by department, organization, or type of service and control access to the services. Only users with the itoa_admin role can create a team. After teams have been created, you can assign services to the appropriate teams.

You do not need to create any teams if you do not need to administer services separately in your organization. In this case, all services belong to the default Global team. See ITSI service-level permissions before creating any teams.

Prerequisite

Before you create a team, you must create the team admin role that will administer the team so that you can assign permissions to the role when creating the team. See Create roles to administer your teams for information.

Steps

1. Click **Configure > Teams**.
   If you do not see this menu item, you do not have permission to configure teams.
2. The Global team (default team) and any other teams that have been created are listed on the Teams page.
3. Click **Create New Team**. The Create Team dialog opens.
4. Type a team name and description.
   Duplicate team names are allowed, but be aware of other team names and use naming conventions to avoid confusion.
5. Assign read or write access to the listed itoa roles as appropriate. The itoa_admin role has read/write permissions by default and these permissions cannot be changed. If a role has write permissions for a team, a user with this role can create and modify services in the team. The user can't delete a service in the team unless the role has the delete capability for a service.

   If you do not see the custom team admin role listed for which you want to assign permissions, make sure the role has been created and inherits from the itoa_team_admin role. If you are logged in using the itoa_admin role, rather than the admin role, also make sure that the itoa_admin role inherits from the custom team admin role and any other custom roles you have created.
When you create a service, you specify the team it belongs to. You can assign existing services to teams on the Services viewer.

To see the services that belong to a team or to review or change the team permissions, click the team name in the Teams lister page.

To delete a team, locate the team in the Teams lister page and click Edit > Delete in the Actions menu. If the team contains services, you are prompted to move the services to another team before the team can be deleted. The Global team cannot be deleted.

The following video gives a demo of creating a team and assigning services to the team:

Video link: "https://www.youtube.com/watch?v=XfW2CGKPpkE?rel=0"

**Move a service to another team**

You must have write permissions for both teams to move a service from one team to another.

To move a service from one team to another:

1. On the Teams lister page, click the name of the team that contains the service.
2. Click the name of the service you want to move.
3. Click the **Settings** tab and select the team to move it to in the Team dropdown.
4. Click **Save**.

Alternatively, you can navigate to the Services lister page under **Configure > Services** and select Edit > Edit Team for the service you want to move.

To move more than one service at a time, go to the Services lister page and select the checkboxes next to the services you want to move. Click **Bulk Actions > Edit Team** and select the team to move the services to.

Moving a service from one team to another team breaks service dependencies if the service is dependent on another service that cannot be accessed from the new team. If a service is dependent on another service within the same team and one of the services is moved to another team, that dependency is broken. A service in a private team cannot have a dependency on a service in another private team.
Service Configuration

Overview of creating services in ITSI

This topic provides an overview of how to create services for your Splunk IT Service Intelligence (ITSI) deployment. For instructions on configuring services, see Configure services in ITSI in this manual.

What is an ITSI service?

An ITSI service is a representation of a real-world IT service that lets you monitor the health of IT systems and business processes. For more information on ITSI services, see ITSI concepts and features in this manual.

After you create and configure a service, you can use ITSI to monitor service health, perform root-cause analysis, set up threshold-based alerts, and track compliance with organizational SLAs (service-level agreements).

Services must be assigned to a team. A service is only visible for roles that are assigned read permissions to the team. A service can only be edited by roles that are assigned read and write permissions to the team. If your organization has decided not to create private teams, all services will reside in the Global team. For information about teams, see ITSI service-level permissions in this manual.

Note: Before you create a service, it is a best practice to define the entities that you want the service to contain. You can then add entities to the service when you configure the service. For more information, see Define entities in ITSI in this manual.

How to create services

There are three ways to create services:

Create a single service in ITSI
   Create new services one at a time in the UI. You can use service templates to quickly configure services.

Import from CSV
   Import new services and link them to service templates from a CSV file. This method lets you import a hierarchy of dependent services with entities already associated. You can also create a modular input that runs
automated recurring imports of the CSV file contents.

Import from search
Add services and link services to service templates from an ITSI module, saved search, or ad hoc search.

Define entities in ITSI

In ITSI, an entity is an IT component that requires management to deliver an IT service. Each entity has specific attributes and relationships to other IT processes that uniquely identify it. Entities are usually hosts, but can also be items as diverse as network devices, applications, users, and cell towers.

ITSI entities can be any of the following components:

- Physical or virtual hosts
- Network devices (switches, routers)
- Users (AD/LDAP)
- Storage systems, volumes
- Operating system processes
- Software application (db, web server, business app)
- Application process instances (for example, 2 instances of the same web server application is 2 separate entities)
- Cell towers

Entities contain information that ITSI uses to associate services with information found in Splunk searches. You can use this entity information to filter items according to the entity definition.

An entity is similar to a "configuration item" in the ITIL framework, but an entity is never a service itself.

Define entities before creating services. When you configure a service, you can specify entity matching rules based on entity aliases that automatically add the entities to your service. For more information, see Configure services in ITSI in this manual.

How to define entities

There are four ways to create entities in ITSI:

- Auto-detect entities using ITSI modules
• Create a single entity
• Bulk import entities from a CSV file
• Import entities from searches

All entities exist in the Global team. Only a user with write permissions to the Global team can create a single entity. Only a user with the itoa_admin role can import entities from CSV files or searches.

Auto-detect entities using ITSI modules

The modules included with ITSI can help automatically discover entities. For example, when a new server comes online, ITSI can automatically add it as an entity. Entity discovery occurs on a scheduled basis (usually every 4 hours) if the modules included with ITSI are properly configured and the add-ons required for data collection are installed and properly configured.

For example, the OS module automatically detects all the servers that are sending data into the Splunk platform using the Splunk Add-on for Unix and Linux or the Splunk Add-on for Microsoft Windows. Entity types such as OS hosts, virtualization hypervisors, VMs, web servers, database servers, and load balancers can all be created and populated as entities in ITSI in this way. For more information, see ITSI module entity discovery in the Splunk IT Service Intelligence Modules manual.

Create a single entity

You can create single entities one at a time in the UI.

1. Click Configure > Entities.

   If your role does not have write access to the Global team, you will not see the Create Entity option.
3. Configure the following fields to define your new entity:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of your new entity.</td>
</tr>
<tr>
<td>Description</td>
<td>(Optional) Enter a brief description of the entity.</td>
</tr>
<tr>
<td>Team</td>
<td>All entities are created in the Global team. This cannot be changed.</td>
</tr>
<tr>
<td>Aliases</td>
<td></td>
</tr>
</tbody>
</table>
Enter field-value pairs that identify the entity. For example:
host=webserver-01
IP=10.2.1.1
MAC=C6:4B:B9:E8:E6:2A.

Add field-value pairs that associate specific attributes with the entity. For example:
role=webserver
owner=Ops

The following characters are not supported in the Aliases and Info Fields: commas (,), single quotes ('), double quotes ("), $ (as first character in field name), period (.), equal sign (=), and spaces ( ).

4. Click Create. Your new entity appears in the Entities lister page.
5. (Optional) Click View Health to view detailed entity information, including the health status of associated services and KPIs.

**Bulk import entities**

ITSI provides two methods for bulk import of entity definitions:

- Import from CSV
  - Import entity definitions from a CSV file.
- Import from search
  - Import entity definitions from ITSI module searches, saved searches, and ad hoc searches.

By default, only a user with the itoa_admin role can perform bulk imports of entities.

**Use cases**

Create entities in ITSI
- Create a CSV file or a search with the results in table format with a column to use for the entity title and optionally, one or more entity alias and entity information columns. The columns do not need to use these exact names as they will be mapped in the import process.

Create entities in ITSI and associate each entity with a service
- Add a Service column containing the name of the service in addition to a column for the entity title and optionally, one or more entity alias and entity information columns. If the service does not already exist, the service will be created. To associate multiple entities to a service, provide multiple rows that contain the same service and the name of each unique entity.
An entity rule will be created for the service with Entity Title matches <value from Entity Title column> for each entity.

For information about creating services and linking them to service templates, see Bulk import services in ITSI.

**Import from CSV**

You can import entities from CSV files that contain one or more entity definitions. Importing entities from CSV files is an efficient way to define multiple entities. You can dump data from a change management database (CMDB) or asset inventory database into a CSV file. The CSV import can be automated for ongoing updates.

In the first row of the CSV, specify column names. In each subsequent row, specify an entity title, as well as one or more entity aliases, and one or more entity information fields. To associate an entity with a service, provide a column with the name of the service.

Import from CSV has a limit of one service and one entity per row. There is no limit on the number of dependent services, entity aliases, or entity rule values per row. A CSV file can contain multiple rows. Import from CSV supports five different separators: comma (,), semicolon (;), pipe (|), tab (\t), and caret (^).

**Prerequisite**

To import entities from CSV, you must create a CSV file that contains your entity definitions. In this example we want to create two entities called **appserver-04** and **appserver-05**, and associate **appserver-04** with the Web A service and associate **appserver-05** with the Web B service. The Web A service already exists in ITSI but the Web B service does not.

<table>
<thead>
<tr>
<th>ITService</th>
<th>service_desc</th>
<th>host</th>
<th>vendor</th>
<th>IP</th>
<th>itsi_role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web A</td>
<td>This is Web A</td>
<td>appserver-04</td>
<td>Linux</td>
<td>10.2.1.133</td>
<td>web_server</td>
</tr>
<tr>
<td>Web B</td>
<td>This is Web B</td>
<td>appserver-05</td>
<td>Microsoft</td>
<td>10.2.1.134</td>
<td>web_server</td>
</tr>
</tbody>
</table>

**Steps**

1. Click **Configure > Entities**.
2. Click **Create Entity > Import from CSV**.

If your role does not have write access to Global team, you will not see the Create Entity button.
3. Click **browse**... and select your CSV file, or drop it into the UI. The CSV file uploads and a file preview appears.

4. Click **Next**.

5. Under **Import Column As**, select the appropriate column type for each column.

<table>
<thead>
<tr>
<th>Column type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity Title</td>
<td>Makes the column entry the entity title.</td>
</tr>
<tr>
<td>Entity Description</td>
<td>Makes the column entry a description of the entity.</td>
</tr>
<tr>
<td>Entity Alias</td>
<td>Makes the column entry a searchable entity identifier.</td>
</tr>
<tr>
<td>Entity Information field</td>
<td>Makes the column entry a tag that provides user-facing validation.</td>
</tr>
<tr>
<td>Service Title</td>
<td>Makes the column entry the name of the service to associate the entity with. The service will be created if it does not already exist.</td>
</tr>
<tr>
<td>Service Description</td>
<td>Makes the column entry the description of the service.</td>
</tr>
<tr>
<td>Do Not Import</td>
<td>Removes the column entry from the imported data.</td>
</tr>
</tbody>
</table>

You can edit the Column Name field for columns that you import as Entity Title, Entity Alias, and Entity Information, for custom field mappings.

For this example, we map the columns in the CSV file to the following fields in ITSI:

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Import Column As</th>
<th>Sample Value From CSV File</th>
</tr>
</thead>
<tbody>
<tr>
<td>service</td>
<td>Service Title</td>
<td>A</td>
</tr>
<tr>
<td>service_desc</td>
<td>Service Description</td>
<td>Title is Web A</td>
</tr>
<tr>
<td>host</td>
<td>Entity Title</td>
<td>appserver-M</td>
</tr>
<tr>
<td>service</td>
<td>Entity Information Field</td>
<td>about</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EL21555</td>
</tr>
<tr>
<td></td>
<td></td>
<td>web.jane</td>
</tr>
</tbody>
</table>

6. In the Settings section, choose a Conflict Resolution option to determine how ITSI will update and store your entity data:
   - **Skip Over Existing Entities**: Adds all new entity data to the datastore.
   - **Update Existing Entities**: Combines two records into a single record.
**Replace Existing Entities:** Replaces existing entity data with new entity data.

For more information about Conflict Resolution, see Conflict Resolution examples below.

7. If you are also importing services, configure the following options in the Settings section:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Team</td>
<td>Select the team to create the services in. (Entities can only be created in the Global team.)</td>
</tr>
<tr>
<td>Import Services As</td>
<td>Choose whether services are enabled or disabled upon import.</td>
</tr>
</tbody>
</table>

8. Click **Import**.

9. Click **View all Entities** and **View all Services** to confirm that your imported entities and services appear.

For example, for service Web A (which already existed), an Entity Title matching rule has been added for `appserver-04`. `appserver-04` is listed in the table of matching entities and the aliases and Information fields have been added to the entity from the CSV file.

We also see that the service Web B has been created and that an Entity Title matching rule has been added for `appserver-05`. We see `appserver-05` listed in the table of matching entities at the bottom and we see the aliases and information fields have been added to the entity from the CSV file.
10. (Optional) Click **Set up Recurring Import** to create a modular input for the CSV file. For more information, see Set up recurring import in this manual.

**Import from search**

ITSI lets you import multiple entities from ITSI module searches, saved searches, or ad hoc searches using any data coming into the Splunk platform. The Import from Search workflow is identical to the Import from CSV workflow, except that you specify a search string instead of uploading a CSV file.

For CMDB integration, you can set up the Splunk platform to directly query the database where the CMDB data is stored so that a Splunk search can be used to import the CMDB data into ITSI as entities. The import from search can be automated for ongoing updates.

1. Click **Create Entity > Import from Search**.

   If your role does not have write access to the Global team, you will not see the Create Entity option.

2. Select one of the following search types:

<table>
<thead>
<tr>
<th>Search Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module</td>
<td>Choose from a list of pre-defined entity discovery searches based on ITSI modules.</td>
</tr>
<tr>
<td>Saved Searches</td>
<td>Choose from a list of pre-defined ITSI saved searches.</td>
</tr>
<tr>
<td>Ad hoc Search</td>
<td>Enter a custom search string.</td>
</tr>
</tbody>
</table>

3. Enter an ad hoc search string or select a pre-defined module search or saved search. Make sure the results are presented in a table. In this example we want to import entities using an ad hoc search.

4. Click the Search icon to view a preview of the search results.
5. Click **Next**.

6. Use the Select Column page to specify how to classify and store the file column entries that define your entities.
   In this example, we select to import the title column as **Entity Title** and the hostname column as **Entity Alias**.

   ![Select Column page](image)

7. In the Settings section, select a conflict resolution option to determine how ITSI will update and store your entity data:
   - **Skip Over Existing Entities**: Adds all new entity data to the datastore.
   - **Update Existing Entities**: Combines two records into a single record.
   - **Replace Existing Entities**: Replaces existing entity data with new entity data.

   For more information on Conflict Resolution see Conflict Resolution examples below.

8. In the **Preview** section, click **Entities to be imported** to confirm that your entity import configuration is correct.

   ![Preview section](image)

9. Click **Import**.
   A message appears confirming that the import is complete.

10. Click the **View all Entities** link to confirm your imported entities appear in the Entity viewer page.

11. (Optional) Click **Set up Recurring Import** to create a modular input for the CSV file. See Set up recurring import below.

For an example of importing both entities and services, see the Import from CSV section.

### Set up recurring import

After the bulk import process is complete, ITSI gives you the option of creating a modular input that repeats the import function on a recurring basis. This is convenient if you want to add or update entities or services without repeating the entire import from CSV or search workflow.

In a search head cluster environment, you cannot enable the modular input for the recurring import from the UI as described in the steps below. Configure
modular inputs on individual search head cluster members. To do so, copy the inputs.conf file that you've created on a search head into shcluster/apps manually and let the deployer push it to cluster peers. Note that when you redeploy the configuration via the deployer, the configuration is overwritten. If you place the CSV modular input on a single search head using the deployer, the modular input works, but logs will show error messages on the machines where the modular input does not exist.

1. After the import from CSV or search process is complete, click **Set up Recurring Import**.
2. Provide a name for the recurring input.
3. In the case of Import from CSV, enter the full path to the CSV file on the server.

   The CSV file must be on the same server as your ITSI installation.
4. Set the scheduled time to run the import.
5. Click **Submit**. ITSI creates a new modular input of type "IT Service Intelligence CSV Import".

For more information, see Modular inputs overview in the *Developing views and apps for Splunk Web* manual.

If you configure a search-based modular input via the deployer, and you use the append mode, duplicate entities will be created.

**Conflict resolution examples**

When you import entities from CSV or search, you must select a conflict resolution type. This determines how ITSI updates and stores your entity data. The following examples demonstrate the behavior of each conflict resolution type.

**Skip Over Existing Entities:** Adds all new entity data to the datastore. For example, if we import the following CSV file, all of the entity data in the file is added to the datastore:

```
title, IP, host
Server_1, 192.168.1.1, somehostName
Server_2, 192.168.2.2, anotherhostName
```

**Update Existing Entities:** Combines the two records into a single record, merging the attributes (aliases and identifiers) of both. For example, if we import this entity definition:

```
title, IP, host
Server_1, 192.168.1.1, somehostName
Server_2, 192.168.2.2, anotherhostName
```
Server_1, 192.168.1.1, somehostName
Server_2, 192.168.2.22, anotherhostName

Server_2 will now have 2 IPs (192.168.2.2 and 192.168.2.22), while no new data is added to Server_1.

**Replace Existing Entities:** Replaces existing entity data with new entity data. For example, if we import this updated entity definition:

```
title, IP, host
Server_1, 192.168.1.11, somehostName
Server_2, 192.168.2.2, anotherhostName
```

The Server_1 IP (192.168.1.11) in our new entity definition replaces the Server_1 IP (192.168.1.1) in our existing entity definition, and the existing Server_1 IP is removed.

**About duplicate entity aliases**

Entity aliases must uniquely identify the entity. For example, host=appserver-01. The same alias field value cannot be used for more than one entity. If more than one entity is using the same alias field value (such as appserver-01), this can cause incorrect statistical aggregation results for KPI base searches.

For information on how to find and fix duplicate entity aliases, see Duplicate entity aliases in the Troubleshooting section.

**Create a single service in ITSI**

When you create a service, you have three options for populating the service:

- Manually add the KPIs and entity rules.
- Link the service to a service template to populate the KPIs and entity rules from the service template. See Create a service template in ITSI for information about service templates.
- Select prebuilt KPIs from ITSI modules to populate KPIs and entity rules.

**Prerequisite**

By default, users assigned the itoa_admin role or a role that inherits from the itoa_team_admin role can create a new service. You can only create a service in a team for which you have read and write permissions.
Steps

1. Click **Configure > Services**.
2. Click **New Service > New Service**.

If your role does not have write access to any teams, you will not see the New Service option.

3. Provide a title and description for the service.

4. Select the team to associate the service with. This service will only be visible for roles that are assigned read permission to this team. If no private teams have been created for your organization, the only team available will be Global and this is where you will create all services.

5. Select one of the following options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manually add service content</td>
<td>Manually add entities, KPIs, and other information for this service.</td>
</tr>
<tr>
<td>Link service to a service template</td>
<td>Populate the service with the entity rules and KPIs from a service template.</td>
</tr>
<tr>
<td>Add prebuilt KPIs from modules</td>
<td>Populate the service with the entity rules and pre-built KPIs from ITSI modules.</td>
</tr>
</tbody>
</table>

6. If you selected **Link service to a service template**, do the following:
   1. Select the template to link the service to.
   2. Review the entity rules and KPIs in the template. These will be added to the new service.
   3. (Optional) Select **Enable 7 days of backfill for all Service KPIs** if you need to populate the summary index with data in order to use adaptive thresholding or anomaly detection with any of your KPIs. This setting applies to all the KPIs in the service. Individual KPIs cannot be selected for backfill. Note that you cannot enable backfill for a KPI that came from the template in the service later on as this action will unlink the service from the template.

7. If you selected **Add pre-built KPIs from modules**, do the following:
   1. Select any number of ITSI modules from the list. Before using pre-built KPIs, you must configure the specific ITSI module. See About ITSI modules in the ITSI Modules manual.
   2. Select the KPIs to add to the service. Recommended KPIs are pre-selected. Each KPI template populates the new service with specific entity rules and selected KPIs.

8. Click **Create**.
After creating the service, see Overview of configuring services in ITSI to decide how to configure the newly created service.

**Overview of configuring services in ITSI**

This topic provides an overview of how to configure IT Service Intelligence (ITSI) services. The instructions assume you have already created a service. If you have not yet created a service, see Overview of creating services in ITSI in this manual.

**Service configuration workflow**

The following diagram shows the steps involved in configuring an ITSI service.

Many of these steps are completed for you if you created a service from a service template or used pre-built KPIs from modules.

**Configure a service created from a service template**

If you created a service using a service template, the entity rules and KPIs are populated for you. See the following topic:

- Configure a service created from a service template in ITSI

**Configure a service created from pre-built KPIs**

If you created a service using pre-built KPIs from modules, the entity rules and KPIs are populated for you.
Steps

1. Apply anomaly detection to a KPI in ITSI (optional)
2. Add service dependencies in ITSI (optional)

Configure a service manually

You can choose to manually add service content.

Steps

1. Add entity rules to a service in ITSI (optional)
2. Add KPIs to a service in ITSI
3. Apply anomaly detection to a KPI in ITSI (optional)
4. Add service dependencies in ITSI (optional)

Add entity rules to a service in ITSI

Entity rules let you dynamically filter KPI searches based on entity alias matches. You can use entity rules to associate entities with KPIs at the service level, which makes it unnecessary to specify entity identifying fields for each KPI search.

When to add entity rules

Adding entity rules to a service is optional and you can add them at any time. Add entity rules if you want to be able to filter a KPI by the entities in the service. There are many scenarios where entity rules can make it easier to configure your services, including the following:

- You want to match entity ID data not recognized inside Splunk Enterprise (such as mapping a naming scheme to specific devices). For example, your organization might use a server naming convention such as server-01, server-02, and so on. These names do not appear as fields inside Splunk searches. Adding rules that match your entity aliases to your server naming scheme lets you apply KPI searches to those servers.
- You want to disambiguate between multiple fields that identify the same machine (such as a host with multiple IP addresses).
How to set up entity rules

You can set up entity rules to match entities based on entity aliases, info, or entity title. You can also create rules based on multiple AND/OR conditions.

For example, if you want to add entity rules that identify your database servers, and those servers have aliases of host=mysql-01, host=mysql-02, host=mysql-03 and so on, you can add an entity rule such as "host matches mysql*" to identify the servers on which to run the KPI search.

This entity rule matches the host field in Splunk data with your mysql* servers and adds each server to all KPI searches in the service.

Entity rule values can be left blank. For example you could specify "web_server does not match" and leave the value field empty to include all values for the web_server field.

Add service dependencies in ITSI

After you configure a service, you can add other services as impacting services, or service dependencies. Impacting services have some sort of effect or influence on the primary service. Adding service dependencies can help you detect if one service is having a negative impact on another service, and can be useful in performing root cause analysis.

For example, you might have a web service that has a dependency on a database service. By adding the database service as a dependency, you can monitor the impact that the database service is having on the web service.

When you add a service dependency, you select the specific KPIs from the impacting service that you want ITSI to include in the health score calculation of the primary service.
Avoid creating circular service dependencies.

1. Click the **Service Dependencies** tab in the service definition.
2. Click **Add Dependencies**.
3. Select the service(s) that you want to add as a dependency. You can choose services within the same team as the primary service or from the Global team.
4. Select the specific impacting KPIs that you want to monitor as part of the primary service’s health score.

5. Click **Done**.
6. Click **Save** to save the dependency into the service definition.

   ITSI adds the selected KPIs to the list of service dependencies, and includes the selected KPIs in the primary service health score calculation.

---

**Set importance values for service dependencies**

ITSI lets you adjust the importance value of impacting service KPIs in the Settings tab of the service as described in Set KPI importance values in ITSI. Set the importance value to give impacting KPIs appropriate weighting in health score calculations and to generate health score values that more accurately reflect the impact of the dependency on the primary service.

By default, impacting service health scores have an importance value of 11. Regular KPIs have the same importance value that is set for the KPI in the original service.

When an impacting KPI with an importance value of 11 reaches critical, the health score of the primary service will read Critical with a value of 0, regardless of the status of other KPIs.
Enable and disable services

You must enable a service to run service KPIs. When you enable a service, all KPI searches associated with that service start to run in the background. You can enable a service when you first create it in the service configuration workflow, or at a later time from the Settings tab of the service definition or the Services viewer.

You can disable a service to stop KPI searches at any time. Disabling a service is useful if you are still configuring a service and want to avoid generating inaccurate health scores and false positives. It is also useful for reducing search load for performance purposes.

When you disable an impacting service, it has no impact on the health of an enabled parent service.

To enable or disable an existing service:

1. Click Configure > Services.
2. In the Status column of the service, click the toggle for Enabled or Disabled.

If you disable a service that has KPI backfill enabled, the backfill will continue and those KPIs will have data for the backfill period.

Change the team a service belongs to

You can change the team a service belongs to from the Settings tab in the service definition. In order to move a service from one team to another, you must have write permissions for both teams.

Changing the team a service belongs to might break service dependencies. Check to see if the service has a dependency on another service and note the team that the dependent service belongs to before moving the service to another team. If a service is dependent on a service in the Global team, the service dependency will not be broken. However, if a service is dependent on another service within the same team, when one of the services is moved to another team, that dependency will be broken because the dependent service cannot be accessed from the other team. Services in private teams cannot have a dependency on a service in another private team.

To bulk move services to a different team, select the checkboxes on the Services lister page, then click Bulk Action > Edit Team.
Backfill service health scores in ITSI

Backfill service health scores to fill the summary index (itsi_summary) with calculated historical service health scores. Backfill uses past KPI data (including dependent KPIs) to calculate service health scores for a given time period and populate the summary index. In other words, even though the summary index only started collecting service health score data at the start of this week when the service was created, if necessary you can use the backfill option to fill the summary index with data from the past month.

Prerequisites

- You must have the itoa_admin or itoa_team_admin role, or write access to the service, in order to backfill service health scores.
- To ensure that the health score is backfilled correctly, make sure that all KPIs within the service and all dependent KPIs are already backfilled (or have indexed raw data) for at least the amount of time you plan to backfill the service health score.

Steps

1. Click Configure > Services.
2. Open the service whose health score you want to backfill.
3. Click the Settings tab.
4. Under Service Health Score, toggle Backfill.
5. Use the dropdown to select the number of days of data to backfill. You can choose a predefined time range, like last 7 days, or select a custom date prior to the current date. If you choose a specific date, the dropdown dynamically updates with the number of days you’re backfilling to.
6. Click Save to begin the backfill process. A message appears when backfill is completed.

The backfill period is the time range of data that is available after backfill is complete. For example, if you select last 60 days, ITSI fills the summary index with data from the past 60 days. In other words, you now have 60 days of service health score data available.

See also

- Add a KPI to a service in ITSI (see the "Enable Backfill" section for instructions on backfilling KPI data)
- How service health scores work in ITSI
Configure a service created from a service template in ITSI

After creating a service from a service template, review the content the service received from the template. Entity rules, KPIs, KPI threshold values, and KPI importance values will be populated from the template, taking care of most of the service configuration. However, some service configuration must be done directly in the service:

- If the service template contains blank entity rules whose values need to be specified in the service, you must provide those in the service entity rules.
- If you want to enable anomaly detection for any of the KPIs, you must do this in the service.
- If the service has service dependencies, you need to specify these in the service.

To finish configuring the service:

1. Review the entity rules in the Entities tab of the service definition.
   - The entity rules from the template will be added to the service.
   - Any entity rules without values that came from the template contain a 🕵 icon. Provide a value in order to filter the service to a specific set of entities. These entity rule values are meant to be custom for each service. Note that some entity rules may have blank values intentionally; these do not display the notification icon.
   - Add or change entity rules as desired for the service. See Add entity rules to a service in ITSI for information. You can choose to preserve any custom entity rules you define in the service when the service template is updated.

2. In the KPI tab, review the KPIs.
   - The KPIs in the template will be added to the service. These KPIs display a lock icon 🕵️ to indicate that they are linked to the template. You can set custom threshold settings and enable anomaly detection for linked KPIs as desired. See Apply anomaly detection in ITSI and the "Set Thresholds" step of Add a KPI to a service in ITSI for information.

Making any other changes to a linked KPI, such as deleting it, changing the name of the KPI, or changing the base search it uses, will cause the service to be unlinked from the template.
- Add any new KPIs that are specific to this service. These KPIs will not be overwritten when the service template is updated. KPIs
3. Click the **Service Dependencies** tab to specify any service dependencies. Service dependencies must be specified in the service directly; service dependencies are not provided in the template. See Add service dependencies in ITSI for information.

4. Click the **Settings** tab to enable/disable the service, change the team the service belongs to, or change KPI importance values. Custom KPI importance values for a service, which are used to determine the service health score, can be preserved when the service template is updated. See Set KPI importance values in ITSI for information.

5. Click **Save** to save any changes.

Any changes you make to the service, apply to the service only. Edit the service template if you want to make changes that will be propagated to all services linked to the template.

**Summary of changes that unlink a service from its service template**

When a service is linked to a service template, you generally want to make updates to the template and not the service itself. However, you can customize some attributes of the service that are unique to the service.

You can make the following changes to a service without breaking the link to the service template:

- Add a KPI
- Modify an unlinked KPI (a KPI that does not display a lock icon)
- Update KPI thresholds
- Enable anomaly detection for a KPI
- Update KPI importance values which contribute to the service health score
- Add, modify, or delete entity rules

The following changes to a service will break the link to the service template:

- Modifying the name of a linked KPI
- Modifying the base search of a linked KPI (or modifying any other settings in the Search and Calculate section, including enabling backfill)
- Deleting a linked KPI
Linked KPIs have a lock icon next to the name of the KPI in the KPI tab of the service definition.

If you clone a service that is linked to a template, the clone will also be linked to the service template.

**Clone a service in ITSI**

When you clone a service in ITSI, all entities, KPIs, and settings from the original service are cloned into the new service. Cloning a service does not clone summary index content of KPIs within the service. If the original service is linked to a service template, the cloned service is linked to the same service template. Service dependencies and Predictive Analytics models are not copied to the cloned service.

**Prerequisite**

You must have the write_itsi_service capability to clone a service in ITSI.

**Steps**

1. From the ITSI top menu bar, click **Configure > Services**.
2. In the Actions column of the service you want to clone, click **Edit > Clone**.
3. Provide a title that is different than the title of the original service (for example, Database 2). Two services can not have the same title.
4. Select the team that the cloned service will belong to.
5. Click **Clone**.

**Bulk import services in ITSI**

ITSI provides two methods to bulk import service definitions:

- **Import from CSV**
  - Import service definitions from a CSV file.
- **Import from search**
  - Import service definitions from Splunk searches.

You can use either bulk import method to do the following:

- Create multiple services.
• Add service dependencies.
• Link services to service templates (and provide entity rule values for configurable service template entity rules).
• Add associated entities to services.

If importing services without linking them to service templates, you can import a few thousand services at a time. Do not attempt to import more than a thousand services with links to service templates. If you need to import more than a thousand services and link them to service templates, do the imports in smaller batches. It is recommended to import a maximum of 200-300 services with links to service templates at a time. In general, if a service template has a small number of KPIs and entity rules, a larger number of services can be imported and linked to it. If a service template contains a very large number of KPIs and entity rules, a smaller number of services can be imported and linked to it.

Use cases

Create new services without linking them to service templates

To do this, create a CSV file or a search that contains the services to import. In the first row of a CSV, specify column names. In each subsequent row, specify one service, and, optionally, a service description and one or more service dependencies. Note that the services created through this method will not be populated with KPIs or entities.

Create new services and link them to service templates or link existing services to service templates

When you link a service to a service template, the service will receive the KPIs and entity rules from the template. Note that if a service already exists, any entity rules in the service will be replaced by the entity rules in the service template. For more information about service templates, see Create and manage service templates in ITSI.

Create a CSV file or a search that contains the services to import and the corresponding template to link them to. In the first row of a CSV, specify column names. In each subsequent row, specify one service, and, optionally, a service description and one or more service dependencies, and the name of the service template to link the service to. Also include one or more columns that provide values for the configurable entity rules in the service templates. Before creating the CSV or search, review the entity rules for the service templates you want to link services to in order to determine which entity rules have configurable values so that you can provide those values for each service in the CSV or search.
**Create services with associated entities**

Create a CSV file or search that contains the services and entities to import. An entity rule will be created for the service with Entity Title matches <value from Entity Title column> for each entity. See Bulk import entities for an example.

**Import services, link them to service templates, and import entities**

When you import services and link them to service templates, you must provide one or more columns to use as values for any configurable entity rules in the templates. If you also want to import entities, you may do so, using either a separate column for this purpose or the same column that is used for configurable entity rules. In this case, entities will be created for the column you import as Entity Title but instead of Entity Title matches <value from Entity Title column> entity rules being added to the service (which is the case when not linking a service to a service template), the entity rules from the linked template will be added to the service and any configurable entity rules will use the column(s) you specify as the value for the configurable entity rules.

You can set up a recurring import (via modular input) at the end of the bulk import process.

Only users assigned the itoa_admin role can perform a bulk import of services.

**Import from CSV**

You can create new services by importing the contents of a CSV file. The rows and columns that your CSV file should contain depends on the use case. This section contains an example of importing services and linking them to service templates.

Import from CSV has a limit of one service and one entity per row. There is no limit on the number of dependent services, entity aliases, entity information fields, or entity rule values per row. You can use multiple columns for service dependencies or multiple rows containing the same service and each of its service dependencies. Likewise you you can have multiple rows containing the same service and each entity to add to it. A CSV file can contain multiple rows. Import from CSV supports five different separators: comma (,), semicolon (;), pipe (|), tab (\t), and caret (^).

**Prerequisite**
If you are linking services to service templates, create the service templates first. Service templates cannot be created through the bulk import process; the service templates you want to link services to must already exist in ITSI. See Create and manage service templates in ITSI for information on how to create a service template.

**Steps**

1. Create a CSV file that contains your services. In this example, we want to create five services and link them to two different service templates. **App Service 1 depends on Web A. App Service 2 depends on Web B.** Three columns (applicationname, app_title, and host) provide entity rule values for the services.

<table>
<thead>
<tr>
<th>Service</th>
<th>Service Description</th>
<th>Dependent Service</th>
<th>Template Link</th>
<th>applicationname</th>
<th>app_title</th>
<th>host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web A</td>
<td>This is Web A.</td>
<td>App Service 1</td>
<td>Web Store template</td>
<td><em>Web</em></td>
<td>New_Relic_Bustersp_Web-08775109</td>
<td>webserver*</td>
</tr>
<tr>
<td>Web B</td>
<td>This is Web B.</td>
<td>App Service 2</td>
<td>Web Store template</td>
<td><em>Web</em></td>
<td>New_Relic_Bustersp_Web-05321111</td>
<td>webserver*</td>
</tr>
<tr>
<td>Web C</td>
<td>This is Web C.</td>
<td>App Service 3</td>
<td>Web Store template</td>
<td><em>Web</em></td>
<td>New_Relic_Bustersp_Web-05390022</td>
<td>webserver*</td>
</tr>
<tr>
<td>App Service 1</td>
<td>Microsoft App Service</td>
<td>Middleware template</td>
<td><em>API</em></td>
<td>New_Relic_Bustersp_Web-05212128</td>
<td>appserver*</td>
<td></td>
</tr>
<tr>
<td>App Service 2</td>
<td>Linux App Service</td>
<td>Middleware template</td>
<td><em>API</em></td>
<td>New_Relic_Bustersp_Web-05321111</td>
<td>appserver*</td>
<td></td>
</tr>
</tbody>
</table>

   The following image shows what the final service dependency tree will look like:

2. After you create your CSV file, click **Configure > Services**.
3. Click **New Service > Import from CSV**.
4. Click **browse...** and select your CSV file, or drop it into the UI. The CSV file is uploaded and a file preview displays.
5. Click **Next** to see a table populated by the content of your CSV file.
6. In the **Import Column As** column, choose how to categorize each column in your file.

<table>
<thead>
<tr>
<th>Column type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Title</td>
<td>Makes the column entry the name of the service.</td>
</tr>
<tr>
<td>Service Description</td>
<td>Makes the column entry a description of the service.</td>
</tr>
<tr>
<td>Dependent Services</td>
<td>Makes the column entry a dependent service.</td>
</tr>
<tr>
<td>Service Template Link</td>
<td>Makes the column entry the service template to link the service to. Note that the service template must already exist in ITSI.</td>
</tr>
<tr>
<td>Column type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Do Not Import</td>
<td>Removes the column entry from the imported data. Use this for columns containing entity rule values if you are linking services to service templates. You will specify the column to use for entity rules later in the workflow. For our example, we are importing the columns in the CSV file as follows. Note that applicationname, app_title, and host are not being imported. This is because these columns contain entity rule values that are different for each service. They will be used in the Define Entity Rules step later.</td>
</tr>
</tbody>
</table>

7. In the **Settings** section, configure the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Team</td>
<td>Select the team to create the services in.</td>
</tr>
<tr>
<td>Import Services As</td>
<td>Choose if you want services enabled or disabled upon import. KPI searches will not run if the service is disabled.</td>
</tr>
<tr>
<td>Enable 7 days of backfill for all service KPIs</td>
<td>This setting appears if you have selected to import columns as Service Title and Service Template Link. This backfills the KPIs in imported services that are linked to service templates. Enable this setting if you do not have at least seven day’s worth of data in the ITSI summary index and you want to be able to use anomaly detection or adaptive thresholding for any of the KPIs in the service template. Note that backfill cannot be enabled for individual KPIs in service templates or services.</td>
</tr>
</tbody>
</table>

8. Use the Preview section to confirm that your service import configuration is correct.

9. Click **Next**.

10. Perform one of the following tasks:

   ♦ If you selected to import a column as a Service Template Link, go to the Define Entity Rules step (Step 11).
If you are not linking services to service templates, skip to the Import step (Step 14).

11. In the Define Entity Rules step, select a service template to complete the template entity rules.

For the templates that have configurable entity rules, the Entity Rules Complete column displays "Incomplete".

In this example, the Web Store template has four entity rules. One of them has a value defined in the template: itsi_role matches operating_system_host. Every service linked to this template will receive this field value pair. The other three entity rules have values in the service template of matches a value to be defined in the service. Therefore the value must be provided for each service in this step using a column from the CSV file.

12. Click Select a column to provide the entity rule value for each configurable entity rule in the template as shown in the example below.

13. Preview the entities for each service linked to the template. Note that only existing entities in ITSI can be matched. Any entities being imported will not be displayed in the preview.

14. Click Import. A message confirms that the import is complete.

15. Click View All Services to confirm your imported services now appear on the Service viewer page. If you linked services to templates, the Service Template column indicates that it's synced with the service template. The KPIs and entity rules for the services are populated from the template and any configurable entity rules that came from the template contain the value from the column you specified in the import.

16. (Optional) Click Set up Recurring Import to create a modular input for the CSV file. See Set up recurring import in this manual.
Import from search

You can import multiple services from ITSI module searches, saved searches, and ad hoc searches. The Import from Search workflow is identical to the import from CSV workflow, except that you specify a search string instead of uploading a CSV file.

The example in this section shows you how to import services and entities and associate the entities with the services. For an example of importing services and linking them to service templates, see the Import from CSV section.

Prerequisite

If you are linking services to service templates, create the service templates first. Service templates cannot be created through the bulk import process. The service templates must already exist in ITSI. See Create and manage service templates in ITSI for information on how to create a service template.

Steps

1. Select New Service > Import from search.
2. Select one of the following search types:

<table>
<thead>
<tr>
<th>Search Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modules</td>
<td>Choose from a list of pre-defined entity discovery searches based on ITSI modules.</td>
</tr>
<tr>
<td>Saved Search</td>
<td>Choose from a list of pre-defined ITSI saved searches.</td>
</tr>
<tr>
<td>Ad hoc Search</td>
<td>Enter a custom search string.</td>
</tr>
</tbody>
</table>

3. Enter an ad hoc search string or select a predefined module search or saved search. Make sure the results are presented in a table.
   For this example we are using an ad hoc search to import services and entities.
4. Click the Search icon.
   The specified search runs and a Search Results Preview appears.
   We want to create two services called Splunk Search Head Service and Splunk Indexer Service (using the service_title column) and create three entities for the hosts listed in the host column and add these entities to the
services.

5. **Click Next.**

   The Select Columns page appears, showing a table populated by your search results. Use this page to specify how to classify and store the file column entries that define your entities.

6. For our example, we are importing the columns in our search results as follows.

7. **In the Settings section, configure the following items:**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Team</td>
<td>Select the team to create the services in. Entities are always created in the Global team.</td>
</tr>
<tr>
<td>Import Services As</td>
<td>Choose if you want services enabled or disabled upon import.</td>
</tr>
<tr>
<td>Conflict Resolution</td>
<td>This determines how ITSI updates and stores your entity data:</td>
</tr>
<tr>
<td></td>
<td>◆ <strong>Skip Over Existing Entities:</strong> Adds new entity data to the datastore only if the entity does not already exist. If an entity exists, the information is not updated.</td>
</tr>
<tr>
<td></td>
<td>◆ <strong>Update Existing Entities:</strong> Merges the imported data and the existing data associated with the entity.</td>
</tr>
<tr>
<td></td>
<td>◆ <strong>Replace Existing Entities:</strong> Replaces existing entity data with new entity data.</td>
</tr>
<tr>
<td>Conflict Resolution Field</td>
<td>This is the field that is used to identify entities. Entities that have the same field value are considered the same entity. If Conflict Resolution is set to <strong>Update Existing Entities</strong> or <strong>Replace Existing Entities</strong>, ITSI resolves duplicate entities based on this field.</td>
</tr>
</tbody>
</table>
For more information about Conflict Resolution, see Conflict Resolution examples.

8. In the Preview section, click Services to be imported to confirm that your service import configuration is correct. Also check the entities to be imported if you are also importing entities.

9. Click Import.
A message appears confirming that the import is complete.

10. Click View All Services or View all Entities to confirm that your imported services and entities now appear on the respective Service or Entity lister page.

11. (Optional) Click Set up Recurring Import to create a modular input that runs the import from search job on a recurring basis. See Set up recurring import.

How service health scores work in ITSI

ITSI generates a health score for each service that you create. The health score is a good indicator of the status of a service and is a useful metric to display in Service Analyzer, glass tables, and deep dive visualizations. A decline in the service health score value can be the first sign of an issue that might lead to an outage. ITSI continuously monitors and updates service health scores.

Service health score calculations

Service health scores range from 0 to 100, with 0 being most critical and 100 being most healthy. The health score calculation is based on the current severity level of service KPIs (Critical, High, Medium, Low, and Normal) and the weighted average of the importance values of all KPIs in a service. See Set KPI importance values in ITSI.

The "Info" severity level is not included in the service health score calculation.

The following formula is used to calculate service health scores:

\[
\text{Service Health Score} = \sum_{\text{KPI}} K_x \cdot \frac{G_x}{\sum_{y=1} G_y}
\]
Where:

- $N$ = count of KPIs
- $G$ = importance value of one KPI
- $K$ = the severity of the KPI (assuming normal=100, low=70, medium=50, high=30, critical=0)

For example, if you set KPI importance values as follows:

<table>
<thead>
<tr>
<th>KPI Title</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI1</td>
<td></td>
</tr>
<tr>
<td>KPI2</td>
<td></td>
</tr>
<tr>
<td>KPI3</td>
<td></td>
</tr>
</tbody>
</table>

The service health score is calculated as follows:

$$
\text{Service Health Score} = \left( \frac{100 \times 10}{22} \right) + \left( \frac{70 \times 7}{22} \right) + \left( \frac{30 \times 5}{22} \right) = 45.45 + 22.27 + 6.81 = 74.53
$$

**Impact of per-entity thresholds on service health scores**

When a KPI is split by entity, if any entity has a severity level (based on entity thresholds) that is worse than the service aggregate severity, the service health score will be impacted. In some cases, this can cause the overall service health score to change significantly, while the aggregate KPI severity level changes only marginally or not at all.

For example, if you have a CPU % utilization KPI that is running against three entities, and 2 of those entities show normal severity, while the third shows critical, the overall service health score might show critical, while the aggregate KPI severity level remains normal.
Impact of service dependencies on service health scores

Any service dependencies that you add to a service will impact the service health score, based on the importance value that you set for dependent service KPIs. For more information, see Set importance values for service dependencies in this manual.
KPI Configuration

Overview of configuring KPIs in ITSI

A KPI (Key Performance Indicator) is a recurring saved search that returns the value of an IT performance metric, such as CPU load percentage, memory used percentage, response time, and so on.

ITSI lets you create KPIs and add them to your services. You can then use KPI search result values inside ITSI to monitor service health, check the status of IT components, and troubleshoot trends that might indicate an issue with your IT systems.

For example, cpu_load_percent is a KPI that measures the CPU load percentage on a server. If your organization has a site uptime guarantee of 99.9% per month, you will need to monitor the status of this KPI (and others) to ensure that CPU performance remains within acceptable parameters.

When you initially create a KPI search, you configure the following set of search properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source search</td>
<td>A search string that you define as the basis for your KPI, using a data model, ad hoc search, metrics search, or base search.</td>
<td>Yes</td>
</tr>
<tr>
<td>Entity Split Field</td>
<td>A field in your data that can be used to break down the KPI. This option lets you apply a KPI search to multiple entities, enabling comparative analysis of search results on a per-entity basis. This field can be different from the Entity Filter Field.</td>
<td>Optional</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Required?</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td><strong>Entity Filter Field and Entity Alias Filtering</strong></td>
<td>Filter entities in or out of a KPI search using Entity Filter Field. Map entity aliases to fields in your search data to determine the specific entities to which a KPI search applies.</td>
<td>Optional</td>
</tr>
<tr>
<td><strong>Monitoring calculations</strong></td>
<td>The recurring KPI search schedule and statistical operations on search results, including service health score calculations.</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Backfill</strong></td>
<td>Fills the summary index (itsi_summary) with historical raw service health score data.</td>
<td>Optional</td>
</tr>
<tr>
<td><strong>Severity-level thresholds</strong></td>
<td>Thresholds that you apply to KPI search results. Severity-level thresholds let you monitor KPI status (normal, low, medium, high, and critical) and set trigger conditions for alerts.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

For example, to monitor the CPU load percentage of an entity (machine) in a service, you can create a KPI using an ad hoc base search that returns the value of the field `cpu_load_percent` at 5 minute intervals over a 5 minute time range, then set a range of severity-level thresholds between 0% and 100%.

For a step-by-step guide to adding a KPI to a service, see Add a KPI to a service in ITSI.

After you add a KPI to a service, you can perform other configuration tasks, like setting KPI importance values and applying anomaly detection.
See also

- Set KPI importance values in ITSI.
- Create KPI threshold time policies in ITSI.
- Apply anomaly detection to a KPI in ITSI.

Add a KPI to a service in ITSI

ITSI uses KPI searches to monitor the performance of your IT services. You must add at least one KPI search to a service to use ITSI. For information on how the number of KPIs can impact performance, see Performance considerations in this manual.

This topic walks you through the basic KPI search creation modal. These instructions assume that you have already created a service. If not, see Overview of creating services in ITSI in this manual. You must create a service with at least one KPI to run ITSI.

Step 1: Add a new KPI

1. Click Configure > Services from the ITSI top menu bar.
2. Select an existing service.
3. Click New in the KPI tab and choose one of the following options:
   - Select Generic KPI.
   - Select a KPI template. For example, Application Server: CPU and Memory > Memory Used. KPI templates provide pre-configured KPI source searches, including ad hoc searches and base searches, based on ITSI modules. KPI templates are tailored for specific service monitoring use cases, such as operating systems, databases, web servers, load balancers, virtual machines, and so on.
4. In Step 1 of the KPI creation modal, enter the KPI title and optional description. Click Next.

Step 2: Define a source search

When you create a KPI, you must define a source search on which to build the KPI. You can chose from four source search types: data model search, metrics search, ad hoc search, and base search.
**Note:** Before you define your source search, consider the performance implications for your particular deployment. While data models are suitable for smaller test environments, base searches generally provide best performance in larger production settings. See Create KPI base searches in ITSI.

**Define a source search from a data model**

1. Configure your data model search.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI source</td>
<td>Data Model</td>
</tr>
<tr>
<td>Data Model</td>
<td>The data model object, child, and attribute fields. For example, Host Operating System &gt; Memory &gt; mem_used_percent. When you create a KPI search from a data model, the data model object field becomes the threshold field. When you create a KPI search from an ad hoc search, you must manually enter the threshold field.</td>
</tr>
<tr>
<td>Filters (optional)</td>
<td>Click Add Filter to add data model filter conditions. Data model filters let you include/exclude search result data based on the filter conditions. For example, the filter condition host Equals ipaddress filters out all values for the data model search field host, except for values that equal ipaddress. Data model filtering can help improve the speed and accuracy of your searches by excluding extraneous data from search results.</td>
</tr>
</tbody>
</table>

2. Click **Generated Search** to preview your KPI search string. Use the Generated Search box to view changes that ITSI makes to your search string as you build your KPI. Click anywhere on the Generated Search itself to run the search.

3. Click **Next**.

**Define a source search from a metrics search**

1. Configure your metrics search.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
If there are no metrics indexes configured in your Splunk deployment, you will receive the message: "No metrics found."

For more information about metrics, see Get started with Metrics in the Splunk Enterprise Metrics Manual.

Metrics Search
- Select the metrics index from which to choose a metric.

Metric
- Select the metric to use for the KPI. For example, `memory.used`.

2. Click Generated Search to preview your KPI search string. Metrics searches begin with the `mstats` command.

3. Click Next.

**Define a source search from an ad hoc search**

1. Configure your ad hoc search.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI source</td>
<td>Ad hoc Search</td>
</tr>
<tr>
<td>Search</td>
<td>The ad hoc search string that you create. This is the event gathering search for the KPI.</td>
</tr>
<tr>
<td>Note:</td>
<td>The use of transforming commands, the <code>mstats</code> command, the <code>gettime</code> macro, or time modifiers in your KPI search is not recommended as this may cause issues with KPI backfill, the display of raw data on ITSI views such as glass tables and deep dives that allow you to run KPI searches against raw data, and the KPI threshold preview.</td>
</tr>
<tr>
<td>Threshold Field</td>
<td>The field in your data that the KPI aggregates and monitors. For pure counts use <code>_time</code>.</td>
</tr>
</tbody>
</table>

2. Click Generated Search to preview your KPI search string.

3. Click Next.

**Define a source search from a base search**

1. Configure your base search.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI source</td>
<td>Base Search</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Base Search</td>
<td>The base search that you want to associate with the KPI. For example, DA-ITSI-OS: Performance.Memory. Base searches provide pre-configured KPI templates built on ITSI modules.</td>
</tr>
<tr>
<td>Metric</td>
<td>The metric that you want to associate with the KPI. For example, <code>mem_free_percent</code>.</td>
</tr>
</tbody>
</table>

2. (Optional) click **Generated Search** to preview your KPI search string.
3. Click **Next**.

**Note:** Most fields in the next window (steps 3 through 6) are pre-populated for the base search by the KPI template. For more information on how to create and configure KPI base searches, see Create KPI base searches.

### Step 3: Filter entities

Filter entities to have more granular control of your KPI at the entity level.

**Split by Entity**

The Split by Entity option lets you maintain a breakdown of KPI values at the entity level. Use Split by Entity to enable monitoring of KPI values for each individual entity against which a KPI is running.

You must split KPIs by entity to use the following ITSI features:

- Per-entity thresholds. See Set per entity threshold values in this manual.
- Entity overlays. See Add overlays to a deep dive in ITSI in the *ITSI User Manual*.
- Maximum severity view in the Service Analyzer. See Aggregate versus maximum severity KPI values in ITSI in the *ITSI User Manual*.
- Cohesive anomaly detection. See Apply anomaly detection to a KPI in ITSI in this manual.

Configure the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Split by Entity</td>
<td>Enable/disable a breakdown of KPI values at the entity level. The KPI must be running against two or more entities.</td>
</tr>
<tr>
<td>Entity Split</td>
<td>Specify the field in your data to use to look up the corresponding split by entities. The default lookup field for data model searches and ad</td>
</tr>
</tbody>
</table>
Field searches is **host**. For metrics searches, select a dimension associated with the metric. This field is case sensitive.

When filtering a KPI down to entities, you can split by a field other than the field you are using for filtering the entities (specified in the Entity Filter Field). This allows you to filter to the hosts that affect your service, but split out your data by a different field. For example, you might want to filter down to all of your database hosts but split the metric by the processes running on the hosts.

### Entity filtering

Entity filtering lets you specify the entities against which a KPI search will run. Provide the entity filter field and apply entity alias filters to reduce collection of extraneous data.

Configure the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter to Entities in Service</td>
<td>Enable/disable entity filtering.</td>
</tr>
<tr>
<td>Entity Filter Field</td>
<td>Specify the field in your data to use to look up the corresponding entities by which to filter the KPI. For metrics searches, select a dimension for the metric. The default field for data model searches, ad hoc searches, and metrics searches is <strong>host</strong>. This field can be different than the field used for the Entity Split Field.</td>
</tr>
</tbody>
</table>
| Entity Alias Filtering | The entity alias that you want to use as a filter. This filters out all aliases from your search, except the specified alias. For example, a value of `host` filters out all entity alias values from the KPI search, except those that match the name `host`. For example, if your service has these two entity rules:

```plaintext
host=sr-centos2.sv.splunk.com
IP= 10.141.20.37
```

your KPI search initially includes an OR clause, such as:

```plaintext
| search Performance.dest=sr-centos2.sv.splunk.com OR dest=10.141.20.37
```
When you specify `host` as an entity alias filter, it limits the KPI search clause to:

```
| search Performance.dest=sr-centos.sv.splunk.com
```

The Entity Alias Filtering field will be removed in the next major version of ITSI. See Entity Alias Filtering field in the Removed features section of the Splunk IT Service Intelligence Release Notes for information on what you need to do to prepare.

For more information on entity aliases, see Define Entities in ITSI in this manual.

### Step 4: Add monitoring calculations

Configure the following KPI monitoring calculations:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **KPI Search Schedule**    | Determines the frequency of the KPI search.  
  
  Avoid scheduling searches at one minute intervals. Running multiple concurrent KPI searches at short intervals can produce lengthy search queues and is not necessary to monitor most KPIs. |
| **Service/Aggregate Calculation** | The statistical operation that ITSI performs on KPI search results. The correct aggregate calculation to use depends on the type of KPI search. For example, if your search returns results for CPU Load percentage, you could use `Average`. If your search returns a count, such as number of errors, then you would want to use `Count`. |
| **Calculation Window**     | The time period over which the calculation applies. For example, `Last 5 Minutes`.                                                                                                                         |
| **Fill Data Gaps with**    | How to treat gaps in your data. This affects how KPI data gaps are displayed in service analyzers, deep dive KPI lanes, glass table KPI widgets, and other dashboards in ITSI populated by the summary index.  
  
  - Select **Null values** to fill gaps in data with N/A values. Also select the severity level to use for Null values. |
Select **Last available value** to use the last reported value in the ITSI summary index. For aggregate level KPIs, service aggregate data gaps are filled with the last reported aggregate KPI value. For entity level KPIs, entity data gaps are filled with the corresponding entity’s last available value. After the entity gaps have been filled, the service aggregate result is calculated for the KPI.

- Select **Custom value** to specify a specific value to use when there is a gap in data. Enter a positive integer.

Filled gap values are not used in the calculations performed for Anomaly Detection and Adaptive Thresholding.

Click **Next**.

**How filling data gaps with last reported value works**

Each time the saved search runs for a KPI with the Fill Data Gaps with option set to **Last available value**, the alert value for the KPI is cached in a KV store collection called itsi_kpi_summary_cache. ITSI uses a lookup named itsi_kpi_alert_value_cache in the KPI saved search to fill entity-level and service-aggregate gaps for the KPI using the cached alert value.

To prevent bloating of the collection with entity/service-aggregate KPI results, a retention policy runs on the itsi_kpi_summary_cache collection using a Splunk modular input. The modular input runs every 15 minutes and removes the entries from cache that have not been updated for more than 30 minutes. 15 minutes is the default frequency and 30 minutes is the default retention time for entries in cache. You can change the frequency and retention time in the [itsi_age_kpi_alert_value_cache://age_kpi_alert_value_cache] stanza of the SA-ITOA/local/inputs.conf file.

The filling of data gaps with the last reported value occurs for at most 45 minutes, in accordance with the modular input interval and retention time (15 minutes + 30 minutes by default). If data gaps for a KPI continue to occur for more than 30 to 45 minutes, the KPI will stop getting filled with the last reported value and data gaps will start displaying as N/A values.
Step 5: Unit and Monitoring Lag

Configure the following optional settings:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit</td>
<td>The unit of measurement that you want to appear in KPI visualizations. For example, GB, Mbps, secs, and so on.</td>
</tr>
<tr>
<td>Monitoring lag</td>
<td>The monitoring lag time (in seconds) to offset the indexing lag. When indexing large quantities of data, an indexing lag can occur, which can cause performance issues.</td>
</tr>
</tbody>
</table>

Click Next.

Step 6: Enable backfill

Enable backfill to fill the summary index (itsi_summary) with historical raw KPI data. Backfill runs a search in the background that populates the summary index with historical KPI data for a given time range (backfill period) as it would have been populated at a regularly scheduled time by KPI saved searches. In other words, even though the summary index only started collecting data at the start of this week when the KPI was created, if necessary you can use the backfill option to fill the summary index with data from the past month.

Backfill is a one-time operation. Once started, it cannot be redone or undone. For example, if you backfill 60 days of data and then later decide that you want 120 days, you cannot go back and change the backfill period. Think carefully about how many days of data you want to backfill before saving the service.

The backfill option requires you to have indexed adequate raw data for the backfill period you select.

When you enable backfill, you must indicate how many days of data to backfill. You can choose a predefined time range like last 7 days, or select a custom date prior to the current date. If you choose a specific date, the dropdown dynamically updates with the number of days you’re backfilling to.

The backfill period is the time range of data that is available after backfill is complete. For example, if you select last 60 days, ITSI fills the summary index with data from the past 60 days. In other words, you now have 60 days of KPI data available.

You must save the service to initiate the backfill. A message appears in Splunk.
Web that informs you when the backfill is complete.

If you backfill a KPI that uses **Last available value** to fill data gaps, data gaps are backfilled with filled-in alert values (using the last reported value for the KPI) instead of N/A alert values. If you backfill a KPI that uses a **Custom value** to fill data gaps, data gaps are backfilled with filled-in alert values (using the custom value provided) instead of N/A alert values. See Step 4: Add monitoring calculations.

**Step 7: Set thresholds**

Severity-level thresholds determine the current status of your KPI. When KPI values meet threshold conditions, the KPI status changes, for example, from high (yellow) to critical (red). The current status of the KPI is reflected in all views across the product, including service analyzers, glass tables, and deep dives.

You can manually add threshold values for your KPIs one at a time using the threshold preview window. Or apply threshold time policies, which automatically adapt threshold values based on day and time. See Create KPI threshold time policies in ITSI.

ITSI supports two types of KPI severity-level thresholds: Aggregate thresholds and per-entity thresholds. Adaptive thresholds can be used with aggregate thresholds but not per-entity thresholds.

**Set aggregate threshold values**

Aggregate thresholds are useful for monitoring the status of aggregated KPI values. For example, you might apply aggregate thresholds to monitor the status of KPIs that return the total number of service requests or service errors, based on a calculation that uses the stats count function.

1. Click **Aggregate Thresholds**.
2. Click **Add threshold** to add a range of severity-level thresholds to the threshold preview graph.
3. Click Finish.

**Set per-entity threshold values**

Per-entity thresholds are useful for monitoring multiple separate entities against which a single KPI is running. For example, you might have a KPI, such as Free Memory %, that is running against three separate servers. Using per-entity thresholds, you can monitor the status of Free Memory % on each individual server.

Adaptive thresholding cannot be used on a per-entity basis.

**Prerequisites**

To use per-entity thresholds, a KPI must be split by entity. See "Step 3: Filter entities" above.

**Steps**

1. Click **Per Entity Thresholds**.
2. Click **Add threshold** to add a range of severity-level thresholds to the threshold preview graph. Optionally, if you want to use the same values as the aggregate thresholds, click **Apply values** to copy those threshold values over.

The threshold preview shows a separate search results graph for each entity that the KPI is running against.
1. Click **Finish**.

### Set KPI importance values in ITSI

After you create your KPI, you must assign the KPI an importance value. ITSI uses the importance value, along with the KPI severity level, to calculate a service health score.

Importance values range from 0 to 11. KPI importance values from 1-11 are included in the health score calculation, with 1 being the least important and 11 being the most important. KPIs with an importance value of 0 are not included in the health score calculation. The greater the KPI importance value, the greater the impact that KPI has on the service health score.

ITSI considers KPIs that have an importance value of 11 as a special case that represents a "minimum health indicator" for the service. When a KPI with an importance value of 11 reaches the critical state, the overall health score for the service turns critical, regardless of the status of other KPIs in the service.

### Steps

1. Open the service that contains your KPI.
2. Click the **Settings** tab. The Health Score Calculation section shows the importance value of each KPI, along with the simulated health score of the service.
3. Use the slider to set the importance value for your new KPI.
4. Click **Save**.

Note the impact that different severity-levels have on the simulated health score given the KPI importance value. Use this feature to help fine-tune your KPI importance values. The simulated health score is for preview purposes only and has no impact on actual severity-level thresholds or service health scores.

## Create KPI threshold time policies in ITSI

ITSI lets you create KPI threshold time policies with threshold values that change over time. You can use KPI threshold time policies to accommodate normal variations in usage across your services and improve the accuracy of KPI and service health scores. For example, if your organization’s peak activity is during the standard work week, you might create a KPI threshold time policy that takes into account higher levels of usage during work hours, and lower levels of usage during off-hours and weekends.

You can only have one active time policy at any given time. When you create a new time policy, the previous time policy is overwritten and cannot be recovered.

## Use KPI threshold templates

ITSI provides a set of 32 default thresholding templates that you can use to build your time policies. You can select templates with different time block combinations, such as work hours, off hours, weekends; AM/PM; 3 hour block; 2 hour block: and so on.

Thresholding templates are static or adaptive. Static templates let you create time policies that do not change after you configure them. Adaptive templates let you create time polices that generate thresholds dynamically and update daily based on changes in your data. Adaptive thresholds can be used with aggregate thresholds but not per-entity thresholds.

## Steps
1. In the KPIs list, select the specific KPI for which you want to set a threshold time policy.
2. Expand the Thresholding panel.
3. Select Use Thresholding Template. Then select a thresholding template, such as "3-hour blocks every day (adaptive/stdev)." Selecting an adaptive template automatically enables Adaptive Thresholding and Time Policies. The Preview Aggregate Thresholds window opens. ITSI backfills the preview with aggregate data from the past 7 days. If there is data in the summary index it loads the summary data, otherwise it loads raw data. Loading raw data can take some time.
4. Expand the Configure Thresholds for Time Policies panel.
5. (Optional) Select a time policy block, then click Apply Adaptive Thresholding. Make sure to use the correct policy type. Policy types for adaptive thresholds include Standard deviation, Quantile, and Range.

ITSI generates your new threshold time policy, which you can view in the Preview Aggregate Thresholds window. Adaptive thresholds update once daily at midnight.
6. Click Save.

Time policies cannot overlap. If you attempt to create a policy that overlaps with another policy, a validation error appears.

Create custom threshold time policies

You can create custom threshold time policies tailored to your specific monitoring needs and variations in the usage of your services. You can also enable adaptive thresholding for custom time policies, and set a training window over which historical KPI data is analyzed for adaptive threshold adjustments.

Time policies support time zone offsets of 15, 30, and 45 minutes for compatibility with non-hourly time zones (such as (GMT+05:30) Chennai, Kolkata, Mumbai, New Delhi). This allows users in non-hourly time zones to accurately apply time policies created by users in hourly time zones.
Steps

1. In the KPIs list, select the KPI for which you want to set a threshold time policy.
2. Expand the **Thresholding** panel.
3. Select **Set Custom Thresholds**.
4. For Enable Time Policies, select **Yes**.
5. (Optional) Click **Yes** to enable adaptive thresholding. Set the **Training Window**, which is the time window over which historical KPI data is analyzed for adaptive threshold updates.
6. Expand the **Configure Thresholds for Time Policies** panel.
7. Click **Add Time Policy**.
8. Configure your custom time policy:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>The name of the threshold time policy.</td>
</tr>
<tr>
<td>Start Time (HH:MM)</td>
<td>The specific hour and minute at which the threshold time policy begins. Note that the threshold start time supports 15, 30, and 45 minute offsets for compatibility with off-hour time zones.</td>
</tr>
<tr>
<td>Duration</td>
<td>The number of hours to which the threshold applies.</td>
</tr>
<tr>
<td>Repeat</td>
<td>The specific days of the week to which the time policy applies.</td>
</tr>
<tr>
<td>Apply the threshold values from an existing time policy?</td>
<td>(Optional). Choose whether to copy the threshold values from an existing policy over to this policy. This option saves you time and effort if your policies have identical or similar thresholds.</td>
</tr>
</tbody>
</table>

9. Click **Add**.
10. (Optional) To apply adaptive thresholding to this specific time policy, select the policy and click **Apply Adaptive Thresholding**.

Policy types that support adaptive thresholds include Standard deviation, Quantile, and Range.

**Copy threshold values between policies**

If you’re creating multiple time policies that require the same threshold values, you can save time by copying the threshold levels and their corresponding values from one policy to another. For example, if you configure **Policy 1** with the threshold values Critical=94, High=88, and Medium=75, and you want to use those same levels and values in **Policy 2** and **Policy 3**, you can copy them over
rather than inputting the values manually.

1. Click the More Options icon on a policy and choose **Copy threshold values**.

![Aggregate Threshold Values](image)

2. Select the policies to apply the current threshold values to. You can apply aggregate or per-entity thresholds to aggregate and/or per-entity thresholds of the same policy and/or other policies.

3. Click **Save**. ITSI copies the threshold levels, values, and base severity from the original policy over to the policies you selected.

**Create a new KPI threshold template**

Create a custom KPI threshold template with specific time block combinations that are tailored to your business case.

**Prerequisites**

To create a new KPI threshold template, you must have write permissions for the Global team. You must also possess the appropriate capabilities to be able to read, create, or delete KPI threshold templates.

**Steps**

1. In the ITSI main menu, click **Configure > KPI Threshold Templates**.
2. Click **Create Threshold Template**.

If your role does not have write access to the Global team, you will not see the **Create Threshold Template** option.

3. (Optional) Enter a title and description.
4. KPI threshold templates can only exist in the Global team. The Team cannot be changed.
5. Click **Create**.
6. Select the template you just created in the KPI Threshold Templates lister page.
7. Select a **Preview Service** and **Preview KPI**. For example, Database Service and CPU Utilization %, respectively.
8. (Optional) For **Enable Adaptive Thresholding**, click **Yes** to enable time varying thresholds that update periodically based on historical KPI data.
   ♦ For **Training Window** select the time period over which to evaluate KPI data for adaptive thresholding.
9. Select **Aggregate Thresholds** for the preview to display the single aggregate search value. Or, select **Per Entity Thresholds** for the preview to display KPI values for each individual entity against which the KPI runs.
10. Expand the **Configure Thresholds Time Policies** panel.
11. Define a time policy for this template. Click **Save**.

You can now apply your custom thresholding template to any number of KPIs.

**Edit thresholding templates**

ITSI provides an editing page where you can modify any existing thresholding templates. For example, you might want to change the days and hours for a specific time policy, change the policy type, or apply adaptive thresholding.

**Steps**

1. Select a KPI in the service definition.
2. Expand the **Thresholding** panel.
3. Select **Use Thresholding Template** and select the template you want to edit.
4. Click **Edit Template**.
5. Make your modifications to the thresholding templates.
6. Click **Save**.

**Apply anomaly detection to a KPI in ITSI**

ITSI provides anomaly detection algorithms that learn KPI patterns continuously in real time, and detect when KPIs depart from their own historical behavior. You can use anomaly detection to identify trends and outliers in KPI search results that might indicate an issue with your system.

Anomaly detection in ITSI 2.5.0 or later requires Java 7 or Java 8. If you are using Java 7, any version of ITSI running on Splunk Enterprise version 6.6.0 or later also requires Java Cryptography Engine (JCE). See Install required Java components in this manual.
Anomaly detection algorithms

ITSI provides two anomaly detection algorithms: trending and entity cohesion.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trending</td>
<td>The trending algorithm detects anomalous patterns in a single time series (or metric). A sliding window on the time series is monitored by a scoring function (based on non-parametric statistics), which continuously generates scores to reflect the anomalousness of patterns in the current window, compared to patterns in historical data. Thresholds are computed adaptively without any distribution assumptions and are robust to outliers. Anomalously high scores generate an alert. Trending anomaly detection applies to aggregate KPI events, and is useful for tracking anomalous KPI behavior on the service level.</td>
</tr>
<tr>
<td>Entity cohesion</td>
<td>The entity cohesion (cohesive) algorithm detects anomalous patterns in multiple time series simultaneously. The group of time series the algorithm monitors is assumed to have similar or &quot;cohesive&quot; behavior and patterns. A scoring function continuously monitors all time series within a sliding window and generates scores for each time series to reflect its departures in pattern from the rest of the time series. Significant departures of a time series from its cohesive peers are indicated by high anomaly scores, which trigger alerts using similar thresholding techniques as the trending algorithm. Entity cohesion anomaly detection applies to KPIs that are shared across multiple entities (4 minimum), and is useful for tracking anomalous behavior on the entity level. To use entity cohesion a KPI must be split by entity.</td>
</tr>
</tbody>
</table>

Analyze KPI data and enable algorithms

Before you enable anomaly detection algorithms, it is a best practice to use the Analyze KPI Data tool to confirm that your KPI is recommended. Use this tool to determine if the algorithms will produce meaningful results for the KPI based on specific criteria. KPIs that do not meet the criteria for the algorithm are likely to generate false positives and are not recommended.
You can enable anomaly detection algorithms for any KPI irrespective of the analysis results. The only scenario in which you cannot enable the algorithm is in the case of entity cohesion, if the KPI is not split by entity.

1. Click **Configure > Services**.
2. Open the service containing the KPI for which you want to apply anomaly detection.
3. Select the KPI.
4. Expand the **Search and Calculate** panel.
5. In the Unit row, click **Edit**
6. Select **Enable backfill** and define the backfill period over which you want to analyze KPI data.
7. Click **Finish**.
8. Expand the **Anomaly Detection** panel.
9. In **Analysis Time Window**, select the time range for KPI data analysis.
10. Click **Analyze KPI Data**.
    - If Algorithm Analysis Result shows **Recommended**, then the KPI meets the criteria for the algorithm.
    - If Algorithm Analysis Result shows a warning message, then the KPI does not meet the criteria for use with the algorithm. Mouse over the tooltip to learn more about the algorithm requirements.
    See Algorithm analysis criteria below.

11. For each recommended algorithm, click **Yes** to enable it.
12. Adjust the **Algorithm Sensitivity** slider to set the algorithm’s sensitivity to variance in data. The more sensitive the algorithm is, the more likely it is to generate an anomalous event. The algorithm now evaluates the KPI data continuously and generates anomalous events based on the algorithm sensitivity threshold.
13. Click **Save**.

**Algorithm analysis criteria**

The table shows the specific criteria a KPI must meet to be recommended for use with the respective algorithm.

<table>
<thead>
<tr>
<th>KPI Criteria</th>
<th>Trending algorithm</th>
<th>Entity Cohesion algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min amount of data</td>
<td>24 hrs.</td>
<td>24 hrs.</td>
</tr>
<tr>
<td>KPI Criteria</td>
<td>Trending algorithm</td>
<td>Entity Cohesion algorithm</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>% of anomalous data points</td>
<td>&lt; 10%</td>
<td>&lt; 10%</td>
</tr>
<tr>
<td>Min number of entities</td>
<td>N/A</td>
<td>4 entities min</td>
</tr>
<tr>
<td>Max number of entities</td>
<td>N/A</td>
<td>30 entities per KPI</td>
</tr>
</tbody>
</table>

**Triage and investigate anomalous events**

An anomalous event is an event that is inconsistent with or deviating from what is usual. ITSI generates a notable event in Episode Review when it detects an anomalous event. You can then open the event in a deep dive to perform root cause analysis.

- The trending algorithm generates notable events with the heading "Service level alert on KPI."
- The entity cohesion (cohesive) algorithm generates notable events with the heading "Entity level alert on KPI."

The type of algorithm that generated the notable event appears in the Details section of the Overview tab.

For more information, see Overview of Episode Review in ITSI in the Splunk ITSI User Manual.

**Open anomalous events in a deep dive**

You can drill down to a deep dive from any anomalous notable event. This lets you view the event over a default 10 minute time range, and perform root cause analysis in the context of other service KPIs.

1. Select the anomalous notable event in Notable Events Review.
2. Under Drilldowns, click the Drilldown to `<service_name>` link. A deep dive opens with an overlay of the anomaly in the KPI lane.
3. (Optional) Add additional KPIs to the deep dive for contextual analysis of the anomalous event.

For more information, see Add anomaly overlays in the Splunk ITSI User Manual.

**Set max entity limit**

The entity cohesion anomaly detection algorithm supports a maximum of 30 entities per KPI. If you run KPI analysis against a KPI that has more than the 30 entities, a warning message appears stating that the KPI has too many entities,
and the KPI is not recommended.

If you want to lower the maximum number of entities at which KPI analysis triggers a warning message, lower the value of `metrics_maximum` in the `[cohesive]` stanza of `$SPLUNK_HOME/etc/apps/SA-MetricAD/local/mad.conf`. For example:

```
metrics_maximum = 15
```

### Change memory configuration in SA_ITSI_MetricAD

The default memory configuration for anomaly detection is set to 1GB, which supports up to 1000 KPIs for trending analysis or up to 1000 metrics for cohesive analysis. To support more than 1000 KPIs for trending analysis or 1000 metrics for cohesive analysis, increase the Heap memory size in `SA-ITSI-MetricAD/local/command.conf` on your search heads. To determine the amount of memory you should use, size your analysis requirement first, then calculate your memory requirements based on that.

Trending anomaly detection requires about 600MB per 1000 KPIs. Cohesive anomaly detection requires about 1GB per 1000 metrics (for example, a combination of 10 KPIs with 100 entities or 20 KPIs with 50 entities).

Note that these recommendations should be used as a guideline only as memory usage for anomaly detection can be influenced by many factors such as the size of historical data, algorithm configurations, and available CPU.

On each search head:

1. Go to `$SPLUNK_HOME/etc/apps/SA-ITSI-MetricAD/local/`
2. Edit `commands.conf`
3. In the `[MAD]` stanza, change `command.arg.1=-J-Xmx1G` to increase the memory.
4. Restart Splunk Enterprise.

### Create KPI base searches in ITSI

KPI base searches let you share a search definition across multiple KPIs. You can create base searches to consolidate multiple similar KPIs, reduce search load, and improve search performance. For example, if you have similar ad hoc searches whose only difference is an entity or threshold field, you can consolidate these searches into a single base search definition and achieve
better search performance.

**ITSI module base searches**

ITSI includes several pre-configured KPI base searches based on ITSI modules that you can use with your services. The titles of these base searches begin with "DA-ITSI". KPI base searches that come with ITSI modules are read-only and cannot be modified or deleted. To customize a base search that comes from an ITSI module, clone the base search, then perform your edits on the clone.

**Service templates and base searches**

Service templates use base searches for their KPIs. When a service template is created from a service, all of the KPIs in the service are imported into the template. Any service KPIs that use ad hoc searches, data model searches, or metrics searches are converted into base searches. These base searches are listed on the KPI Base Searches lister page and are available to use for KPIs in any service, just like any other base search. Base searches that are created for service template KPIs use the following naming standard: `<service name>:<KPI name>_<last 8 digits of KPI ID>`.  

**Create a new base search**

You can create new KPI base searches, then use those base searches to build KPIs in the configure services workflow. You can create a base search using an ad hoc search or a metrics search. See Overview of metrics in the Splunk Enterprise *Metrics* manual for information about Splunk metrics.

**Prerequisites**

All KPI base searches exist in the Global team. Only a user with write permissions to the Global team can create a KPI base search. The user must also possess the appropriate capabilities to be able to read, create, or delete base searches.

**Steps**

1. Click **Configure > KPI Base Searches**.
2. Click **Create KPI Base Search**.
3. Enter a title for your base search. For example, **CPU load base search**.
4. (Optional) Enter a description for your base search. For example, This base search can be used to build KPIs for CPU metrics.
You cannot change the Team field because all KPI base searches exist in the Global team. All users have read access to the Global team.

5. Click **Create**.

6. Configure your base search:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Search</strong></td>
<td>Select either <strong>Ad hoc Search</strong> or <strong>Metrics Search</strong>.</td>
</tr>
<tr>
<td></td>
<td>♦ For an Ad hoc Search, provide the source ad hoc search. For example, index=os source=ps.</td>
</tr>
<tr>
<td></td>
<td>♦ For a Metrics Search, select the metrics index to use and the metric. If you do not have a metrics index, you will see the message: &quot;No metrics found&quot;.</td>
</tr>
<tr>
<td><strong>KPI Search Schedule</strong></td>
<td>The frequency of the search (Every 1, 5, or 15 minutes).</td>
</tr>
<tr>
<td><strong>Calculation Window</strong></td>
<td>The time period over which the search applies. (Last 1 min, 5 min, 15 min, or 24 Hours).</td>
</tr>
<tr>
<td><strong>Monitoring Lag</strong></td>
<td>Pushes the search back by the specified number of seconds. This value must match the index lag time.</td>
</tr>
<tr>
<td><strong>Split by Entity</strong></td>
<td>Select <strong>Yes</strong> to maintain a breakdown of KPI values on the entity level.</td>
</tr>
<tr>
<td><strong>Entity Split Field</strong></td>
<td>The field in your search results that is used to look up the corresponding entity to split the KPI. For example, you might want to split your KPI by the processes running on the hosts instead of just the host.</td>
</tr>
<tr>
<td><strong>Filter to Entities in Service</strong></td>
<td>Select <strong>Yes</strong> to filter the search based on the entity alias.</td>
</tr>
<tr>
<td></td>
<td>To filter to entities in a service, the service must have associated entities.</td>
</tr>
<tr>
<td><strong>Entity Filter Field</strong></td>
<td>The field in your search results that is used to look up the corresponding entity to filter the KPI. For example, you might want to filter down to all of your database hosts.</td>
</tr>
<tr>
<td><strong>Entity Alias Filtering</strong></td>
<td>The entity alias that you want to use as a filter. This filters out all aliases from your search, except the specified alias.</td>
</tr>
</tbody>
</table>

The Entity Alias Filtering field will be removed in the next major version of ITSI. See **Entity Alias Filtering field** in the Removed features in Splunk IT Service Intelligence section of the **Release Notes** for information on what you
need to do to prepare.

7. Click **Add Metric**. The Add Metric modal appears. You can add multiple metrics to your base search. Each metric defines a threshold field (for ad hoc searches only) and the calculation method used to aggregate KPI search results on the entity and service level.

8. Configure your metric.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>The name of the metric. For example, CPU load percent</td>
</tr>
<tr>
<td>Threshold Field (Ad hoc search type only)</td>
<td>This is the field in your data that the KPI aggregates and monitors. For pure counts, use _time.</td>
</tr>
<tr>
<td>Unit</td>
<td>The type of measurement that the KPI calculates. For example, %, MB, and so on.</td>
</tr>
<tr>
<td>Entity Calculation</td>
<td>Sets the calculation method for calculating aggregate search results on the entity level if Split by Entity is set to Yes.</td>
</tr>
<tr>
<td>Service/Aggregate Calculation</td>
<td>Sets the calculation method for calculating the service/aggregate level.</td>
</tr>
<tr>
<td>Fill Data Gaps with</td>
<td>Select how you would like to treat gaps in your data. This affects how KPI data gaps are displayed in service analyzers, deep dive KPI lanes, glass table KPI widgets, and other dashboards in ITSI populated by the summary index.</td>
</tr>
<tr>
<td></td>
<td>♦ Select <strong>Null values</strong> to fill gaps in data with N/A values. Also select the severity level to use for Null values in the Threshold level for Null values dropdown.</td>
</tr>
<tr>
<td></td>
<td>♦ Select <strong>Last available value</strong> to use the last reported value in the ITSI summary index. For aggregate level KPIs, service aggregate data gaps are filled with the last reported aggregate KPI value. For entity level KPIs, entity data gaps are filled with the corresponding entity’s last available value. After the entity gaps have been filled, the service aggregate result is calculated for the KPI. See How filling data gaps with last</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>reported value works for more information.</td>
</tr>
<tr>
<td></td>
<td>♦ Select <strong>Custom value</strong> to specify a specific value to use when there is a gap in data. Enter a positive integer.</td>
</tr>
</tbody>
</table>

9. Click **Add**.
   This adds the metric to the list of metrics defined for your base search. When you build a KPI from a base search, you can select one and only one metric for the KPI.

10. Click **Save**.
    You can now use the base search to build KPIs in the configure services workflow.

If you delete a base search, any service KPIs that use the base search are converted to ad hoc searches. You cannot delete a base search that is being used by a KPI in a service template. You must select a different base search for any service template KPIs that use it before you can delete it. Additionally, you cannot delete a metric that is being used by a base search in a service template.

**Build new KPIs from a base search**

You can use KPI base searches to build new KPIs. Each KPI that you build is linked to the base search. If you edit and save a base search, those changes are propagated to all linked KPIs.

KPI base searches contain metric specifications. The metric specification for ad hoc base searches includes a threshold field. The metric specification for both ad hoc base searches and metrics base searches contains a method of calculation for aggregate search results at the service and entity level. When you apply a base search to a new KPI, you must select a metric specification from the base search to complete the new KPI search definition.

For example, if you want to create a new KPI to measure CPU load, you might select a metric specification from the base search that contains `cpu_load_percent` as the threshold field and `average` as the calculation method.

To build a KPI from a KPI base search:

1. Select **Configure > Services** and select a service.
2. In the KPIs tab, select **New > Generic KPI**.
3. Enter a title and description. Click **Next**.
4. For **KPI Source** select **Base Search**.
5. In the **Base Search** dropdown, select the base search for your new KPI. You can choose from base search templates provided by ITSI modules, or from your own custom base searches.

6. Select a metric from the **Metric** menu. Click Next. The Entities page appears. All fields are populated from the selected base search. Click Next. The Calculation page appears. All fields are populated from the selected base search. Click Next. The Optional Setup page appears. All fields are populated from the selected base search, with the exception of **Enable Backfill**.

7. Select the **Enable Backfill** check box, then select a **Backfill Period** (optional). Click Next.

8. Set appropriate severity-level thresholds for the KPI. Click Finish. The new KPI is created and appears in the list of KPIs for the service.

To unlink a KPI from the base search, edit the KPI and change the search type to adhoc, then save the KPI. This lets you use KPI base searches as a starting point for new KPIs.

**KPI base search performance considerations**

The performance of KPI base searches (the amount of time it takes to run the search) is dependent on the following factors:

- The number of KPIs that use the base search.
- The number of services that contain KPIs that use the base search.
- The number of entities matching service entity rules.

Most of the KPI base searches delivered with ITSI are configured to run every minute. Based on testing on a system with 32 cores and 16 GB of memory, a single KPI base search can support up to 5,000 KPIs with 15 entities matched by service entity rules reasonably well.

In general, a KPI base search can support fewer KPIs with many entities or many KPIs with fewer entities. It is not advised to use a single KPI base search for both a high number of KPIs and a high number of entities. As the number of services or matching entities goes up, the search runtime also increases.

You can check the runtime for your KPI base searches on the **Activity > Jobs** page. The runtime is the actual time it takes to run the search. Check the KPI search schedule (or frequency) of the KPI base search: every minute, every 5 minutes, or every 15 minutes. If a KPI base search is scheduled to run every minute, and the runtime of that search is longer than 1 minute, the search is...
taking too long to run. To reduce the search runtime, you need to reduce the number of KPIs using the KPI base search, the number of services that have the KPIs, or the number of entities for each service accordingly. The easiest solution is to clone the KPI base search and use the cloned base search for some of the KPIs.

**Increase write search result limit**

Search results are processed, created, and written to the itsi_summary index via an alert action. The default limit on the number of rows that can be written is 50,000 as specified in the $SPLUNK_HOME/etc/system/default/limits.conf file. You can increase this limit if necessary.

Calculate the number of the result rows generated by a shared base search using the following formula:

\<\text{number of services} \times \text{number of KPIs in each service} \times \text{number of entities per service entity rule}\> + \text{number of services} \times 2

For example, for 500 services with 10 KPIs in each service and 15 matching entities, the expected number of result rows is:

\[500 \times 10 \times 15 + 500 \times 2 = 76,000\] rows

If the number of result rows expected is more than 50,000, the results will be truncated. As a result, ITSI will display incorrect KPI values.

If you believe you are running into this limitation, create a new limits.conf file in the $SPLUNK_HOME/etc/apps/SA-ITOA/local directory and add the following stanza and setting:

```
[scheduler]
max_action_results = 1000000
```

Set the value for max_action_results to a number higher than 50,000. In the example above it is set to 100,000.

**Increase the KV store bulk get limit**

The KPI base search tries to get all the relevant services from the KV store internally for thresholding related operations. When a KPI base search is attached to a lot of services, the bulk get might reach the KV Store bulk get size
limit (the default limit is set to 500MB).

As a guideline, for one service with 20 fully populated KPIs in which all KPIs have custom thresholds with time policies configured, as well as cohesive anomaly detection configured, the size is roughly 0.8 MB in the KV store.

If you have a large number of services containing a lot of KPIs and metadata, it is recommended to increase the KV store bulk get limit in

$SPLUNK_HOME/etc/apps/SA-ITOA/local/limits.conf. Increase the max_size_per_result_mb value as necessary.

[kvstore]
# The maximum size, in megabytes (MB), of the result that will be returned for a single query to a collection.
# ITSI requires approximately 50MB per 1,000 KPIs. Override this value if necessary.
# Default: 500 MB
max_size_per_result_mb = 500

Schedule KPI searches to update at the same time in ITSI

By default, ITSI staggers the search scheduling of KPIs in order to reduce search load. For example, if you have five KPIs that are scheduled to run every 5 minutes, the search to update the value of each KPI from the summary index is staggered over the 5 minute interval (the first KPI at minute 1, the second KPI at minute 2, and so on).

You can synchronize KPI searches so they update at the same time during the scheduled interval. For example, if 5 KPIs are scheduled to run every 15 minutes, they ALL run at the 15/30/45/00 mark instead of being staggered over the interval.

Steps

1. Create a local version of itsi_settings.conf in
   $SPLUNK_HOME/etc/apps/SA-ITOA/local/
2. Add the following stanza:

   [synced_kpi_scheduling]
disabled = 0
When disabled = 0, newly created KPI saved searches run at the same time during each scheduled interval. After a KPI is updated, its search schedule will be overwritten to follow the synchronized schedule.

To reset the search schedules of all existing KPIs to use the new synchronized search schedule after you set disabled = 0, restart Splunk software and then use mode 4 of the kvstore_to_json.py script. See Regenerate KPI search schedules (mode 4) for details.

This setting affects all KPIs, including base searches, for all services. Enabling synced scheduling can have a significant performance impact because it increases the scheduler load. The increased load can result in a delay in search execution due to the number of searches being dispatched at the same time. You might need to scale up your hardware in order to support the increased load.
Service Templates

Manage services in bulk with service templates in ITSI

Manage similar services by creating a service template and linking the services to the service template. After services are linked, any changes you make to the service template are propagated to the linked services. For example, if you add a KPI to a service template, it is added to all the linked services. KPIs and entity rules that are unique to a service can still be configured directly in the service.

Use service templates to do the following:

- Create a single service and link it to a service template to populate the KPIs and entity rules. See Create a single service in ITSI.
- Bulk import many services at once and link them to a service template. See Bulk import services in ITSI.

You can also use the ITSI REST API to link services to service templates. It is best practice to create the service on which to base the template in the UI rather than creating the service through the ITSI REST API. See itoa_interface/service/<_key>/base_service_template in the ITSI REST API Manual for information.

When you update a template, the following changes are propagated to the linked services:

- New KPIs
- Deleted KPIs

For the following properties, you can control how the updates are propagated to linked services:

<table>
<thead>
<tr>
<th>Property</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity rules</td>
<td>Choose whether to overwrite the rules in the linked services or not.</td>
</tr>
<tr>
<td>KPI threshold templates</td>
<td>Choose to update the threshold templates for all KPIs, no KPIs, or only unchanged KPIs in the linked services.</td>
</tr>
<tr>
<td>Health score</td>
<td></td>
</tr>
</tbody>
</table>
Choose whether to overwrite the KPI importance values in the linked services with the importance values in the template. KPIs that exist in the service but not the template and dependent KPI values are not affected by overwriting.

### Property Options

<table>
<thead>
<tr>
<th>Property</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Choose whether to overwrite the KPI importance values in the linked services with the importance values in the template. KPIs that exist in the service but not the template and dependent KPI values are not affected by overwriting.</td>
</tr>
</tbody>
</table>

Service templates differ from services in the following ways:

- Templates do not contain a service health score.
- Template KPIs can only use a base search (not ad hoc or data model searches).
- Templates do not have service dependencies. These must be defined in individual services.
- Templates allow entity values to be defined in the linked services rather than in the template. These are known as configurable entity rules.
- Anomaly detection cannot be enabled for KPIs in a template; it must be enabled for KPIs in the individual services.

Templates exist in the Global team. You must have write access to the Global team and the write capability for service templates in order to create service templates. By default, only the itoa_admin possesses this capability.

Team admins (users with a role that inherits from the itoa_team admin role) can create services for their teams using service templates. The same template can be used for services in different teams. Dependent services do not have to be linked to the same service template as the parent service.

The higher the number of services linked to a service template, the higher the number of KPIs that use the same base search. For example, if a service template has 10 KPIs and is linked to 12 services, each KPI base search in the template is being used 12 times, once for each of the linked services. If the same KPI base search is used by too many services, this can lead to performance issues. See KPI base search performance considerations for information.

### Service template use cases

The following use cases illustrate how using service templates can streamline the process of creating and maintaining services.
Create one or more services from a service template

In this example, as the ITSI administrator, you need to create multiple services to monitor databases. Each service needs the same set of KPIs: Database Service Errors, Database Service Response Time, Storage Free Space: %, and CPU Utilization: %. However, each service will be monitoring different database servers (entities). One service needs to be filtered to the servers running Oracle database and another service needs to be filtered to the servers running MySQL.

1. A database service already exists that has the KPIs you want your services to have. You use this service to create a Database service template by selecting Edit > Create Service Template from the Services viewer page.

2. You name the template "Database service template" and click Create. The service template receives the entity rules, KPIs, KPI thresholds, and KPI importance values from the service.

3. The Database service didn't have any entity filtering rules, so you decide to add some entity rules to the template. First you add a rule that filters the template down to all entities that have "Category matches database." This helps you narrow down the entities that have a database installed, but you need a way to differentiate between the different services. To do this, you add a configurable entity rule. This allows you to select a key to filter on whose value will be defined in each service linked to the template. You add the following entity rule to the Database template: "host matches a value defined in the service".

4. Save the template and select to overwrite the entity rules so that the entity rules you added in the template will be added to the Database service that you used to create the template. You push the changes immediately and see the new entity rules in the Database service.

5. Now you are ready to create the new services based on the service template. You create a new service named "Oracle database service" and select the option to Link service to a service template and select the Database service template you created. Then you create a "MySQL database service" using the Database service template. These two services are now linked to the template and have received the entity rules and KPIs from the service template.

6. You open the Entity tab of the service definition for the Oracle Database service. You see an empty value for the host entity rule that the service received from the template. Specify the value for the host entity rule as oracle* since all the Oracle hosts begin with "oracle", for example: oracle-01, oracle-02, oracle-03.

In the service definition of the MySQL service, specify a value of mysql*
since all the MySQL hosts begin with "mysql," for example: mysql-01, mysql-02, mysql-03. Now the KPIs that are filtered to entities (such as Database Service Errors, Storage Free Space: %, and CPU Utilization: %) will be filtered to the appropriate hosts for each service.

7. You decide to go back to the template to make sure that KPI thresholds are set up properly. You select an adaptive threshold template for a KPI and apply it. You look over the preview and toggle it between each linked service to make sure it makes sense for each of the services. You know that if one of the services requires a custom adaptive threshold settings, you can make that change in the service and choose to only overwrite KPI thresholds for unchanged KPIs when changes are pushed from the template in the future.

8. You also decide to adjust the KPI importance value for one of the KPIs.

9. When you save the template, you choose to overwrite KPI thresholds and overwrite health score so that the KPI thresholds and KPI importance values in the services will be overwritten with the new ones you just added to the template.

10. So as not to cause disruption to the Splunk environment, you schedule the changes to be pushed to the linked services at 12:00am when the system should not be as busy.

11. The next morning, you see that the updates have been propagated to the database services.

See Also

- Create a single service in ITSI
- Create a service template in ITSI

Customize a service that is linked to a service template

The team admin who administers the Oracle database service in ITSI uses a custom plug-in with their Oracle database. She wants to add a KPI that checks the status of the plug-in. Because this KPI does not apply to every service linked to the Database service template, you are hesitant to add it to the Database service template. You tell the team admin to add the KPI to the service.

1. The team admin opens the service definition of the Oracle Database service and adds the KPI to monitor the plug-in and saves it.
2. The next time the ITSI admin updates the Database service template, this unique KPI for the Oracle Database service is not modified.

See Also
Configure a service created from a service template in ITSI
• Summary of changes that unlink a service from its service template

**Bulk import services and link them to a service template**

The director of IT has asked you to create Apache services for every Apache instance in the company. There are dozens of instances and you realize you cannot do this manually. You decide to use the ITSI bulk import feature. An Apache service template already exists with a configurable entity rule to filter entities by a different value for each linked service.

1. You query the CMDB and pull down a list of every Apache service in the company. You export the list as a CSV file and open it in Excel.
2. You add a new column that denotes the template each service will map to. In it you write "Apache Service Template" and copy it for every row.
3. You open up the ITSI bulk import and upload the CSV. You map the proper columns that denote the service name, service dependencies, and service template name.
4. After examining the bulk import preview, you decide the import looks good and move on to the entity configuration step. All of the services have a column called "serviceoid." This makes it very easy to map the entities for each service. You set the configurable entity rule for the Apache template to map to the values in the serviceoid column.
5. You click **Import** and all of your services are created and linked to the Apache service template.

**See Also**

• Bulk import services in ITSI

**Create a service template in ITSI**

To create a service template, you must first create a service on which to base the service template. The entity rules, KPIs, KPI importance values, and KPI thresholds of the service are imported into the service template. The service you create the template from is linked to the template by default. After the template is created, you can link multiple services to it. A service can be linked to only one template.

To create a service template, you must possess the capability to create a service template and have write permissions to the Global team. Only a user with the itoa_admin role can create a service template by default.
Users with team admin roles (roles that inherit from the itoa_team_admin role) have a read-only view of the Service Templates viewer page. Team admins cannot create or change service templates but they can create services for their teams using service templates.

Note that service templates are always created in the Global team, regardless of the team of the service that you use to generate the template.

**Prerequisite**

You must first create at least one service to generate a template.

**Steps**

1. Click **Configure > Service Templates** from the top menu bar.
2. Click **New Service Template**.
3. Select the service on which to base the template. The entity rules, KPIs, KPI importance values, and KPI thresholds of the service will be imported into the template. Additionally, the service you use to create the template will be linked to the template.

   Services with a lock icon are already linked to a template. If you choose a service that is already linked to a template, it is unlinked it from its current template and linked to the new template you are creating.

4. Type a name and description for the template and click **Create**.
   The entity rules and KPIs from the service are imported into the template, then the service template definition page opens.

5. Click the **Entities** tab to view the entity rules that have been imported from the service.
   Note that matching entities are not displayed in the service template. You can view matching entities at the bottom of the Entities tab in the service definition.

6. Click the **KPIs** tab to see the KPIs imported from the service. Any KPIs in the service that were based on an ad hoc search, metrics search, or data model are converted to a base search in both the service template and the service.

7. Click the **Settings** tab to see the importance values for the KPIs that were imported from the service.

8. In the **Linked Services** tab, note that the service you used to create the template is listed as a linked service and that the sync status is Synced.

**See also:**
• To make changes to the service template, see Update a service template in ITSI.
• To create a new service from a service template, see Create a single service in ITSI.

Update a service template in ITSI

Make updates to a service template to update all the services linked to the template. You can choose to push updates to the linked services immediately or schedule a specific date and time to push the updates.

1. Click Configure > Service Templates.
2. Open a service template.
3. On the Entities tab, add, modify, or delete entity rules as desired.
   Use the matches a value to be defined in the service or does not match a value to be defined in the service options to specify the value for an entity rule in each service rather than in the template. This is useful if each service linked to the template will be monitoring different entities. For example, you could specify alias vm_name matches a value defined in the service. Then, after the services have been synced with the template, specify the value for vm_name in each service. These options are particularly useful if you plan to bulk import services and link them to the template.

If you use the configurable entity rules described above and choose to Overwrite Entity Rules when you save the service template, any entity rules already defined in the linked services will be replaced with the entity rules from the template and they will have blank values (matching all entities).
4. On the KPI tab, add, modify, or delete the KPIs in the template, and set KPI thresholds as desired. Any KPIs you add to the template must use a base search. Template KPIs cannot use ad hoc, metrics, or data model searches.

   You cannot enable anomaly detection in a service template. You must enable anomaly detection for a KPI in the service directly.
5. On the Settings tab, set importance values for the KPIs and see how they affect the health score of linked services.
6. When you finish making modifications to the service template, click Save.
7. In the Save Service Template dialog box, indicate what information to update and select when to push the changes to the linked services.
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overwrite Entity Rules</td>
<td>Replaces the entity rules in the linked services with the entity rules in the template.</td>
</tr>
<tr>
<td>Overwrite KPI Thresholds on</td>
<td>Choose whether to update KPI thresholds. If you have customized some of the KPI thresholds in your services and you want to keep these thresholds, select <strong>All Unchanged KPIs</strong>. To overwrite customized KPI thresholds in linked services, select <strong>All KPIs</strong>. This setting does not affect KPI thresholds for any KPIs that are unique to the service and not linked to the template.</td>
</tr>
<tr>
<td>Overwrite Health Score Calculation</td>
<td>Replaces the KPI importance values that are used to calculate the health score in the linked services with the KPI importance values from the template. KPI importance values for any additional KPIs in the service that are not present in the template are not affected. KPI importance values for dependent services are also not affected.</td>
</tr>
<tr>
<td>Push changes to services</td>
<td>Choose to push the changes immediately, or select <strong>Later</strong> to choose a date and time to update the linked services. It is usually recommended to push changes to services outside of regular business hours to avoid disruption.</td>
</tr>
</tbody>
</table>

8. Click **Save**. The changes in the template are immediately propagated to the linked services if you chose to push the changes now or at the date and time you selected.

9. Click the **Linked Services** tab to see the sync status of the services linked to the template.

After the linked services are synced with the template, the status says **Synced**. If you elected to push the changes now, you might need to refresh your browser to see the status change. You will also receive a message in Splunk Web with the status of the sync.

You can only schedule one sync for a service template. If you make additional updates to a service template that is already scheduled, you can keep the scheduled time (the default) or choose to sync now or at a different time. All updates that have been made to the template are pushed at the time you choose.
Link a service to a service template in ITSI

You can link multiple services to a service template in order to manage them collectively. A service can be linked to only one service template.

When linking a service to a service template, any existing KPIs in the service are preserved and KPIs in the template are added to the service. You can choose how to update the entity rules in a service: append the template entity rules to the service entity rules, replace the service entity rules with the template entity rules, or keep the entity rules in the service and not import any of the entity rules from the service template.

There is no limit to the number of services that can be linked to a service template. However, the higher the number of services linked to a service template, the longer it will take to sync service template updates with the linked services.

This procedure describes how to link a single service to a service template. To link multiple services to a service template, use the bulk import feature. See Bulk import services in ITSI.

To create a new service from a service template, see Create a single service in ITSI.

Prerequisite

At least one service template must be created.

Steps

1. Click **Configure > Services** from the top menu bar.
2. In the Actions column for the service you want to link, click **Edit > Link to Service Template**
3. Click **Select...** and choose a template. If the service is already linked to a template, choose a different template to link the service to.
4. In the template details section, review the entity rules, the KPIs that are contained in the template, and the services already linked to the template.

When linking a service to a template for the first time, any KPIs that already exist are retained and new KPIs from the template are added. If any KPIs in the service have the same name as a KPI in the template, the KPI in the service is renamed using a unique identifier.
If the service was already linked to a template and you are linking it to a different template, linked KPIs (those that came from the original template) are removed, unlinked KPIs (KPIs that were added directly to the service) are retained, and KPIs from the newly selected template are added.

5. In the **Settings** section, choose how to update the entity rules in the service:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Append template rules</td>
<td>Adds the entity rules from the template to the service and keeps the ones that were already there.</td>
</tr>
<tr>
<td>Replace with template rule</td>
<td>Deletes the existing entity rules in the service and replaces them with the entity rules from the service template.</td>
</tr>
<tr>
<td>Keep existing service rules</td>
<td>Keeps the existing entity rules in the service and does not add any entity rules from the template.</td>
</tr>
</tbody>
</table>

6. Click **Link**. The Service Template column indicates that the service is synced with the service template.

To review the content updates that the service received from the template, click the name of the service. See Review updates received from the template and customize the service for more information.

## Manage service templates in ITSI

Service templates let you update services in bulk. To manage service templates, ITSI administrators can perform these tasks:

- Review content updates received from the template
- Unlink a service from a service template
- Clone a service template
- Delete a service template

### Review updates received from a template and customize the service

After linking a service to a template, or changing the template a service is linked to, review the content updates the service received from the template. You can also customize the service to meet your needs. If the service template contains entity rules with values that need to be specified in the service, provide them in the service entity rules.
Review the following tabs within the linked service:

**KPIs**

- Make sure the KPIs from the template were added to the service. These KPIs display a lock icon ♦ to indicate that they are linked to the template.
- Set custom threshold settings and enable anomaly detection for linked KPIs as desired. If you change the threshold settings, or if the adaptive thresholding value is recalculated, a ♦ icon appears in the Thresholding section to indicate the KPI threshold settings differ from those in the service template, but the service is still linked. To preserve custom thresholds in the service, make sure to select **Overwrite KPI Thresholds on: All Unchanged KPIs** when saving the template and pushing updates to the services.

Making any other changes to a linked KPI, such as changing the name of the KPI, changing the base search it uses, or enabling backfill, will cause the service to be unlinked from the template.

- KPIs that already existed in the service are listed without a lock icon. Any changes you make to the service template will not affect them.
- Any existing KPIs in the service that had the same name as a KPI in the template are renamed with _custom and a unique identifier appended to the name of the KPI. For example, if both the service and the template have a KPI called CPU Utilization: %, the KPI in the service is renamed to CPU Utilization: %_custom_02f8cab3-e5d1-11e7-b55a-005056923ff0 and the CPU Utilization: % KPI from the template is added. You can delete the duplicate KPI if desired.
- Add any new KPIs to the service as desired. These KPIs will not be overwritten when the service template is updated.

**Entities**

- If you selected **Append template rules** when you linked the service to the template, any entity rules that existed in the service are retained and the entity rules from the template are added to a new rule block (with an implicit OR between the rule blocks).
- If you selected **Replace with template rules** when you linked the service to the template, any entity rules that were in the service are gone and the entity rules from the service template are added.
If you selected **Keep existing service rules** when you linked the service to the template, the existing entity rules in the service are listed and no new entity rules are added from the template.

Any entity rules with blank values that came from the template contain a □ icon. Provide a value in order to filter the service to a specific set of entities. These entity rule values are meant to be custom for each service. Some entity rules might have blank values intentionally. These do not display the icon.

Add or change entity rules as desired for the service. You can choose to preserve the entity rules in the service when you update the service template.

**Service Dependencies**

Specify any service dependencies. Service dependencies, or impacting services, have some sort of effect or influence on the primary service. Adding service dependencies can help you detect if one service is having a negative impact on another service, and can be useful in performing root cause analysis.

**Settings**

Enable or disable the service, change the team the service belongs to, and change KPI importance values. Custom KPI importance values for a service, which are used to determine the service health score, can be preserved when the service template is updated.

**Unlink a service from a service template**

There is no direct method to unlink a service from a template. Once a service is linked to a service template, it can only be unlinked from the template by doing one of the following:

- Deleting the service template
- Linking the service to a different service template
- Modifying the name of a linked KPI
- Modifying the base search of a linked KPI (or modifying any other settings in the Search and Calculate section, including enabling backfill)
- Deleting a linked KPI

Linked KPIs have a lock icon next to the name of the KPI in the KPI tab of the service definition.
You can link a service to a different service template from the Linked Services tab of the service template by selecting **Link to another Service Template**.

To enable backfill for a linked KPI in a service, you must delete the service, recreate it using the service template, and enable backfill for all KPIs in the Settings section of the Create Service dialog box.

**Clone or delete a service template**

Cloning a service template creates an exact copy of it. However, the cloned template is not linked to the services that are linked to the original service template. Deleting a service template unlinks it from all of its linked services. You can choose to leave them unlinked or link them to another service template.

1. From the IT Service Intelligence main menu, click **Configure > Service Templates**.
2. In the Actions column of the service, click **Edit > Clone/Delete**.

To delete templates in bulk, select all the templates you want to delete and click **Bulk Action > Delete**.
Predictive Analytics

Predict and prevent outages with ITSI Predictive Analytics

ITSI Predictive Analytics uses machine learning algorithms to predict the health score value of a selected service. The models use historical service health score and KPI data to approximate what a service's health might look like in 30 minutes.

ITSI Predictive Analytics allows Splunk users of all skill levels to use machine learning technology. ITSI provides visualization tools that guide you through the process of creating machine learning models without having to learn complex machine learning algorithms and technology. Once your models are ready, ITSI makes it easy to obtain and use service health score predictions.

Use Predictive Analytics if you want to identify and fix service outages before they happen. By receiving a warning that your service is likely to degrade in the next 30 minutes, you can take steps to resolve the problem before it affects other areas of your system.

Use Predictive Analytics if:

- You’ve had unplanned outages in the past.
- You want to prevent future outages.
- You want to understand and identify patterns in your service.
- You want to understand how future outages can impact your business.
Set up Predictive Analytics in ITSI

Before you can predict service health scores with ITSI Predictive Analytics, you need to configure the Splunk Machine Learning Toolkit application and optionally set up permissions.

Enable Predictive Analytics

To use Predictive Analytics, you need to install the Splunk Machine Learning Toolkit (MLTK) and share the machine learning macros with all apps so ITSI can access them. To minimize memory and disk consumption, configure the Machine Learning Toolkit configuration file. This will allow the MLTK to handle more memory and events.

1. Install the Python for Scientific Computing add-on version 1.3 or later for your operating system:
   - Mac
   - Linux 64-bit
   - Linux 32-bit
   - Windows 64-bit

2. Install the latest version of the Splunk MLTK. Follow the steps in Install the Splunk Machine Learning Toolkit in the MLTK User Guide.

3. Give the MLTK app Global permissions:
   1. In ITSI, click App: IT Service Intelligence > Manage Apps.
   2. In the filter bar, enter Splunk Machine Learning Toolkit.
   3. Click Permissions.
   4. Ensure that All apps is selected.
   5. Click Save.

4. (Optional) Configure the MLTK to handle more memory and events:
   1. Create a copy of mlspl.conf in $SPLUNK_HOME/etc/apps/Splunk_ML_Toolkit/local.
   2. Configure the following settings under the [default] stanza and in each algorithm-specific stanza:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Recommended value</th>
</tr>
</thead>
<tbody>
<tr>
<td>max_inputs</td>
<td>1000000</td>
</tr>
<tr>
<td>max_memory_usage_mb</td>
<td>2000</td>
</tr>
<tr>
<td>max_model_size_mb</td>
<td>30</td>
</tr>
<tr>
<td>max_fit_time</td>
<td>2400</td>
</tr>
</tbody>
</table>

3. Save the file and restart Splunk.
Permissions requirements

The following table summarizes the ITSI roles and corresponding Predictive Analytics capabilities:

<table>
<thead>
<tr>
<th>Role</th>
<th>Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>itoa_admin</td>
<td>• Train a model</td>
</tr>
<tr>
<td></td>
<td>• Test a model</td>
</tr>
<tr>
<td></td>
<td>• Add a model to a glass table</td>
</tr>
<tr>
<td></td>
<td>• Create alerts from a model</td>
</tr>
<tr>
<td></td>
<td>• Delete a model</td>
</tr>
<tr>
<td>itoa_team_admin</td>
<td>• Train a model (with write access to the service)</td>
</tr>
<tr>
<td></td>
<td>• Test a model</td>
</tr>
<tr>
<td></td>
<td>• Add a model to a glass table</td>
</tr>
<tr>
<td></td>
<td>• Create alerts from a model</td>
</tr>
<tr>
<td></td>
<td>• Delete a model</td>
</tr>
<tr>
<td>itoa_analyst</td>
<td>• Read models (with read access to the service)</td>
</tr>
<tr>
<td></td>
<td>• Test a model</td>
</tr>
<tr>
<td></td>
<td>• Add a model to a glass table</td>
</tr>
<tr>
<td></td>
<td>• Create alerts from a model</td>
</tr>
<tr>
<td>itoa_user</td>
<td>• Read models (with read access to the service)</td>
</tr>
<tr>
<td></td>
<td>• Test a model</td>
</tr>
<tr>
<td></td>
<td>• Create alerts from a model</td>
</tr>
</tbody>
</table>

For more information about ITSI role capabilities, see Configure users and roles in ITSI in this manual.

Access Predictive Analytics in ITSI

ITSI Predictive Analytics is accessible through the service configuration workflow. When you want to create a model, edit that service directly.

Prerequisites
You must have the itoa_admin or itoa_team_admin role to train and create a predictive model. After you create a model, open it in the Predictive Analytics dashboard to do root cause analysis.

Steps

1. In the ITSI menu bar, click Configure > Services.
2. Click the name of the service that you want to predict a health score for.
3. Click the Predictive Analytics tab.

Determine whether your service is a good fit for modeling in ITSI

ITSI provides two visualizations to help you decide whether your service is a good fit for predictive modeling. Experiment with different time periods and review the graphs to determine which time period most accurately represents your data. The key principle is to provide data that reflects what actually happens in your business scenario.

Service Health Score and KPIs over time

The Service Health Score and KPIs over time graph displays the values of service health scores and KPIs over a selected time period. Splunk machine learning algorithms are sensitive to outliers because those values affect statistical significance calculations. If you see unusual values or outliers in your data, confirm whether these data points are relevant and real. If the extreme data points are accurate, predictable, and reoccurring, then your service might be a good candidate for modeling.

The shape of this graph can also help you determine which algorithm to use. For example, if you have cyclical data, which consists of regular rises and falls, a more basic algorithm like linear regression is sufficient. If your data is random and complex, a more robust algorithm like random forest regressor is a better fit.

For example, the following graph has cyclical data:
Distribution of Service Health Score Values

The Distribution of Service Health Score Values graph is a histogram of service health score values over a selected time period. Each bar is a count of how many times your service hits that health score range. Use the graph to confirm that your data is relatively uniformly distributed (that is, the data is spread equally across the entire range). If the data is skewed, with most data points falling near one health score, Splunk machine learning algorithms could produce biased models.

Use the shape of this graph to determine where most of your service health score values fall and how they vary. For example, if most of your service health scores are in the 90-100 range but some are scattered in the lower ranges, this indicates that you've had several outages. This is a perfect use case for predictive modeling.
Train a predictive model in ITSI

The process of training a machine learning model involves providing a machine learning algorithm with training data to learn from. The term "model" refers to the model artifact that is created by the training process. The learning algorithm finds patterns in the training data and produces a machine learning model that captures these patterns.

To train a model, you need to decide whether the data for the selected service is a good fit for predictive modeling. For more information, see Determine whether your service is a good fit for modeling in ITSI. If you decide to create a model, select the training inputs and train the model. After you train the model, you can save it into the service and create an alert for potential service degradation.

If you have a lot of data in the summary index, the process of training a model can take up to 30 minutes to complete. During this training period, do not navigate away from the page.

The following diagram shows a high-level workflow for training a model:
Check if service has “good” data for training.

I want to train a model.

Train
To train a predictive model for a selected service, you must do the following:

1. Specify a time period.
2. Choose an algorithm type.
3. Choose a machine learning algorithm.
4. Split your data into training and test sets.

**Specify a time period**

The time period determines how far back ITSI looks when training a predictive model. The amount of data a predictive model needs depends on the complexity and shape of your data within the time period, and the complexity of the algorithm.

Consider the following guidelines when you choose a time period:

- Use at least 14 days of data. 30 days or more is recommended.
- In general, a model needs enough data to reasonably capture the relationships that exist among inputs (past KPI values and service health scores) and between inputs and outputs (future service health scores).
- To encapsulate as much variance in your data as possible, use as much data as you have available to train a model. It is best to operate under the assumption that the more data you provide, the better.

If you don't have enough data in the summary index to adequately train a model, consider backfilling the summary index with service health score data. For more information, see Backfill service health scores in ITSI.

To look more closely at the shape and complexity of your data, consult the following graphs:

- Service Health Score and KPIs Over Time
- Distribution of Service Health Score Values

**Choose an algorithm type**

There are two types of algorithms you can use to train predictive models in ITSI: regression and classification. Fundamentally, regression is about predicting a numerical service health score, and classification is about predicting a service health state.
Regression algorithms

Regression algorithms predict numeric values. For example, salary or age. In ITSI Predictive Analytics, regression algorithms predict an actual numeric health score value (for example, health score=89). Such models are useful for determining to what extent peripheral factors, such as historical KPI values, contribute to a particular metric result (service health score). After the regression model is computed, you can use these peripheral values to make a prediction on the metric result.

The following visualization illustrates a scatter plot of actual versus predicted results. The test data is plotted, and the yellow line indicates the best fit line for predicting server power consumption.

Classification algorithms

A classification algorithm predicts a category. For example, diabetic or not diabetic, rather than a discrete numeric value such as cholesterol levels. The algorithm learns the tendency for data to belong to one category or another based on related data. In ITSI Predictive Analytics, a classification algorithm uses the training data to predict a service health state (Normal, Medium, or Critical).

In short, classification classifies data and constructs a model based on the training set, and uses that model in classifying new data.

For example, the following classification table shows the actual state of a field versus the predicted state of the field. The yellow bar highlights an incorrect prediction.
Choose a machine learning algorithm

ITSI provides the following machine learning algorithms:

- Linear regression (Regression)
- Random forest regressor (Regression)
- Gradient boosting regressor (Regression)
- Logistic regression (Classification)

Linear regression

The linear regression algorithm creates a model for the relationship between multiple independent input variables (historical KPI and service health score values) and an output dependent variable (future service health score). The model remains linear in the sense that the output is a linear combination of the input variables. The algorithm learns by estimating the coefficient values in the test set using the data available in the training set.

<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Useful when the relationship to be modeled is not extremely complex and if you don’t have a lot of data.</td>
<td>Tends to underperform when there are multiple or non-linear decision boundaries.</td>
</tr>
<tr>
<td>Simple to understand, which can be valuable for business decisions.</td>
<td>Sometimes too simple to capture complex relationships between variables.</td>
</tr>
</tbody>
</table>

Random forest regressor

The random forest regressor algorithm operates by constructing a multitude of decision trees during training and outputting the mean prediction (regression) of the individual trees. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of
reducing the variance of the prediction.

<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good at learning complex, highly non-linear relationships.</td>
<td>Can be prone to major overfitting.</td>
</tr>
<tr>
<td>Usually achieves high performance.</td>
<td>Slower and requires more memory.</td>
</tr>
</tbody>
</table>

**Gradient boosting regressor**

The gradient boosting regressor algorithm involves three elements: A loss function to optimize, a weak learner to make predictions, and an additive model to add weak learners to minimize the loss function. Decision trees are used as the weak learner. Trees are added to the model one at a time in an effort to correct or improve the final output of the model.

<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usually achieves high performance.</td>
<td>A small change in the training data can create radical changes in the model.</td>
</tr>
<tr>
<td>Generally accurate even if the data is very complex or nonlinear in nature.</td>
<td>Can be prone to overfitting.</td>
</tr>
</tbody>
</table>

**Logistic regression**

The logistic regression algorithm is a classification algorithm used to assign observations to a discrete set of classes. For example, logistic regression predicts values like win or lose rather than the actual numeric score of a hockey game. In ITSI Predictive Analytics, the logistic regression algorithm predicts whether a service’s health score will be Normal, Medium, or Critical in the next 30 minutes. Unlike linear regression, which outputs continuous numeric values, logistic regression transforms its output to return a probability value, which can be mapped to two or more discrete classes.

<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performs well when the input data is relatively linear.</td>
<td>Tends to underperform when there are multiple or non-linear decision boundaries.</td>
</tr>
<tr>
<td>Avoids overfitting.</td>
<td>Sometimes too simple to capture complex relationships between variables.</td>
</tr>
</tbody>
</table>

Because classification divides your data into separate categories, your data must be highly variable for logistic regression to work. If your data is fairly stable and usually hovers around the same health score, logistic regression cannot divide it
into three distinct categories.

**Split your data into training and test sets**

Think of your data as a limited resource. You can spend some of this resource to train your model, and some of it to test your model. But you can’t reuse the same data for both. If you test your model on the same data you used to train it, your model could be overfit without your knowledge. It is best if a model is judged on its ability to predict new, unseen data. Therefore, ITSI lets you create separate training and testing subsets of your dataset.

A common machine learning strategy is to take all available data and split it into training and testing subsets, usually with a ratio of 70-80 percent for training and 20-30 percent for testing. ITSI uses the training subset to train and predict patterns in the data. The train partition has a single and essential role: it provides the raw material from which the predictive model is generated. ITSI uses the test data to evaluate the performance of the trained model.

For example, if the training data predicts a health score of 60, and the test data shows a score of 62, the model has very high performance. ITSI evaluates predictive performance by comparing predictions on the test data set with actual values using a variety of metrics. Usually, you use the model that performed best on the test subset (the "recommended" model) to make predictions on future health scores.

By default, ITSI selects a random 70 percent of the source data for training and uses the other 30 percent for testing. Use the slider in the training section to specify a custom split ratio.

Consider the following guidelines when splitting your data:

- Always split your data before training a model. This is the best way to get reliable estimates of a model's performance.
- After splitting your data, don’t change the "Test Period" until you choose your final model. The test period is meant for retraining purposes.
- When dealing with large datasets (for example, 1 million data points), a
larger split of 90:10 is a good choice.  
• With a smaller data set (for example, 10,000 data points), a normal split of 70:30 is sufficient.  
• In most cases, the more training data you have, the better the result.

Test a predictive model in ITSI

Always evaluate a model to determine if it will do a good job predicting future health scores. When you assess the quality of a predictive model, you determine what will happen if you use that model to make predictions. That is, whether the predictions will be close to the eventual outcomes.

ITSI provides several industry-standard metrics and insights to review the predictive accuracy of your model. Although ITSI provides a recommendation based on the best-performing algorithm, it is important to review the visualizations to decide if your model is performing well enough to match your business requirements.

Select a model to test under Test a Model. A checkmark indicates which regression model had the best overall performance. For more information about the difference between regression and classification algorithms, see Choose an algorithm type in this manual.

Use these questions to help you choose the best model:

• Which model had the best performance on the test set? ("recommended" model)  
• Does the model perform well across various performance metrics?

Do not change the Test Period unless you want to test on all of the available data. Leaving the data in separate train and test partitions, as configured in the training/test split, provides an honest assessment of the model's performance.

Test a regression model

Use the following metrics to test and evaluate a regression model:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R²</td>
<td>R² represents how well the model explains the variability of the result. 100 percent (a value of 1) means the model fits perfectly. The closer the value is to 1, the better the result.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RMSE</td>
<td>Root Mean Squared Error (RMSE) is a measure of the difference between values predicted by the model and the values observed. RMSE explains the variability of the result, which is essentially the standard deviation of the residual. This value gives you an idea of how close or far the model is from completely accurate predictions. These values only make sense within one dataset. Do not compare them across datasets.</td>
</tr>
<tr>
<td>Actual vs. Predicted Service Health Score</td>
<td>A sequential overlay that compares the actual service health score to what the model predicted the number will be 30 minutes into the future.</td>
</tr>
<tr>
<td>Residual Error Histogram</td>
<td>A histogram of the residual error of each service health score during the test period. The residual error is the difference between the observed service health score and the estimated score that the model predicted. A graph that is normally distributed with values closer to 0 is most accurate.</td>
</tr>
</tbody>
</table>

**Test a classification model**

Because classification divides your data into separate categories, your data must be highly variable for logistic regression to work. If your data is fairly stable and usually hovers around the same health score, logistic regression cannot divide it into three distinct categories.

Use the following metrics to test and evaluate a classification model:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>The percentage of time a model identifies only the relevant data points. For example, of all predicted Severe data points, Precision measures how many of them are actually Severe. Precision is a good metric to use when the cost of a false positive is high. A false positive means that a health score that is Normal was actually predicted to be Severe. In this case, a user might waste important time investigating a predicted outage that never occurs.</td>
</tr>
<tr>
<td>Recall</td>
<td>The percentage of time a model predicts the correct severity. Recall is a good metric to use when there is a high cost associated with a false negative value. For example, if a Severe health score is predicted as Normal, your model might...</td>
</tr>
</tbody>
</table>
Create an alert for potential service degradation in ITSI

The predictive models in ITSI can detect when a service's health is about to decline. Create an alert from a model which will generate an event in Episode Review notifying you when the service health score is predicted to reach a configured threshold.

Prerequisites

- To create an alert from a predictive model, a trained model must be saved in the service definition. For more information, see Train a predictive model in ITSI.
- Make sure you're viewing the Predictive Analytics tab from the service definition.

Steps
1. From the Predictive Analytics tab, select the model under **Test a Model**.
2. Click the bell icon ( ) in the Worst Case Service Health Score panel.
3. Configure the following fields in the Create Correlation Search box:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search Name</td>
<td>The name of the correlation search.</td>
</tr>
<tr>
<td>Create a notable event</td>
<td>The threshold values used by the search to trigger an alert. When the service's health score drops below or between the values or severities you specify, ITSI generates an event in Episode Review.</td>
</tr>
<tr>
<td>Notable Event Title</td>
<td>The title of the notable event generated in Episode Review.</td>
</tr>
<tr>
<td>Run every</td>
<td>How often the correlation search runs.</td>
</tr>
<tr>
<td>Severity</td>
<td>The initial severity of the notable event. This severity is not tied to the severity of the prediction itself. Rather, it represents the importance of the prediction. For example, the predicted health score might be 60, which is technically in the Medium range for ITSI. However, if this service’s health is very important to you, you might make the Severity of the notable event <strong>High</strong> so that it will be prioritized and investigated sooner.</td>
</tr>
</tbody>
</table>

4. Click **Create**.

If the service’s health score drops to the value or severity you configured, ITSI generates a notable event in Episode Review. Click the drilldown link in the event to open the Predictive Analytics dashboard and perform more granular root cause analysis.

### Manage alerts

Alerts are stored with other correlation searches in ITSI. To delete or disable a Predictive Analytics alert, click **Configure > Correlation Searches** from the top menu bar. You can edit the correlation search to change fields such as Time Range, Severity, and Description.

### See also:

- Create correlation searches in ITSI
Add a predictive model to a glass table in ITSI

You can add predictive models to glass tables as ad hoc search widgets. This lets you actively monitor service health and troubleshoot imminent issues before they impact your services.

Ad hoc searches are not optimized because each widget runs its own individual search. Thus, the loading time for your glass table visualization is proportional to the number of predictive model widgets you add.

Prerequisite

- To create a glass table widget from a predictive model, you must have a trained model that is saved into the service definition. For more information, see Train a predictive model in ITSI.
- Make sure you’re on the Predictive Analytics tab within the service.

Steps

1. On the Predictive Analytics tab, select the model under **Test a Model**.
2. Click the magnifying glass in the Predicted Average Case Service Health Score or Predicted Worst Case Service Health Score panel to open the predictions in the Search app.
   
   The search is the same whether you select average or worst case.
3. Copy the search string.
4. Open your glass table in Edit mode.
5. Drag **Ad hoc Search** from the Services panel onto the canvas.
6. In the Configurations panel, paste the search string into the **Search** box.
7. In the time range picker, select the end time for your ad hoc search.
8. Expand **Earliest Time** and select the start time for your ad hoc search relative to the end time in the time range picker. This determines the time range over which the ad hoc search applies.
9. In **Threshold Field**, type `next30m_avg_hs` to display the average prediction, or `next30m_min_hs` to display the worst-case prediction.
10. Enable Thresholds to display thresholds for the ad hoc search widget:
    1. For Thresholds, click **On**.
    2. Click **Edit** and configure the thresholds for the service health score prediction.
    3. When finished, click **Done**.
11. (Optional) Add a drilldown to the Predictive Analytics dashboard.
    1. For Drilldown, click **On**.
    2. Click **Default**, and select **Custom URL**.
3. In a separate ITSI window, navigate to **Dashboards > Predictive Analytics**.
4. Select the service and corresponding model to display the health score prediction.
5. Copy the URL and paste it into the URL field in the glass table.
6. Edit the URL from `auto_load=false` to `auto_load=true` to automatically load all of the dashboard panels.
12. Select a viz type for your ad hoc search widget. You cannot use the sparkline or trending value viz types because the prediction is a static value.
13. Click **Update** to update the widget to the new visualization and display your search results over the specified time range.
14. Click **Save**.

For more information, see Configure ad hoc search widgets in the *ITSI User Manual*.

**Retrain a predictive model in ITSI**

For a model to accurately predict health scores, the data it’s making predictions on must have a similar distribution as the data on which the model was trained. Because data distributions are expected to drift over time, deploying a model is not a one-time exercise, but a continuous process.

It is good practice to continuously monitor your incoming data (historical KPI and service health score values) and retrain a model on newer data if KPIs or entities are added, removed, or changed. For example, if you add an Oracle database to a service, it is best practice to retrain the service’s model because the new architecture will create new KPI relationships.

If monitoring your services for changes is too time consuming, a simpler strategy is to train the model periodically. For example, to capture changes to KPIs or service architecture, you might retrain a model every 10 days.

Retrain a service’s model in the following situations:

- You added a new KPI or entity to the service.
- You removed or changed a KPI or entity in the service.
- You restored your ITSI configuration (ITSI does not restore MLTK lookup files).
- You notice that the model’s performance is starting to degrade.
Before you retrain a model, test it on recent data to evaluate whether it needs to be retrained.

**Prerequisites**

- To retrain a model, the model must be saved in the service definition. For more information, see Train a predictive model in ITSI.
- Make sure you're viewing the Predictive Analytics tab from the service definition.

**Steps**

1. Test the model on recent data:
   1. From the Predictive Analytics tab of the selected service, navigate to the **Test a Model** section.
   2. Change the test period to a recent time range. Changing the time period retests the model on a recent set of data to determine if it needs to be retrained. You must test on at least 90 minutes of data.

   The appropriate test period varies based on your specific data. For example, if a new KPI was added yesterday, test the model on the last 24 hours. If an outage occurred last week, test it on the last seven days.

   3. Select the model in the Regression Models or Classification Models table to populate the model's metrics.
   4. Analyze the model's metrics. If the metric values have dropped to unacceptable levels for your business, consider retraining the model.

2. Retrain the model on new data:
   1. Select the same algorithm and algorithm type as you used to train the model. You can modify the time period and training/test split.
   2. Click **Train**. The existing model is replaced by the retrained model.
   3. Reevaluate the model's metrics to ensure that they are at acceptable levels. For information about evaluating models, see Test a predictive model in ITSI in this manual.
   4. Click **Save** to save the retrained model into the service definition.

The following diagram illustrates the workflow for retraining a model:
Delete a predictive model in ITSI

Every time you create a model, it is added to a CSV lookup in Splunk web. CSV lookups are file-based lookups represented by a CSV file. Access, manage, and delete predictive models through Lookups.

1. In ITSI, click **Settings > Lookups**.
2. Select **Lookup table files**.
3. Locate the model you want to delete and click **Delete** in the Actions column.
   Services are listed according to service ID. Delete all three models associated with each algorithm, with the following suffixes: `_avg`, `_ss`, and `_worst`.

When you delete a model within Lookups, its name still appears in the UI in the Test a Model section with the message "This model has been deleted. If it was deleted by accident, retrain in Service Configuration".

For information about managing model permissions, see Managing model permissions within Lookups.

Predictive Analytics performance considerations in ITSI
While training and testing models in ITSI, there are several performance considerations to take into account. This topic provides guidance for choosing services to model, selecting training inputs, and retraining models.

While it’s impossible to provide prescriptive advice for maximizing performance in every situation, the following observations and tips can help you tune and improve performance in your unique environment:

1. **Avoid creating models for services that contain more than approximately 20 KPIs and 50 entities.** The training time for a model depends on three factors: the number of KPIs in the service, the number of entities in the service, and the frequency of KPI searches. For a service with 20 total KPIs, 50 total entities, and 1-minute KPI searches, the average training time is 5 minutes. Scaling any of these three factors increases the training time.

2. **Do not create models for more than 75 total services.** Management of models is manual and can be difficult if you create too many.

3. **Configure the MLTK to handle more memory and events.** For more information, see Set up Predictive Analytics in ITSI.

4. **Use at least 14 days worth of data to train your model.** The training period is determined by the time period you specify before training a model. 30 days or more is recommended. In general, it is best to use as much data as you have available to train a model.

5. **Base your training/test split ratio on the size of your dataset.** Use a large split when dealing with large datasets (for example, 1 million data points). Use a smaller split when dealing with smaller datasets (for example, 10,000 data points). For more information, see Split your data into training and test sets.

6. **Do not change the Test Period until you select a final model for the service.** Leaving the data in separate train and test partitions, as configured in the training/test split, provides an honest assessment of the model’s performance.

7. **Retrain a service’s model if KPIs or entities are added, removed, or changed.** Changes in service architecture can cause changes in KPI behavior and service health score trends. For more information, see Retrain a predictive model in ITSI.

8. **Before retraining a model, test it on at least 90 minutes of data.** If you test on fewer than 90 minutes, the results could be inaccurate or incomplete.

9. **Do not use the training search as a scheduled search.** Predictive Analytics models do not require repetitive retraining, such as retraining every night. Retraining them too often can cause scale implications. Training is specifically designed to be an intermittent expense.
More performance help

If you experience performance issues, or want to receive feedback tailored to your setup, you have the following options:

- Post a request to the community on Splunk Answers.
- Contact Splunk Support.

ITSI Predictive Analytics use case

With ITSI Predictive Analytics, you can build and train predictive models and use them to create alerts. This example demonstrates how to use ITSI Predictive Analytics to build a machine learning model, and use the model to generate predictions that you can use to make business decisions.

Scenario

You are an IT operations admin for the Middleware team. The Middleware service has experienced several costly outages in the past. To predict and avoid future outages, you want to create a machine learning model of the Middleware service using ITSI’s Predictive Analytics capability.

Use case overview

This use case includes the following high-level tasks:

1. Determine if the service is a good fit for modeling
2. Train a model
3. Test the model
4. Create an alert from the model
5. Perform root cause analysis using the Predictive Analytics dashboard
6. Retrain the model on new data

Determine if the service is a good fit for modeling

Before you create a predictive model for the Middleware service, you want to make sure that the service is suitable for predictive modeling. You set the time period on the Predictive Analytics tab to Last 14 days so you can get an idea of how the Middleware service’s health scores have varied over the last two weeks.
You look at the graph of service health scores and KPIs over time and see that the data is relatively cyclical with no obvious outliers.

You consult the histogram of service health scores and see that the distribution is fairly uniform (the data is spread evenly across the range of health scores). With this even distribution, ITSI's machine learning models are less likely to produce biased results.

Because the Middleware service data contains no outliers and is evenly distributed, you determine that the Middleware service is a good fit for predictive modeling.

**Train a model**

Now that you've decided to model the Middleware service, it's time to train a predictive model. You go to the Middleware service's Predictive Analytics tab to configure the training inputs.

**Time period**

You want to provide the model with enough data to capture the relationships that exist among the Middleware service's past KPI values and service health scores. Your last outage occurred two months ago, and you want to provide that data to the model so that it can be trained on the difference between normal and abnormal data. You decide to use the Last 90 days of data.

**See also:**

- Specify a time period

**Algorithm type**

There are two types of algorithms you can use to train predictive models in ITSI: regression and classification. Regression algorithms predict a numerical service health score, and classification predicts a service health state.

Regression is best when there is a pattern for health scores in the past, and classification when there is no pattern but rather random drops. Because there is a cyclical pattern to the Middleware service's historical health scores, you decide to go with Regression. When monitoring the Middleware service, you prefer that the health score does not drop below 75.

**See also:**
Choose an algorithm type

Algorithm

Because you chose regression, you have three algorithms to choose from: linear regression, random forest regressor, and gradient boosting regressor. You already determined that the Middleware service data is fairly cyclical, with regular rises and falls in service health score values. Because the relationships to be modeled are relatively simple, you decide to go with **Linear Regression**.

See also:

- Choose a machine learning algorithm

Split for training/test

A common machine learning strategy is to take all available data and split it into training and testing subsets. The train partition provides the raw data from which the predictive model is generated. The test set is used to evaluate the performance of the trained model.

You know that a larger split works well for large datasets, and a more normal split (like 70/30) works best for small datasets. Your dataset of 90 days is fairly large, especially since the Middleware service contains 8 KPIs and 22 entities. You decide to increase the train/test split to **80/20**. This split trains the model on 80% of the data, and uses the remaining 20% to test and evaluate it.

See also:

- Split your data into training and test sets

After everything's configured, your training inputs look like this:

Review the model's predictive performance

It's always a good idea to evaluate a model to determine if it will do a good job predicting future health scores. You want to make sure that the predictions the model generates will be close to the eventual outcomes.
After you click **Train**, the model is created in the Test a Model section.

You select the **LinearRegression** model, the "Recommended" model, to generate the results. The recommended model is the one that had the best performance on the test set. If you don't have a data science or machine learning background and are confused about which model to choose, it is best to choose the recommended model.

ITSI provides the following industry-standard metrics and insights to summarize the predictive accuracy of your model:
$R^2 = 0.96$
You make the following observations about the test results:

- The $R^2$ of 0.96 is very high on a scale of 0 to 1.
- The RMSE of 7.24 is relatively small.
- On the graph of actual versus predicted health scores, most of the points fall close to or exactly on the line. This suggests a large number of correct predictions, which means the model has high performance.
- The residual error histogram has a normal (bell-shaped) distribution with most values close to 0.

Your observations indicate that the linear regression model performs well across various performance metrics. You decide to go with this model, and you click **Save** to save it into the service definition.

**See also:**

- Test a predictive model in ITSI

**Set a score threshold and create an alert**

Now that you've saved the model into the service, you want to create an alert so you can be notified if the health score of the Middleware service is predicted to reach a certain worst-case level in the future.

You click the bell icon (♦) in the Worst Case Service Health Score panel to open the correlation search modal. You want ITSI to generate a notable event in Episode Review the next time the Middleware service's health is expected to degrade.

You configure the following inputs:
Because you want the Middleware service’s health score to stay between 75 and 100, you decide to generate an alert when the predicted health score is predicted to be 75 or less.

You configure the initial severity of the notable event to be Medium because a health score of 75 doesn’t necessarily indicate a severe outage. You might create another correlation search that generates a Critical event if the health score drops below 30, as this is a more critical situation.

See also:

- Create an alert for potential service degradation in ITSI

Use the model to generate predictions and find root cause

Two days later, one of your IT operations analysts is monitoring Episode Review and sees the following notable event:
The event details indicate that the Middleware service is predicted to degrade to an unacceptable level within the next 30 minutes.

The analyst clicks the **Predictive Analytics Middleware Dashboard** drilldown link. This high-level dashboard lets analysts perform root cause analysis by viewing the top five KPIs contributing to a potential outage and how those KPIs are likely to change in the next 30 minutes.

The analyst sees the following information in the dashboard:

**Predicted Service Health Score in 30 Minutes**

63.48
Along with the predicted health score in 30 minutes, ITSI calculates the top 5 KPIs that are likely contributing to the degraded health score. The analyst determines that the Middleware service is likely to degrade because of an increased amount of 5xx errors, which is the top offending KPI. The analyst opens a ticket with the Middleware team to address the 5xx error count before it becomes a problem.

See also:

- Predictive Analytics dashboard

Retrain the model on new data

A month after creating the initial model for the Middleware service, you make a change to one of the KPIs in the service. It is good practice to continuously monitor your incoming data (historical KPI and service health score values) and retrain a model on newer data if KPIs or entities are added, removed, or changed.

You decide to test the existing model on recent data to evaluate whether it needs to be retrained.

On the Middleware service's Predictive Analytics tab, you navigate to the Test a Model section where the current model is saved. You change the Test Period to Last 90 days and select the LinearRegression model. This tests the existing model on the last 90 days of data.

The best metrics to review when retesting a model are $R^2$ and RMSE because they allow you to make a direct numerical comparison to the original values.

You see the following results:
## Test a Model

**Test Period**

Last 90 days

*All data within the selected time period will be used for testing.*

### Regression Models

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Recommended</th>
</tr>
</thead>
<tbody>
<tr>
<td>LinearRegression</td>
<td>✔</td>
</tr>
</tbody>
</table>

### R²

0.41
The test results indicate that $R^2$, a measure of how well the model fits, has decreased from 0.96 to 0.41. RMSE, which should be relatively small, has increased from 7.24 to 28.70. These metrics are unacceptably low, so you decide to retrain the model.

You select the same training inputs as before and click **Train**. Training with the same algorithm replaces the existing model with the retrained model.

After the model is trained, you consult the **Test a Model** section to review the new model's predictive performance. The test results display the following metrics:
The $R^2$ value has increased significantly and RMSE has dropped significantly. The values are close to what they were when you originally trained the model. The model's performance has increased after you retrained it on recent data, accounting for the recent change to the Middleware service's KPI. You are satisfied with the new model's performance and click **Save** to save it into the service definition.

**See also:**

- Retrain a predictive model in ITSI
Notable Events

Customize Episode Review in ITSI

As a Splunk ITSI administrator, you can customize the way that analysts view and interact with events and episodes in Episode Review.

Modify which events analysts can see

By default, ITSI service-level permissions apply to episodes in Episode Review. This means that analysts viewing Episode Review can only see events from services for which they have read permission. If an event is not associated with a particular service (none of the fields in the event contains service information) then all users can view the event.

The ITSI administrator can choose to disable service-level permissions for Episode Review using the itsi_team.conf file. To disable service-level permissions for Episode Review, create a new itsi_team.conf file in the SA-ITOA/local directory and set disabled = 1 under the [notable_event_review_security_group] stanza.

If service-level permissions are disabled for Episode Review, all ITSI users can see all notable events, regardless of which service they are associated with. However, service information for services that a user does not have read access to will not be displayed for notable events.

For information about service-level permissions, see ITSI Service-level permissions in the Installation and Configuration Manual.

Modify analyst permissions

Configure read/write permissions on a saved view of Episode Review to restrict permissions for certain roles. By default, read and write permissions are granted to "Everyone" (all roles) for a newly created Episode Review.

Prerequisites

• You must have the itoa_admin or itoa_team_admin role, or be assigned the configure_perms capability, to set permissions on a saved Episode Review. For more information, see Configure users and roles in ITSI.
Steps

1. Within Episode Review, click the side arrow to show alternate views.

![Episode Review side arrow](image)

2. Click **Full Lister Page**.
3. On the Episode Review lister page, click **Edit > Permissions** on the saved view you want to edit.
4. Allow or prevent analysts from reading or writing to the saved Episode Review. Everyone is granted read/write access by default.
5. Click **Save**.

Change Episode Review columns

You can change the columns displayed in a saved Episode Review.

1. Click the gear icon to open the View Settings modal.
2. Use the **Columns Shown** section to edit, remove, or change the order of the available columns.
3. Add custom columns by selecting **Add Column**.
4. Click **Done**.

Manage notable event indexes in ITSI

Splunk IT Service Intelligence implements custom indexes for notable event storage. In a single instance deployment, the installation of ITSI creates the indexes in `$SPLUNK_HOME/var/lib/splunk`.

<table>
<thead>
<tr>
<th>Index</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>itsi_tracked_alerts</td>
<td>Stores active raw notable event data.</td>
</tr>
<tr>
<td>itsi_notable_audit</td>
<td>Stores all audit events for episodes, including actions, comments, status change, and owner change.</td>
</tr>
<tr>
<td>itsi_grouped_alerts</td>
<td>Stores active episode data.</td>
</tr>
<tr>
<td>itsi_notable_archive</td>
<td>Stores episode metadata (tags and comments) that has been moved from the KV store after a default 6 month retention period, which begins when you close an episode in the UI. Moving data from the KV store removes</td>
</tr>
</tbody>
</table>
extraneous data and helps improve performance.

<table>
<thead>
<tr>
<th>Index</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>extraneous data and helps improve performance.</td>
</tr>
</tbody>
</table>

## Set the notable event retention policy

In addition to the indexes listed above, ITSI stores the status of notable events and episodes in the KV store collections called `itsi_notable_<object type>`. By default, notable event metadata is archived after 6 months to keep the KV store from growing too large. If you have a large number of events, use the ITSI Health Check dashboard to check the collection sizes on disk and decide if you need to change the retention policy.

You can tune the retention policy for notable event metadata using an ITSI configuration file. The retention policy determines how long notable event metadata remains in the KV store before it is moved to `itsi_notable_archive`. Retention policies are based on the `mod_time` (modify time), not the tag or comment creation time.

### Steps

1. Create a local version of `itsi_notable_event_retention.conf` in
   
   `$SPLUNK_HOME/etc/apps/SA-ITOA/local/`

2. In the `[default]` stanza, and in individual stanzas as needed, set `retentionTimeInSec` to a new value.

```plaintext
[default]
# default retention policy is 6 months
retentionTimeInSec = 15768000
disabled = 1

[itsi_notable_event_tag]
# Check retention policy against mod_time (modify time of tags) and move
tags to index once they met condition
disabled = 0
object_type = notable_event_tag

[itsi_notable_event_comment]
# Check retention policy against mod_time (modify time of comment) and
move comment to index once they met condition
disabled = 0
object_type = notable_event_comment

[itsi_notable_event_ticketing]
# Check retention policy against mod_time (modify time of ticket) and
move ticket to index once they met condition
disabled = 0
```
Use notable event real-time search optimization

ITSI uses an indexed real-time search to retrieve notable events from the Splunk platform. Indexed real-time searches have a delay of about 90 seconds before events get processed. If you require faster processing times, you can change the ITSI search to use a real-time search instead of an indexed real-time search.

Changing from indexed real-time search to real-time search significantly increases the requirements of running the system and will impact the speed of indexing.

Change the search to real-time by modifying savedsearches.conf:

1. Create a local version of savedsearches.conf in
   $SPLUNK_HOME/etc/apps/SA-ITOA/local/
2. In the [itsi_event_grouping] stanza, set dispatch.indexedRealtime to 0.

[itsi_event_grouping]
dispatch.indexedRealtime = 0
Backfill notable events into episodes

ITSI uses the `itsi_event_grouping` search command to aggregate notable events into episodes. If this search is disabled, ITSI will not group notable events into episodes. This can occur during a restart of Splunk Enterprise, or if a user manually disables the search.

When the `itsi_event_grouping` search is re-enabled, ITSI looks through notable events that were missed while the search was disabled, and backfills them accordingly into episodes.

The default look back time for missed events is 24 hours. If you want to change the look back time, modify the `group_restore_lookback_time` field in the `itsi_rules_engine.properties` file, as follows:

1. Edit
   `$SPLUNK_HOME/etc/apps/SA-ITOA/local/itsi_rules_engine.properties`.
2. Add the following parameter to the file:
   
   `group_restore_lookback_time=<number of hours>.

   For example, to look back 2 days, add the following:

   `group_restore_lookback_time=48`

Clear all notable events

To permanently delete indexed notable events, use the CLI `clean` command. This command completely deletes the data in one or all indexes or KV store collections, depending on whether you provide an `<index_name>` or `<collection>` argument. For more information, see How to use the clean command in the Managing Indexers and Clusters of Indexers manual.

The `clean` command does not work on indexer clusters unless you run the clean command separately on each indexer.

1. In the CLI, to stop Splunk Enterprise type `$SPLUNK_HOME/bin/splunk stop`
2. On each indexer, run the following commands to clear the indexes:

   `$SPLUNK_HOME/bin/splunk clean eventdata -index itsi_tracked_alerts;`
   `$SPLUNK_HOME/bin/splunk clean eventdata -index itsi_notable_audit;`
   `$SPLUNK_HOME/bin/splunk clean eventdata -index itsi_notable_archive;`
On a single search head, run the following commands to clear the KV store collections:

```bash
$SPLUNK_HOME/bin/splunk clean eventdata -index itsi_grouped_alerts
```

3. To start Splunk Enterprise type

```bash
$SPLUNK_HOME/bin/splunk start
```

4. To start Splunk Enterprise type

4. Tune notable event grouping in ITSI

Notable event aggregation polices group notable events to organize them in Episode Review. ITSI provides a file called `itsi_rules_engine.properties`, located at `$SPLUNK_HOME/etc/apps/SA-ITOA/default/`, where you can tune and customize notable event grouping settings.

Create a local version of `itsi_rules_engine.properties` and increase the following settings as necessary to improve notable event grouping on your deployment.

```plaintext
# The period, in seconds, at which to fetch aggregation policies from the KV store.
policy_fetch_period = 45

# The number of sub-groups that can be created for an aggregation policy with
# split by fields. If you exceed this limit, you will break all
# sub-groups
# that exist for an aggregation policy.
sub_group_limit = 10000

# The group index name.
index_name = itsi_grouped_alerts

# The HTTP token name.
token_name = itsi_grouped_alerts_token
```
# The HTTP sync token name.
# NOTE: If the sync token name and the HTTP token name are the same, a
token
# with async functionality is created.
sync_token_name = itsi_group_alerts_sync_token

# The timeout value for receiving an acknowledgement from HEC.
# When processing a notable event and the action criteria is met, this
# setting
# ensures that the current event is indexed before executing an action.
http_ack_time_out = 10

# The default source.
default_source = itsi_group_alerts

default_sourcetype = itsi_notable:group

# The number of events that can be contained in the parent group, which
# includes
# all sub-groups for an aggregation policy with split by fields. If you
# exceed this
# limit, you will break all sub-groups that exist for an aggregation
# policy.
max_event_in_parent_group = 100000000

# The number of events that can be contained in a single sub-group for
# an aggregation
# policy with split by fields. If you exceed this limit you will break
# the sub-group.
max_event_in_group = 10000

# An ACK token ensures that an event is being indexed before running an
# action on it.
# However, events are forwarded to the indexer from the search head,
# which adds another delay.
# This field (in milliseconds) adds an additional delay before running
# an action on events or groups.
# This setting can help you avoid missing notable events while grouping
# on a slow deployment.
action_execution_delay = 0

# When fetching events to perform actions on an episode, the amount of
time, in seconds, to
# subtract from the earliest_time on the search before executing an
# action.
# This setting helps prevent grouping inaccuracies when events are
# milliseconds apart.
earliest_time_lag = 300

# The delay, in seconds, to batch update episode state. Otherwise, the
Ingest third-party alerts as ITSI notable events

ITSI lets you ingest alerts from third-party products, such as Nagios and SCOM, as notable events. You can then manage those notable events in Episode Review. The ability to ingest third-party alerts as notable events lets you use ITSI as an efficient central management console, where you can view, triage, and investigate alerts from integrated products.

Use a custom integration script

ITSI provides custom scripts with specific onboarding instructions for you to ingest alerts from common third-party systems. Each blog post provides a script with a custom correlation search and notable event aggregation policy that you can import directly into ITSI. If you use one of these custom scripts, you do not need to follow the regular onboarding steps below.

ITSI provides the following custom onboarding instructions:

♦ Ingest Netcool Alerts into Splunk ITSI Event Analytics
♦ Ingest Nagios Alarms into Splunk ITSI Event Analytics
♦ Ingest AppDynamics and New Relic Alerts into Splunk ITSI Event Analytics

Step 1: Ingest alerts into ITSI

There are various methods to get third-party alerts into ITSI. It is recommended that for each alert sender, you have a different sourcetype. Use any of the following methods to ingest alerts into ITSI:

♦ Set up an email receiver bot. Email alerts are received, parsed, and stored to disk in a directory where a Splunk universal forwarder monitors and collects the data.
♦ Set up a polling mechanism that uses the third-party service’s API calls to fetch new alerts every minute and store this information into Splunk.
♦ Use a Splunk universal forwarder to monitor the application logs on the alert-generating servers. The universal forwarder catches the alert in the log files when the tool fires alerts.
Step 2: Normalize the alert data

After you ingest alerts into ITSI, you must normalize the fields so that they display properly in Episode Review. Use an ITSI correlation search to normalize third-party alerts as ITSI notable events. A correlation search is an ad hoc Splunk search that enriches, normalizes, and deduplicates raw alerts. Each resulting row from the search becomes an ITSI notable event.

ITSI correlation searches support field substitution with tokens in the format %fieldname%. Use field substitution to map third-party alert field values to corresponding notable event fields.

Token variables cannot be the same as correlation search field names. For example, you cannot use ?title, ?description, ?owner, ?severity, ?status, and so on, as variables. Instead use token variables such as %orig_description%, %orig_owner%, %orig_severity%, %orig_status%, and so on.

1. In ITSI, click Configure > Correlation Searches.
2. Click Create New Search > Create Correlation Search.
3. Configure a correlation search to ingest third party alerts.

Search Properties

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search Name</td>
<td>A name that describes the correlation search. For example, &quot;nagios_alert&quot;.</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>A description of the type of issue the search is intended to detect.</td>
<td>None</td>
</tr>
<tr>
<td>Search Type</td>
<td>Enter an ad hoc search string with appropriate eval command expressions to map fields in your third-party alert data to ITSI notable event fields. For example, the following ad hoc search maps Nagios severity-level fields to ITSI severity-level fields:</td>
<td>Ad hoc</td>
</tr>
</tbody>
</table>

```
index="main" sourcetype="csv" | head 100 | eval orig_severity=case(severity=="OK", 3, severity=="WARNING", 4,
```
<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>severity=&quot;CRITICAL&quot;, 6, severity=&quot;UNKNOWN&quot;, 5)</td>
<td></td>
</tr>
<tr>
<td>Time range</td>
<td>The time range over which the correlation search applies. Choose Relative and run the search over the last 1 minute or last 5 minutes.</td>
<td>Last 15 minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Association**

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service</td>
<td>The ITSI services to which this correlation search applies. Leave this field blank.</td>
<td>None</td>
</tr>
<tr>
<td>Entity Lookup Field</td>
<td>The field in the data retrieved by the correlation search that is used to look up corresponding entities. This is usually a host or node field (for example, itsiNode). If the host or node of the alert is listed as a filtered entity in one or more ITSI services, these service names are added to the notable event as ?affected service(s)?.</td>
<td>None</td>
</tr>
</tbody>
</table>

**Schedule**

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schedule Type</td>
<td>Choose a Basic schedule. Configure the search interval in the Run Every menu to run every 1 minute or 5 minutes (depending on the time range above).</td>
<td>Basic, Every 5 minutes</td>
</tr>
</tbody>
</table>

**Notable Events**

Use this section to configure the notable event that is generated when search results meet a specific condition. The Splunk platform indexes the event object like any other event. You can track, manage, and update
notable events in Episode Review.

ITSI correlation searches support field substitution with tokens in the format `%fieldname%`. Use field substitution to map third-party alert field values to corresponding notable event fields. To see the available fields that you can use as tokens, click **Run Search** under Search Properties. You can use any of the fields listed in the left panel.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notable Event Title</td>
<td>The title of the notable event in Episode Review. For example, you might enter <code>%name%</code> to substitute Nagios alert names as notable event titles.</td>
<td>None</td>
</tr>
<tr>
<td>Notable Event Description</td>
<td>A brief phrase to describe the notable event. For example, you might enter <code>%reason%</code> to substitute Nagios descriptions for notable event descriptions.</td>
<td>None</td>
</tr>
<tr>
<td>Owner</td>
<td>The ITSI role to which the notable event is assigned in Episode Review. Click <strong>Advanced Mode</strong>, then enter the owner using a token in the format <code>%fieldname%</code>. For example, <code>%owner_field%</code>. In advanced mode, owners must match the names of owners in the system.</td>
<td>Unassigned</td>
</tr>
<tr>
<td>Severity</td>
<td>The level of importance of the event. Click <strong>Advanced Mode</strong>, then provide a token in the format <code>%fieldname%</code> to substitute the value of a third-party alert field. Values must match an integer specified in the default version of itsi_notable_event_severity.conf (or the local version if you created one).</td>
<td>1 - Info</td>
</tr>
<tr>
<td>Status</td>
<td>The triage status of the event in Episode Review. Click <strong>Advanced</strong></td>
<td>0 - Unassigned</td>
</tr>
<tr>
<td>Field name</td>
<td>Description</td>
<td>Defaults</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------</td>
</tr>
</tbody>
</table>
| **Mode**                    | Mode, then provide a token in the format `%%fieldname%%` to substitute the value of a third-party alert field. Values must match an integer specified in the default version of `itsu_notable_event_status.conf` (or the local version if you created one). For incoming third-party alerts the triage status is generally "New". | 1 - New  
2 - In Progress  
3 - Pending  
4 - Resolved  
5 - Closed |
| Drilldown Name              | You can drill down to a specific Splunk search from the All Events tab of Episode Review. Set the name of the drilldown search link.                         | None                      |
| Drilldown Search            | The search you drill down to.                                                | None                      |
| Drilldown earliest offset   | Defines how far back from the time of the event to start looking for related events.                                           | Last 5 minutes            |
| Drilldown latest offset     | Defines how far ahead from the time of the event to look for related events.                                                  | Next 5 minutes            |
| Notable Event Identifier Fields | Determine whether a notable event is unique or not, which drives the events timeline. These identifier fields form the event hash field, which is added to every notable event to help identify unique alarm types. | source                    |
| Drilldown link title        | You can drill down to a specific website from the All Events tab of Episode Review. Set the name of the drilldown website link. | None                      |
| Drilldown link URI          | The website you drill down to.                                               | None                      |

4. Click **Save**.
The correlation search runs as scheduled, mapping third-party alert field values to ITSI notable event fields. Third-party alerts appear as notable events in Episode Review. The events are collected in the itsi_tracked_alerts index.

**Step 3: Aggregate the events**

After you ingest and normalize third-party alerts, they are collected in the itsi_tracked_alerts index as notable events and are ready to be grouped into episodes.

A notable event aggregation policy called Splunk App for Infrastructure is delivered with ITSI. This aggregation policy groups all third-party alerts (including alerts from the Splunk App for Infrastructure) that use ITSI normalized fields. Enable this aggregation policy or configure your own aggregation policy to group third-party alert data.

1. In ITSI, click **Configure > Notable Event Aggregation Policies**.
2. Click **New Notable Event Aggregation Policy**, or configure the Default Policy. See Notable event aggregation policies overview for ITSI.

For example, you might configure the Default Policy by enabling Smart Mode, then selecting several notable events to cluster on (such as itsiNode, itsiAlertId, Location, and OS_version).

After you enable an aggregation policy, episodes start appearing in Episode Review.

ITSI can also take automated actions at the beginning or end of an episode. Consider what makes the most sense for a given type of alert. Alert actions might include opening a ServiceNow ticket, sending an email, or running a custom script. For more information, see Take action on an episode in ITSI.

**Ingest SNMP traps in ITSI**

Simple Network Management Protocol (SNMP) is an application-layer protocol used to manage and monitor network devices and their functions. Learn how to ingest SNMP traps into Splunk enterprise and feed them into ITSI Episode Review for easy review and correlation.

After you configure Splunk enterprise and ITSI to receive and process SNMP traps, enable the default correlation search and aggregation policy to view and
triage the traps in Episode Review. For more information, see Triage episodes in ITSI in the Splunk ITSI User Manual.

Prerequisites

1. To ingest SNMP traps in ITSI, you must install and configure snmptrapd, an application that receives and logs SNMP TRAP and INFORM messages and resolves them using vendor Management Information Bases (MIBs). Before you install snmptrapd on your system, see the manual page for snmptrapd.
2. To configure the logging format, open snmptrapd.conf and add the following stanza:

```
# snmptrapd formatting
# from_3rd_party_Vendors

# SNMPV1
format1 Agent_Address = %A\nAgent_Hostname = %B\nDate = %y-%02.2m-%02.2l
%02.2h:%02.2j:%02.2k\nEnterprise_OID = %N \nTrap_Type = %w\nTrap_SubType = %q\nCommunity_Infosec_Context = %P\nUptime = %T\nDescription = %W\nPDU_Attribute_Value_Pair_Array:
%V
%v
---

# SNMPV2
format2 Agent_Address = %A\nAgent_Hostname = %B\nDate = %y-%02.2m-%02.2l
%02.2h:%02.2j:%02.2k\nEnterprise_OID = %N \nTrap_Type = %w\nTrap_SubType = %q\nCommunity_Infosec_Context = %P\nUptime = %T\nDescription = %W\nPDU_Attribute_Value_Pair_Array:
%V
%v
---
```

Steps

Perform the following steps to ingest SNMP traps into ITSI:

1. Send SNMP events to your Splunk deployment.
2. Configure the snmptrapd index.
3. Enable the SNMP correlation search and aggregation policy.

Send SNMP events to your Splunk deployment

For instructions on getting SNMP events into your Splunk deployment, see Send
SNMP events to your Splunk deployment in the Splunk Enterprise *Getting Data In* manual.

**Configure the snmptrapd index**

After you install and configure `snmptrapd`, you need to tell Splunk software and ITSI where to find the SNMP data and how to parse it. Use the following ITSI configuration files to instruct Splunk software to monitor the file that `snmptrapd` is writing to, and to parse and extract the key-value pairs that make up important information contained in the traps.

Where you deploy these configurations depends on the type of Splunk deployment you have. For more information, see Install ITSI in a distributed environment in this manual.

**inputs.conf**

Add the following stanza to the local version of `inputs.conf`:

```
[monitor:///var/log/snmptrapd.log]
disabled = false
index = snmptrapd
sourcetype = snmptrapd
```

This stanza tells your Splunk instance where to look for your data. Deploy this configuration on forwarders or Splunk instances co-located with `snmptrapd`.

**props.conf**

Add the following stanza to the local version of `props.conf`:

```
[snmptrapd]
DATETIME_CONFIG =
KV_MODE = none
LINE_BREAKER = {[
]+}Agent_Address\s=
MAX_TIMESTAMP_LOOKAHEAD = 30
NO_BINARY_CHECK = true
SHOULD_LINEMERGE = false
TIME_FORMAT = %Y-%m-%d %H:%M:%S
TIME_PREFIX = Date\s=\s
TZ = UTC
category = Custom
description = parse snmptrapd logging with custom kvpair splunk formatting
disabled = false
```
This stanza sets line-breaking rules and extracts the timestamp. In this case, the timestamp is the time that the `snmptrapd` server received the trap. This stanza also ensures that Splunk knows how to parse the time format.

Deploy this configuration to your heavy forwarder, indexer, and search heads.

`transforms.conf`

Add the following stanza to the local version of `transforms.conf`:

```
[snmptrapd_kv]
DELIMS = "\n," = ""
```

This stanza extracts all key-value pairs in the SNMP trap. Deploy this configuration to your heavy forwarder, indexer, and/or search heads.

`default.meta.conf`

Add the following stanza to the local version of `default.meta.conf`:

```
[]
access = read : [ * ], write : [ admin ]
export = system
```

This stanza includes a `local.meta.conf` file in the ITSI application that exports permissions to the system. Deploy this configuration on your search heads.

**Enable the SNMP correlation search and aggregation policy**

ITSI provides the following default correlation search and aggregation policy for processing SNMP traps:

**SNMP Traps**

A default correlation search that generates a notable event for each SNMP trap received through Splunk.

**Default SNMP Policy**

A default notable event aggregation policy that aggregates individual SNMP trap events into episodes in Episode Review.

**Steps**
1. To enable the correlation search, click **Configure > Correlation Searches** and enable the **SNMP Traps** search in the Status column.

2. To enable the aggregation policy, click **Configure > Notable Event Aggregation Policies** and enable the **Default SNMP Policy** in the Status column.

## Use the Notable Event Actions SDK

A new, refactored version of the Notable Event Actions SDK, called the ITSI Event Management SDK, is available on GitHub here. This documentation pertains to the legacy Notable Event Actions SDK that continues to be packaged with ITSI. Scripts based on the legacy SDK will continue to work, however, it is recommended to eventually port your scripts to use the new SDK as only the new SDK will be updated going forward.

The Notable Events Action SDK comes packaged with ITSI and is located in `$SPLUNK_HOME/etc/apps/SA-ITOA/lib/itsi/event_management/sdk`. The SDK is implemented in Python and includes:

1. Methods for a Custom Action on an ITSI notable event: `class CustomEventActionBase in custom_event_action_base.py`
2. Methods for working on an ITSI notable event post-custom action: `class Event in eventing.py`
3. Methods for finding metadata for ITSI notable events: `class EventMeta in eventing.py`

Custom actions require a class that derives from `CustomEventActionBase` and implements the `execute()` method. The `__init__()` method of your class needs to initialize the base class `CustomEventActionBase`.

For more information, see the Notable events action SDK reference in this manual.

The following is a summary of the custom action Ping Host that ships with ITSI:

Because individual notable events are immutable in version 4.0.0 and later, you must pass the unique ID of an episode (`itsi_group_id`) instead of the ID of an individual notable event. Therefore, the previously `event_id` parameter is now `itsi_group_id` in the example below.

```python
class Ping(CustomEventActionBase):
```
def __init__(self, settings):
    # initialize CustomEventActionBase
    super(Ping, self).__init__(settings, self.logger)

def get_host_to_ping(self):
    # from input settings, fetch host to ping
    # some logic is abstracted in custom_event_action_base.py
    ...
    return host

def ping(self, host):
    # does the act of pinging the host
    ...

def execute(self):
    # has all the logic of ping in here...
    host = self.get_host_to_ping()
    std_out, std_err = self.ping(host)

    # do other stuff here, like add a comment to an ITSI episode
    # or add a few tags, and so on....
    # change the state of the episode....
    for data in self.get_event():
        itsi_group_id = data.get('itsi_group_id')
        event = Event(self.get_session_key(), self.logger)
        event.create_comment(itsi_group_id, comment)
        event.create_comment(itsi_group_id, out)
        event.create_tag(itsi_group_id, 'ping')
    return

if __name__=='__main__':
    if len(sys.argv) > 1 and sys.argv[1] == '--execute':
        input_params = sys.stdin.read()
        ping = Ping(input_params)
        ping.execute()
Use the `custom_event_action_base` module when you want to consume the output of your moderator alerts (modalert).

Use the `eventing` module to work with your events in ITSI. This module must reside on the same host where ITSI is installed. All classes in this module inherit from the `EventBase` class. This base class tracks the session key, username, password, and logger info.

```python
class custom_event_action_base.CustomEventActionBase(settings, logger)
```

Import the `custom_event_action_base` module when you want to consume the output of your modalert. In your script, inherit your class from the `CustomEventActionBase` class. Use the `get_event()` method to work on the event that triggered your script.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>settings</td>
<td>basestring or dict</td>
<td>Incoming parameters passed to the script using stdin.</td>
</tr>
<tr>
<td>logger</td>
<td>logger</td>
<td>The logger instance of the inherited class.</td>
</tr>
</tbody>
</table>

**Raises**

**Type Error** - Raised when `settings` is not a serialized JSON string.

**Returns**

A `CustomEventActionBase` instance.

**Return type**

`CustomEventActionBase`.

**execute()**

Implement this method to configure the actions to take based on the alert output. Derived classes must implement this method.

**Raises**

**Not Implemented Error** - Raised when the `execute()` method is not implemented in the derived class.
**extract_event_id(notable_data)**

Use this method to get the event ID of the event that triggered the action.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notable_data</td>
<td>basestring</td>
<td>Notable event object from which the event_ID is extracted.</td>
</tr>
</tbody>
</table>

*Raises*

*Type Error* - Raised if notable_data is of type None.

*Exception* - Raised if notable_data is not JSON type data.

*Returns*

- The event_id.

*Return type*

- basestring.

**get_config()**

Gets the configuration based on the settings.

*Returns*

- The configuration that comes with the settings.

*Return type*

- Configuration.

**get_event()**

Gets the events that triggered the custom action. This method relies on get_results_file() and expects the returned file path to be a .csv.gz format.

The method is implemented as a generator because there can be thousands of events.

*Raises*

*Value Error* - Raised when there is not a valid .csv.gz results file. It prints out what file format was returned instead.

*Returns*

- A dict type object until all received events are returned.

*Return type*
dict.

**get_results_file()**

Gets the results file, which is where results are temporarily stored.

**Raises**

- **Attribute Error**: Raised when no results file is returned.

**Returns**

The results file.

**Return type**

Results file.

**get_session_key()**

Gets the Splunk session key.

**Raises**

- **Attribute Error**: Raised when no session key is returned.

**Returns**

The session key.

**Return type**

basestring.

**class eventing.EventMeta(*args)**

Import this class to get information about ITSI events, such as status, owner, and severities.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_key</td>
<td>basestring</td>
<td>The session key that is given by splunkd when you log in. If you do not have a session key, pass in None, but you will need to provide username and password credentials.</td>
</tr>
<tr>
<td>username</td>
<td>basestring</td>
<td>Your username.</td>
</tr>
<tr>
<td>password</td>
<td>basestring</td>
<td>Your password.</td>
</tr>
<tr>
<td>logger</td>
<td>logger</td>
<td>A place to log actions in an existing logger. If you do not have a logger, then the ITSI logger is used.</td>
</tr>
</tbody>
</table>
Raises

Type Error - Raised when the `session_key` is not a basestring type or when the `session_key` is not valid.
Value Error - Raised when the `session_key` is a zero-length string, or when the `username` and `password` are empty strings and the `session_key` is of type None.

Returns
An `EventMeta` instance.

Return type
EventMeta.

get_all_owners()
Get all valid owner values for a notable event.

Returns
A list of all configured owners.

Return type
list.

get_all_severities()
Get all severity values for a notable event.

Returns
A list of all configured severities.

Return type
list.

get_all_statuses()
Get all status values for a notable event.

Returns
A list of all configured statuses.

Return type
list.
class eventing.Event(session_key, *args, **kwargs)

Import this class to operate on ITSI events.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_key</td>
<td>basestring</td>
<td>The session key that is given by splunkd when you log in. If you do not have a session key, pass in None, but you will need to provide username and password credentials.</td>
</tr>
<tr>
<td>username</td>
<td>basestring</td>
<td>Your username.</td>
</tr>
<tr>
<td>password</td>
<td>basestring</td>
<td>Your password.</td>
</tr>
<tr>
<td>logger</td>
<td>logger</td>
<td>A place to log actions in an existing logger. If you do not have a logger, then the ITSI logger is used.</td>
</tr>
</tbody>
</table>

**Raises**

Type Error - Raised when the session_key is not a basestring type or when the session_key is not valid.
Value Error - Raised when the session_key is a zero-length string, or when the username and password are empty strings and the session_key is of type None.

**Returns**

An Event instance.

**Return type**

Event.

create_comment(event_id, comment)

Add a new comment, given an event ID. Duplicate comments are not allowed for any given event.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_id</td>
<td>basestring</td>
<td>An event ID.</td>
</tr>
<tr>
<td>comment</td>
<td>basestring</td>
<td>A comment to add to the event.</td>
</tr>
</tbody>
</table>

**Raises**

Type Error - Raised when the event_id or comment are empty strings.
**Returns**

The event ID, the comment ID, and the comment.

**Return type**

dict.

`create_tag(*args)`

Create a tag, given an event ID.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_id</td>
<td>basestring</td>
<td>An event ID.</td>
</tr>
<tr>
<td>tag_value</td>
<td>basestring</td>
<td>New value for the tag.</td>
</tr>
<tr>
<td>raise_exceptions</td>
<td>bool</td>
<td>Raise an exception. By default this value is False</td>
</tr>
</tbody>
</table>

**Raises**

Type Error - Raised when the `event_id` or `tag_value` is an empty string.

**Returns**

The event ID, the new tag, and the the tag ID.

**Return type**

dict.

`delete_all_comments(event_id)`

Delete all of the comments associated with an event, given the event ID.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_id</td>
<td>basestring</td>
<td>An event ID.</td>
</tr>
</tbody>
</table>

**Raises**

Type Error - Raised when the `event_id` is an empty string.

**Returns**

Nothing.

`delete_all_tags(event_id)`

Delete all of the tags associated with an event, given an event ID.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_id</td>
<td>basestring</td>
<td>An event ID.</td>
</tr>
</tbody>
</table>

**Raises**

**Type Error** - Raised when the `event_id` is an empty string.

**Returns**

Nothing.

---

`delete_comment(comment_id)`

Delete the comment associated with the comment ID.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>comment_id</td>
<td>basestring</td>
<td>The ID of the comment to delete.</td>
</tr>
</tbody>
</table>

**Raises**

**Type Error** - Raised when the `comment_id` is an empty string.

**Returns**

Nothing.

---

`delete_tag(tag_id)`

Delete the tag value, given a tag ID.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tag_id</td>
<td>basestring</td>
<td>The ID of the tag to be retrieved.</td>
</tr>
</tbody>
</table>

**Raises**

**Type Error** - Raised when the `tag_id` is an empty string.

**Returns**

Nothing.

---

`delete_ticket_info(event_ids, ticket_system, ticket_id)`

Delete external ticket information, given a list of event IDs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
event_ids | basestring or list | A comma separated string of event IDs, or a list of event IDs.
ticket_system | basestring | The name of an external ticket system.
ticket_id | basestring | The external ticket ID.

**Raises**

**Type Error** - Raised when the `event_ids` is not of type basestring or of type list.

**Value Error** - Raised when the `event_ids` is an empty string or list.

**Returns**

The event IDs of successful and failed deletions.

**Return type**
dict.

**get_all_comments(event_id)**

Get all of the comments associated with an event, given an event ID.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_id</td>
<td>basestring</td>
<td>An event ID.</td>
</tr>
</tbody>
</table>

**Raises**

**Type Error** - Raised when the `event_id` is an empty string.

**Returns**

All of the comments corresponding to the event ID.

**Return type**
list.

**get_all_tags(event_id)**

Get all of the tags, given an event ID.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_id</td>
<td>basestring</td>
<td>An event ID.</td>
</tr>
</tbody>
</table>

**Raises**

**Type Error** - Raised when `event_id` is an empty string.
**Returns**
All of the existing tags for a given event.

**Return type**
list.

**get_comment(comment_id)**

Get the comment, given a comment ID.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>comment_id</td>
<td>basestring</td>
<td>The ID of the comment to get.</td>
</tr>
</tbody>
</table>

**Raises**

**Type Error** - Raised when the comment_id is an empty string.

**Returns**
The comment corresponding to the ID.

**Return type**
basestring.

**get_owner(*args, **kwargs)**

Return the owners, given a list of events or event IDs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>events</td>
<td>list of dicts</td>
<td>Each dict in the list represents an event that was sent by Splunk as an outcome of a Custom Action. The get_event() method in the CustomEventActionBase class generates such an item.</td>
</tr>
<tr>
<td>event_ids</td>
<td>basestring or list</td>
<td>A unique ID for an event.</td>
</tr>
<tr>
<td>split_by</td>
<td>basestring</td>
<td>If event_ids is of type basestring, then it is split into a list of event IDs. By default, ',' separates the event IDs.</td>
</tr>
<tr>
<td>**kwargs</td>
<td>dict</td>
<td>A dict with the keys earliest_time and latest_time with corresponding values.</td>
</tr>
</tbody>
</table>

**Raises**
**Type Error** - Raised when `events` is not of type `basestring` or of type list.

**Value Error** - Raised when both `events` and `event_ids` are of type `None`.

**Returs**
The event IDs and corresponding owners.

**Return type**
list of tuples.

### `get_status(*args, **kwargs)`

Return the statuses, given a list of events or event IDs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>events</code></td>
<td>list of dicts</td>
<td>Each dict in the list represents an event that was sent by Splunk as an outcome of a Custom Action. The <code>get_event()</code> method in the <code>CustomEventActionBase</code> class generates such an item.</td>
</tr>
<tr>
<td><code>event_ids</code></td>
<td>basestring or list</td>
<td>A unique ID for an event.</td>
</tr>
<tr>
<td><code>split_by</code></td>
<td>basestring</td>
<td>If <code>event_ids</code> is of type <code>basestring</code>, then it is split into a list of event IDs. By default, ',' separates the event IDs.</td>
</tr>
<tr>
<td><code>**kwargs</code></td>
<td>dict</td>
<td>A dict with the keys <code>earliest_time</code> and <code>latest_time</code> with corresponding values.</td>
</tr>
</tbody>
</table>

### Raises

**Type Error** - Raised when `events` is not of type list.

**Value Error** - Raised when both `events` and `event_ids` are of type `None`.

**Returs**
The event IDs and corresponding statuses.

**Return type**
list of tuples.

### `get_severity(*args, **kwargs)`

Return the severities, given a list of events or event IDs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

188
Each dict in the list represents an event that was sent by ITSI as an outcome of a Custom Action. The `get_event()` method in the `CustomEventActionBase` class generates such an event.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>events</code></td>
<td>list of dicts</td>
<td>A unique ID for an event.</td>
</tr>
<tr>
<td><code>event_ids</code></td>
<td>basestring / list</td>
<td>If <code>event_ids</code> is of type basestring, then it is split into a list of event IDs. By default, ',' separates the event IDs.</td>
</tr>
<tr>
<td><code>split_by</code></td>
<td>basestring</td>
<td>A dict with the keys <code>earliest_time</code> and <code>latest_time</code> with corresponding values.</td>
</tr>
<tr>
<td><code>**kwargs</code></td>
<td>dict</td>
<td>A dict with the keys <code>earliest_time</code> and <code>latest_time</code> with corresponding values.</td>
</tr>
</tbody>
</table>

**Raises**

- **Type Error** - Raised when `events` is not of type basestring or of type list.
- **Value Error** - Raised when both `events` and `event_ids` are of type None.

**Returns**

The event IDs and corresponding severities.

**Return type**

list of tuples.

### get_tag(tag_id)

Get the value of a tag, given a tag ID.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tag_id</code></td>
<td>basestring</td>
<td>The ID of the tag to be retrieved.</td>
</tr>
</tbody>
</table>

**Raises**

- **Type Error** - Raised when the `tag_id` is an empty string.

**Returns**

The tag value corresponding to the tag ID.

**Return type**

basestring.
**update(*args, **kwargs)**

Update the status, severity, and owner of each event individually.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>blob</td>
<td>list / dict</td>
<td>A list of dicts containing event IDs and the severity, status, and/or owner that they should be updated to.</td>
</tr>
<tr>
<td>split_by</td>
<td>basestring</td>
<td>If the event_ids in blob is of type basestring, then it is split into a list of event IDs. By default, ',' separates the event IDs.</td>
</tr>
<tr>
<td>**kwargs</td>
<td>dict</td>
<td>A dict with the keys earliest_time and latest_time with corresponding values.</td>
</tr>
</tbody>
</table>

**Raises**

- **Key Error** - Raised when the event_ids key is missing from the blob dict.
- **Type Error** - Raised when blob is neither of type list nor of type dict.
- **Value Error** - Raised when blob is empty.

**Returns**

The event ID schemas with updated fields, in the order of input.

**Return type**

dict.

**update_comment(*args)**

Update a comment, given a comment ID and an event ID. Duplicate comments are not allowed for any given event.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_id</td>
<td>basestring</td>
<td>An event ID.</td>
</tr>
<tr>
<td>comment_id</td>
<td>basestring</td>
<td>The ID of the comment to update.</td>
</tr>
<tr>
<td>comment</td>
<td>basestring</td>
<td>The new value for the comment.</td>
</tr>
</tbody>
</table>

**Raises**

- **Type Error** - Raised when the event_id or comment_id is an empty string.

**Returns**
The committed comment value.

**Return type**
dict.

`update_owner(*args, **kwargs)`

Update the owner of each event, given a list of event IDs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_ids</td>
<td>basestring or list</td>
<td>A comma separated string of event IDs or a list of event IDs.</td>
</tr>
<tr>
<td>owner</td>
<td>basestring</td>
<td>The owner of the event.</td>
</tr>
<tr>
<td>split_by</td>
<td>basestring</td>
<td>If the event_ids is of type basestring, then it is split into a list of event IDs. By default, ',' separates the event IDs.</td>
</tr>
<tr>
<td>**kwargs</td>
<td>dict</td>
<td>A dict with the keys earliest_time and latest_time with corresponding values.</td>
</tr>
</tbody>
</table>

**Raises**

- **TypeError** - Raised when event_ids is neither of type basestring nor of type list.
- **Value Error** - Raised when event_ids is an empty list, or when owner is not a valid string.

**Returns**
The event ID schemas with updated owners, in the order of input.

**Return type**
dict.

`update_severity(*args, **kwargs)`

Update the severity value of each event, given a list of event IDs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_ids</td>
<td>basestring or list</td>
<td>A comma separated string of event IDs, or a list of event IDs.</td>
</tr>
<tr>
<td>severity</td>
<td>basestring</td>
<td>One of the supported severity values</td>
</tr>
<tr>
<td>split_by</td>
<td>basestring</td>
<td>If the event_ids is of type basestring, then it is split into a list of event IDs. By default, ',' separates the event IDs.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>event IDs.</td>
</tr>
<tr>
<td>**kwargs</td>
<td>dict</td>
<td>A dict with the keys <code>earliest_time</code> and <code>latest_time</code> with corresponding values.</td>
</tr>
</tbody>
</table>

**Raises**

- **TypeError** - Raised when `event_ids` is neither of type basestring nor of type list.
- **Value Error** - Raised when `event_ids` is an empty list, or when `severity` is not a valid string.

**Returns**

The event ID schemas with updated severities, in the order of input.

**Return type**

dict.

**update_status(*args, **kwargs)**

Update the status of each event, given a list of event IDs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>event_ids</code></td>
<td>basestring or list</td>
<td>A comma separated string of event IDs, or a list of event IDs</td>
</tr>
<tr>
<td><code>status</code></td>
<td>basestring</td>
<td>One of the supported statuses</td>
</tr>
<tr>
<td><code>split_by</code></td>
<td>basestring</td>
<td>If the <code>event_ids</code> is of type basestring, then it is split into a list of event IDs. By default, ',' separates the event IDs.</td>
</tr>
<tr>
<td>**kwargs</td>
<td>dict</td>
<td>A dict with the keys <code>earliest_time</code> and <code>latest_time</code> with corresponding values.</td>
</tr>
</tbody>
</table>

**Raises**

- **TypeError** - Raised when `event_ids` is neither of type basestring nor of type list.
- **Value Error** - Raised when `event_ids` is an empty list, or when `status` is not a valid string.

**Returns**

The event ID schemas with updated statuses, in the order of input.

**Return type**

dict.
**update_tag(*args)**

Update the tag, given an event ID and a tag ID.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_id</td>
<td>basestring</td>
<td>An event ID.</td>
</tr>
<tr>
<td>tag_id</td>
<td>basestring</td>
<td>The tag ID.</td>
</tr>
<tr>
<td>tag_value</td>
<td>basestring</td>
<td>The new value for the tag.</td>
</tr>
</tbody>
</table>

**Raises**

**Type Error** - Raised when the `event_id` or `tag_id` are empty strings.

**Returns**

The committed value.

**Return type**

dict.

**update_ticket_info(*args, **kwargs)**

Update events with external ticket information, given a list of event IDs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_ids</td>
<td>basestring or list</td>
<td>A comma separated string of event IDs, or a list of event IDs</td>
</tr>
<tr>
<td>ticket_system</td>
<td>basestring</td>
<td>The name of an external ticket system</td>
</tr>
<tr>
<td>ticket_id</td>
<td>basestring</td>
<td>The external ticket ID</td>
</tr>
<tr>
<td>ticket_url</td>
<td>basestring</td>
<td>The URL to reach the external ticket</td>
</tr>
<tr>
<td><strong>kwargs</strong></td>
<td>dict</td>
<td>A dict of additional parameters to add to the ticket information.</td>
</tr>
</tbody>
</table>

**Raises**

**Type Error** - Raised when the `event_ids` is not of type basestring or of type list.

**Value Error** - Raised when the `event_ids` is an empty string or list.

**Returns**

The event IDs of successful and failed updates.

**Return type**

dict.
dict.

class eventing.EventGroup(session_key, *args, **kwargs)

Import this class to operate on an ITSI Event Group.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_key</td>
<td>basestring</td>
<td>The session key that is given by splunkd when you log in. If you do not have a session key, pass in None, but you will need to provide username and password credentials.</td>
</tr>
<tr>
<td>username</td>
<td>basestring</td>
<td>Your username.</td>
</tr>
<tr>
<td>password</td>
<td>basestring</td>
<td>Your password.</td>
</tr>
<tr>
<td>logger</td>
<td>logger</td>
<td>A place to log actions in an existing logger. If you do not have a logger, then the ITSI logger is used.</td>
</tr>
</tbody>
</table>

**Raises**

- **Type Error** - Raised when the session_key is not a basestring type or when the session_key is not valid.
- **Value Error** - Raised when the session_key is a zero-length string, or when the username and password are empty strings and the session_key is of type None.

**Returns**

An EventGroup instance.

**Return type**

EventGroup

add_drilldown(self, group_id, drilldown)

Adds drilldown to a notable event group.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group_id</td>
<td>basestring</td>
<td>The ID of the group where add_drilldown is operated on.</td>
</tr>
<tr>
<td>drilldown</td>
<td>dict</td>
<td>The drilldown data that is added.</td>
</tr>
</tbody>
</table>

**Raises**
Attribute Error - Raised when the group_id not available.
Value Error - Raised when drilldown data is not dict with link and name.

Returns
Return dict which holds updated keys.

Return type
dict.

update_drilldown(self, group_id, drilldown)

Updates a drilldown for a notable event group.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group_id</td>
<td>basestring</td>
<td>The ID of the group who owns the drilldown that is updated.</td>
</tr>
<tr>
<td>drilldown</td>
<td>dict</td>
<td>The drilldown that is updated.</td>
</tr>
</tbody>
</table>

Raises

Value Error - Raised when drilldown data is not dict with link and name.

Returns
Return dict which holds updated keys.

Return type
dict.

delete_drilldown(self, group_id, drilldown)

Deletes a drilldown for a notable event group.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group_id</td>
<td>basestring</td>
<td>The ID of the group who owns the drilldown that is deleted.</td>
</tr>
<tr>
<td>drilldown</td>
<td>dict</td>
<td>The drilldown that is deleted.</td>
</tr>
</tbody>
</table>

Raises

Value Error - Raised when drilldown data is not dict with link and name.

Returns
Return dict which holds updated keys.
Return type
dict.
Aggregation Policies

Notable event aggregation policies overview for ITSI

Notable event aggregation policies help you group notable events into episodes to organize them in Episode Review. ITSI provides a default aggregation policy to group notable events. You can also create your own aggregation policies if you are familiar with your data and want to define very precisely how events are grouped. You can use Smart Mode on any aggregation policy to employ machine learning algorithms to group events. The process of managing notable events through the use of aggregation policies is often referred to as "event analytics."

Aggregation policies group notable events based on rules that you define. You can also consolidate duplicate events, suppress alerts, or close episodes when a clearing event is received.

Aggregated notable events are displayed in Episode Review when event grouping is enabled in View Settings. These episodes have their own title, description, severity, status, and assignee, separate from the individual notable events within the episode.

A notable event can belong to multiple episode if it matches the criteria for those episodes.

The following aggregation policies are delivered with ITSI:

- Default Policy
- Default SNMP Policy
- Normalized Policy (Splunk App for Infrastructure)

See also

- Use the default aggregation policy in ITSI
- Create a custom aggregation policy in ITSI

Use the default aggregation policy in ITSI

The ITSI default aggregation policy receives notable events that do not match the filtering criteria of any other aggregation policies. If you do not want to create
your own aggregation policies, you can use the default policy to group events. If any aggregation policies have been created and are enabled, the default policy will capture only events that do not match the criteria of one of these policies.

Only a user assigned the itoa_admin role, or a role that inherits from itoa_admin, can modify the default aggregation policy.

The default aggregation policy is defined as follows:

♦ Does not include any filtering criteria. The default policy catches events not captured by the filtering criteria of any other aggregation policies. None can be added.
♦ Splits events into multiple episodes by the source field. You can change the field that is used to split events, specify more than one field by which to split events, or choose to not split events by not specifying a field name.
♦ ITSI stops adding events to the episode if the flow of events into the episode is paused for 7200 seconds (2 hours). This can be changed to a different length of time or to different breaking criteria.
♦ Episode information such as Episode Title, Episode Description, and Episode Severity are set to be the same as the first event in the episode. You can change these if desired.
♦ No action rules are defined. You can add action rules if desired.

To view or modify the default aggregation policy, click Configure > Notable Event Aggregation Policies > Default Policy.

As delivered, the default policy does not have Smart Mode enabled. For information on enabling Smart Mode, see Group similar events with Smart Mode in ITSI.

For information on modifying the default aggregation policy or creating a new aggregation policy, see Create a custom aggregation policy in ITSI.

**Create a custom aggregation policy in ITSI**

You can create one or more customized notable event aggregation policies. By default, users assigned the itoa_admin role or itoa_team_admin role can create, modify, and delete aggregation policies. Users assigned the itoa_analyst role can view notable event aggregation policies but cannot modify or delete them.
If you want to use Smart Mode in a custom aggregation policy, you must first create and save the custom aggregation policy. After you save the policy, re-open it to turn on and configure Smart Mode.

**What aggregation policies do**

An aggregation policy lets you group notable events into episodes based on the following criteria:

- A field matches a string
- A field does not match a string
- A field is greater than a number
- A field is greater than or equal to a number
- A field is less than a number
- A field is less than or equal to a number

The field can be the name of a saved search, the name of a field from a raw event, or the Status, Severity, or Owner field of the notable event. You can create rules based on multiple AND/OR conditions.

An aggregation policy lets you control the grouping of events in the following ways:

- Split events into multiple episodes by one or more fields, such as source or host.
- Stop grouping when certain criteria are met, such as when a certain number of events is reached, the episode has existed for a certain amount of time, or when a certain event occurs, such as "Severity matches Normal". These are called breaking conditions. You can have multiple breaking conditions.

An aggregation policy lets you take the following actions on an episode (group of events):

- Change the severity
- Change the status
- Change the owner
- Add a comment
- Run a script
- Send an email
- Ping a host
- Create an incident in ServiceNow (requires the Splunk Add-on for ServiceNow)
Create an incident in Remedy (requires the Splunk Add-on for Remedy)
• Create an incident in VictorOps (requires VictorOps For Splunk)
• Other custom actions that have been configured

Note that not all actions may be available if role-based permissions are set. For information about episode actions, see Take action on an episode in ITSI.

Create an aggregation policy

When creating a new policy, be mindful of the policies that have already been created. Multiple aggregation policies can be created with filtering criteria that capture the same set or subset of events. If the same notable events are captured by more than one policy, the action rules from each policy are applied to the notable events.

To create an aggregation policy:

1. Click Configure > Notable Event Aggregation Policies. The default policy and any other policies that have been created are listed.
2. Click Create Notable Event Aggregation Policy. The Filtering Criteria step opens.

Filtering Criteria

You can create rules for determining which notable events are included in the group based on multiple AND/OR conditions. You can use wildcard characters in fields. You must specify at least one rule.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include the events if</td>
<td>Click + Add Rule (OR) to add rules for including events in this group. For example, you could specify severity matches Critical. To include events based on a saved search, type search_name matches &lt;name of saved search&gt;.</td>
</tr>
</tbody>
</table>

Click + Add Rule (AND) to add another rule to the rule block, or + Add Rule (OR) to start a new block of rules that is separated from the previous block by an OR condition.

For example, if there are three rules in the first block and two rules in the second block, the total clause would read "(Rule1 AND Rule2 AND Rule3) OR (Rule1 AND Rule2)". |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Split events by field</td>
<td>Split events into separate groups based on a field name. You can split by multiple fields, for example, host and source. After specifying one or more split by fields, the preview shows the events that meet the filtering criteria grouped by the specified fields. Click the arrow to the left of each group to expand the group and see the individual events contained in the group.</td>
</tr>
<tr>
<td>Break episode</td>
<td>Add breaking conditions to stop grouping. If the conditions are met, the current episode ends and a new episode is started. For example, an episode might end when the number of events reaches a certain number, the episode has existed for a certain amount of time, or when a certain event (that meets the filtering criteria for the episode) occurs. Use wildcards to search for a string in a field. For example, you might enter &quot;Break episode if the following event occurs: message matches &quot;status Normal&quot;. Note that the breaking event must be included in the episode (in other words it must meet the filtering criteria for the episode). If an episode is closed in Episode Review, this automatically breaks the episode.</td>
</tr>
<tr>
<td>Episode Information</td>
<td>Specify how you want episode information to appear in Episode Review. This information is at the episode level and is separate from the information for the underlying notable events in the episode. • Same as the first event names the episode using the fields from the first event in the episode. • Same as the last event continually updates the episode fields as each new event enters the episode. • Static value lets you provide a specific name. If you select Static value for Episode Title or Episode Description, you can use a token such as %title% or %description% to insert the value of a field.</td>
</tr>
</tbody>
</table>

**Action Rules**

Add action rules to take specific actions on the episode. Action rules are optional. You can define more than one rule for an episode.

**Steps**
1. Click + Add Rule.
2. Use the If pane to specify the trigger conditions for the action(s).
3. Use the Then pane to specify the action(s) to take if the trigger conditions are met.
4. If you select "Send email", "Run a script", "ServiceNow Incident Integration", "Remedy Incident Integration", or "VictorOps" as an action, click Configure to configure the action. For information on configuring an email, see Send an email in this manual. The options to integrate with ServiceNow, Remedy, or VictorOps are only available if you have installed and configured the corresponding Splunk add-on. For more information, see Create a ticket in ServiceNow or Remedy or Create a ticket in VictorOps.
5. (Optional) Click + and if or + and to add additional trigger conditions or actions. For example, if you want to close the episode and change the severity level to Info when a clearing event comes in, you could specify the following:

   If the following event occurs: severity matches Normal, change severity to Info for the episode, add a comment Don't worry for the episode, and change status to Closed for the episode.

   **Policy Info**

   Specify a title and description for your notable event aggregation policy. The policy is enabled by default and immediately takes effect. Disable it if you do not want it to take effect yet.

   Click Next and you will see the message that the policy has been successfully created. After the policy has been created, new incoming notable events are grouped in Episode Review according to the criteria in the policy. Note that notable events that existed before the policy was created are not grouped by the policy.

   **Enable Smart Mode on an aggregation policy**

   After creating and saving a custom aggregation policy, you can edit the policy to enable and configure Smart Mode. For information on enabling Smart Mode, see Enable Smart Mode in ITSI.

   **Set aggregation policy permissions**

   After you create a notable event aggregation policy, set read/write permissions for it to control who can modify or read it.
1. Click **Configure > Notable Event Aggregation Policies**.
2. Locate the policy and click **Edit > Edit Permissions**.
3. Assign read or write permissions to roles as desired.

   You cannot assign write permissions unless the role possesses the `write_itsi_notable_aggregation_policy` capability.
4. Click **Save**.

**Group similar events with Smart Mode in ITSI**

Smart Mode groups notable events in Episode Review based on their similarities. It uses machine learning algorithms to compare event field values and group events that are related to each other. Smart Mode reduces noise and detects patterns in your events so you don't have to. You can enable Smart Mode on the default aggregation policy or on a custom policy.

Do not enable Smart Mode on more than five aggregation policies in a minimum environment (12 CPUs, 12 GB of RAM on the search head), or more than 20 aggregation policies in a performance environment (48 CPUs, 65 GB of RAM on the search head).

1. Click **Configure > Notable Event Aggregation Policies**.
2. Select a custom policy or the **Default Policy**.
3. Under Smart Mode grouping, enable **Smart Mode**.
4. Click **Select fields**. A dialog displays the fields found in your notable events from the last 24 hours.
5. (Optional) Change the time period for the field analysis and click **Re-run Analysis**.
6. Choose the fields to compare for event similarity. Recommended fields are selected by default. Do not select more than 15 fields, as this might impact performance.

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Category or Text based on the content of the field. Category fields have a distinct value, such as a status field. Text fields consist of a string, such as a description field.</td>
</tr>
<tr>
<td># of Values</td>
<td>The number of values for each field.</td>
</tr>
<tr>
<td>Event Coverage</td>
<td>The percentage of events that contain the field. In general, choose fields with high event coverage.</td>
</tr>
</tbody>
</table>

7. Click **Apply**.

An episode preview uses the last 24 hours of data to illustrate event
grouping with Smart Mode enabled. Expand an episode to see the individual notable events.

8. (Optional) Configure other Smart Mode settings:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart Mode grouping</td>
<td>The importance of text similarity versus category similarity. The episodes in the preview update to reflect the importance factors you set. Setting both to 0, half, or 1 gives the factors equal weight.</td>
</tr>
<tr>
<td></td>
<td>Enable <strong>Split by Service</strong> to provide service context for your events. For example, if two events have similar fields but affect different services, they probably shouldn't be grouped together. If enabled, events are grouped by service first, then by text and category similarity.</td>
</tr>
<tr>
<td></td>
<td>Enable <strong>Split by Entity</strong> to provide entity context for your events. If enabled, ITSI segregates events based on the entity they belong to before applying grouping. Then it groups by text and category similarity. For example, if there are web status errors and disk errors on the same host that occurred in the same time period, those events are now in the same episode.</td>
</tr>
<tr>
<td></td>
<td>If you split by service AND entity, Smart Mode splits by service first (if the event has service association) and does not split further. If the event has no service association, it splits by entity.</td>
</tr>
<tr>
<td>Split events by field</td>
<td>Splits events into separate episodes based on a field name. <strong>host</strong> is a common field to split by so that separate episodes get created for each host. Use commas to separate multiple fields.</td>
</tr>
<tr>
<td>Break episode</td>
<td>The number of seconds that the flow of events is paused before grouping stops.</td>
</tr>
<tr>
<td>Episode information</td>
<td>Determines how the information in each episode is named or assigned. If you select <strong>Static value</strong> for Episode Title or Episode Description, you can use a token such as %title% or %description% to insert the value of a field.</td>
</tr>
</tbody>
</table>

9. Click **Save**.
After you save your aggregation policy, events are grouped in Episode Review according to the policy you configured. Custom aggregation policies take precedence over the Default Policy, so if an event meets the criteria of a custom policy, it is grouped according to that policy's rules.
Correlation Searches

Correlation search overview for ITSI

A correlation search is a recurring search that scans multiple data sources for defined patterns. You can configure a correlation search to generate a notable event (alert) when search results meet specific conditions. Use Episode Review to review notable events that your correlation searches generate and to initiate the investigative process of determining root cause.

You can use an ITSI correlation search to ingest third-party alerts as ITSI notable events. If you are creating a correlation search to ingest alerts from a third-party product, such as Nagios or SCOM, see Ingest third-party alerts as ITSI notable events for specific instructions.

Do not create correlation searches by manually editing $SPLUNK_HOME/etc/apps/itsi/local/savedsearches.conf. The search will not appear on the correlation search lister page. Always create correlation searches directly in the IT Service Intelligence app.

Predefined correlation searches

The following correlation searches are delivered with ITSI. By default, they are disabled. You can enable them and modify them to meet your needs.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitor Critical Service Based on Healthscore</td>
<td>Creates notable events for services with a critical health score.</td>
</tr>
<tr>
<td>SNMP Traps</td>
<td>Creates notable events for SNMP traps being ingested into ITSI. See Ingest SNMP traps in ITSI for more information.</td>
</tr>
<tr>
<td>Splunk App for Infrastructure Alerts</td>
<td>Creates notable events from Splunk App for Infrastructure alerts when integration is enabled between the Splunk App for Infrastructure and ITSI. See Ingest Splunk App for Infrastructure alerts into ITSI as notable events for more information.</td>
</tr>
<tr>
<td>Normalized Correlation Search</td>
<td>Creates notable events for any third-party alerts being ingested into ITSI that include ITSI normalized fields. See</td>
</tr>
</tbody>
</table>
Create correlation searches in ITSI

You can create your own correlation searches to generate notable events, throttle events, and perform other actions automatically based on a correlation in events.

Create a correlation search manually if you are an expert with SPL. You can review the included correlation searches for examples of the search methodology and available options. Test your correlation search ideas on the Search page before implementing them.

**Steps:**

1. From the ITSI main menu, click **Configure > Correlation Searches**.
2. Click **Create New Search > Create Correlation Search**.

You can also create a correlation search by cloning an existing one. In the Actions column on the correlation search lister page, click **Edit > Clone**.

For information on configuring correlation searches, see Configure correlation searches in ITSI.

Configure correlation searches in ITSI

Configure correlation searches to update the settings associated with how they run, change the search logic, and throttle alerts. See Correlation search overview for ITSI to learn more about correlation searches.

**Search Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search Name</td>
<td>A name that describes the correlation search. For example, &quot;cpu_load_percent&quot;.</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>
### Search Type

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search Type</td>
<td>(Optional) A description of the type of issue the search is intended to detect.</td>
<td></td>
</tr>
<tr>
<td>Search Type</td>
<td>The correlation search type:</td>
<td></td>
</tr>
<tr>
<td>♦ Data Model</td>
<td>Create a search based on a Splunk data model. Click Select to select a data model.</td>
<td>Ad hoc</td>
</tr>
<tr>
<td>♦ Ad hoc</td>
<td>Create a search based on a custom search string that you provide.</td>
<td></td>
</tr>
</tbody>
</table>

### Time range

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time range</td>
<td>The time range over which the correlation search applies.</td>
<td>Last 15 minutes</td>
</tr>
</tbody>
</table>

### Association

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service</td>
<td>Select one or more ITSI services to which this correlation search applies. You can only select services that belong to teams for which you have read access.</td>
<td>None</td>
</tr>
<tr>
<td>Entity Lookup Field</td>
<td>The field in the data retrieved by the correlation search that is used to look up corresponding entities. For example, host.</td>
<td>None</td>
</tr>
</tbody>
</table>
### Schedule

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Configure the schedule for the correlation search:</td>
<td>Basic, Every 5 minutes</td>
</tr>
<tr>
<td></td>
<td>♦ <strong>Basic</strong>: Schedule searches to run at regular intervals. Configure the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>search interval in the <strong>Run Every</strong> menu.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>♦ <strong>Cron</strong>: Schedule searches to run periodically at fixed times, dates, or</td>
<td></td>
</tr>
<tr>
<td></td>
<td>intervals. Enter a schedule in <strong>Cron Schedule</strong>. For more information, see</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Use cron expressions for scheduling in the <strong>Alerting Manual</strong></td>
<td></td>
</tr>
</tbody>
</table>

### Notable Events

Use this section to configure the notable event that is generated when search results meet a specific condition. The Splunk platform indexes the event object like any other event. You can track, manage, and update notable events in Episode Review.

ITSI correlation searches support field substitution with tokens in the format `%fieldname%`. Use field substitution to map third-party alert field values to corresponding notable event fields. See Use ITSI correlation search to ingest alerts for specific examples.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notable Event Title</td>
<td>The title of the notable event in Episode Review. For example, <code>mysql-01 server cpu Load %</code>.</td>
<td>None</td>
</tr>
<tr>
<td>Notable Event Description</td>
<td>A brief phrase to describe the notable event. For example, &quot;This alert triggers when DB CPU load on the mysql-01 server reaches 80%.&quot;</td>
<td>None</td>
</tr>
<tr>
<td>Owner</td>
<td>The ITSI role to which the notable event is assigned in Episode Review.</td>
<td>Unassigned</td>
</tr>
</tbody>
</table>
| Severity            | The level of importance of the event. Values must match an integer specified in the default version of `itsi_notable_event_severity.conf` (or the local version if you created one). | 1 - Info  
2 - Normal 
3 - Low 
4 - Medium 
5 - High 
6 - Critical |
| Status              | The triage status of the event in Episode Review. Values must match an integer specified in the default version of `itsi_notable_event_status.conf` (or the local version if you created one). | 0 - Unassigned  
1 - New 
2 - In Progress  
3 - Pending 
4 - Resolved 
5 - Closed |
| Drilldown Search Name | You can drill down to a specific Splunk search from the Overview tab of Episode Review for an individual notable event or from the Grouped Events tab for a group of notable events. Set the name of the drilldown search link. | None     |
| Drilldown Search    | The search you drill down to.                                                | None     |
|                     |                                                                              | Last 5 minutes |
Drilldown earliest offset
Defines how far back from the time of the event to start looking for related events.

Drilldown latest offset
Defines how far ahead from the time of the event to look for related events.

Notable Event Identifier Fields
Determine whether a notable event is unique or not. These identifier fields form the event hash field, which is added to every notable event to help identify unique alarm types.

Drilldown Website Name
You can drill down to a specific website from the Overview tab of Episode Review. Set the name of the drilldown website link.

Drilldown Website URL
The website you drill down to.

Advanced Options

Throttling

When correlation search results meet specific conditions, the search generates a new alert. This can create multiple alerts for the same condition. In most cases, it is best to have a single alert for the same condition. Throttling prevents the creation of multiple alerts for the same condition during a specified time range.

Throttling applies to any correlation search alert type, including notable events and actions (RSS feed, email, run script, and ticketing).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suppress Period</td>
<td>During the suppress period, any additional event that matches any of the Fields to group by does not create a new alert. After the suppress period passes, the next matching event creates a new alert, and throttling conditions resume. Enter a</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>relative time range in seconds. For example &quot;60s&quot; (60 seconds).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fields to group by Fields to compare for similar events. For example, cpu_load_percent.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>During the suppress period, any additional matches in the correlation search results are compared to the fields defined here. If a field matches, it stops the creation of a new alert. You can define multiple fields. The fields available depend on the search fields that the correlation search returns.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Actions**

Actions are other alert types that a correlation search can trigger. You configure action alerts independently from other alert types, such as Notable Events and Risk Scoring.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include in RSS feed</td>
<td>Posts the correlation search alert on the Splunk Enterprise RSS feed.</td>
</tr>
<tr>
<td>Send email</td>
<td>Sends an email about the correlation search alert.</td>
</tr>
<tr>
<td>♦ Email subject:</td>
<td>The email subject defaults to &quot;Splunk Alert: $name$&quot;, where $name$ is the correlation search Search Name.</td>
</tr>
<tr>
<td>♦ Email address(es):</td>
<td>Insert email addresses and/or distribution lists that should receive the alert.</td>
</tr>
<tr>
<td>♦ Include entity information:</td>
<td>Appends entity information to the subject of the email.</td>
</tr>
<tr>
<td>♦ Include results in email:</td>
<td>If you choose to include entity information, adds the entity information as an email</td>
</tr>
</tbody>
</table>
The schedule_search capability and the admin_all_objects capability are required for PDF delivery scheduling.

Note: Email actions require that you configure the mail server in Splunk Enterprise. See Configure email notification settings in the Alerting Manual.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run a script</td>
<td>Triggers a shell script. See Configure scripted alerts in the Alerting Manual.</td>
</tr>
</tbody>
</table>

Create multi-KPI alerts in ITSI

A multi-KPI alert is an alert based on trigger conditions that you define for multiple KPIs. When trigger conditions occur simultaneously for each KPI, a correlation search generates a notable event.

Multi-KPI alerts are useful for correlating the status of multiple KPIs across multiple services. This lets you identify causal relationships, investigate root cause, and provide insights into behaviors across your infrastructure.

For example, to avoid the negative impact that a large spike in traffic can have on website performance, you might create a multi-KPI alert based on two common KPIs: CPU load percent and web requests. A sudden simultaneous spike in both CPU load percent and web request KPIs might indicate a DDOS (Distributed Denial of Service) attack. Multi-KPI alerts can bring such trending behaviors to your attention early, so that you can take action to minimize any impact on performance.

Multi-KPI alerts can apply to one or more KPIs across one or more services.

Multi-KPI alert types

Splunk IT Service Intelligence (ITSI) provides two types of multi-KPI alerts:
Based on the severity level of one or more KPIs. If one or more trigger conditions for each KPI are satisfied simultaneously, ITSI triggers an alert and generates a notable event.

Based on a composite score of all KPIs added to the alert. Composite KPI scores are calculated from the KPI severity-level status and an importance value that you assign to the KPI when you create the alert.

Create a multi-KPI alert by selecting Multi-KPI Alerts from the top menu bar. You can also create one from a deep dive by selecting one or more KPI swim lanes and then selecting Bulk Actions > Create Multi KPI Alert.

**Status over time alerts**

Status over time alerts are multi-KPI alerts based on the severity level status of one or more KPIs. Status over time alerts require you to define one or more trigger conditions per selected KPI.

For example, you select two KPIs for your alert. The first KPI has a trigger for critical 15% of the time and a trigger for high 50% of the time. The second KPI has a single trigger for critical 25% of the time. The boolean expression would read:

\[(\text{kpi1\_perc\_crit} > 15\% \text{ OR kpi1\_perc\_high} > 50\%) \text{ AND (kpi2\_perc\_critical} > 25\%\)]

**Create a status over time alert**

1. On the Multi-KPI Alerts page, click Status over time.
2. In the Services column, select the service(s) that contain the KPIs for which you want to set up an alert. Note that only services that belong to teams for which you have read access are listed.
3. Optionally, select Depends on or Impacts to include KPIs from services that impact or depend on that service.
   All KPIs in each selected service appear in the "KPIs in Selected Services" section.
4. In the "KPIs in Selected Services" section, click +Add to add a KPI to the alert.
5. Set trigger conditions for the KPI alert. The condition statement dynamically updates based on the the trigger condition settings.
6. Click **Apply**. The selected KPI appears in the **Selected KPIs** section.
7. Add additional KPIs to the alert. Set one or more trigger conditions for each KPI.

   At least one trigger condition for each KPI must be satisfied simultaneously to trigger the alert and generate a notable event.
8. Click **Save**. The Create Correlation Search modal opens.
9. Configure the correlation search parameters to determine the schedule type, how often the search runs, and the severity of the notable event generated by the alert.
10. Click **Save**. ITSI saves the correlation search in **Settings > Searches, reports, and alerts**. For more information, see Create correlation searches in ITSI.

**Composite alerts**

Composite KPI alerts are based on a composite score of all KPIs added to the alert. Unlike service health scores, which give you a weighted average of all KPIs in a service, composite KPI scores give you the weighted average of the selected subset of KPIs only.

Composite KPI scores are calculated from the KPI severity-level status and an importance value that you assign to the KPI when you create the alert. This lets you create unique alerts for different groups of KPIs based on the relative importance of a KPI in the context of the alert.

The Importance value that you set for a KPI when you create a composite KPI alert does not change the Importance value that you set for the KPI (which is used to calculate the service health score) when you first create the KPI.
Create a composite KPI alert

1. On the Multi-KPI Alerts page, click Composite score.
2. In the Services column, select the service(s) that contain the KPIs for which you want to set up an alert. Note that only services that belong to teams for which you have read access are listed. The KPIs in each service appear in the "KPIs in Selected Services" section.
3. Click +Add to select KPIs that you want to include in the composite KPI alert. The selected KPIs appear in the "Selected KPIs" section.
4. Set an Importance factor for each KPI. The composite score for all selected KPIs dynamically updates based on the selected Importance factors.
5. Click Save. The Create Correlation Search dialog appears.
6. Configure correlation search parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Alert me if | Defines the suppression conditions for the alert.  
  ◊ **Score** - If the composite score value is less than the value you specify, the alert triggers.  
  ◊ **Status** - If the composite score severity level meets the condition you select, the alert triggers. |
| Suppression | Ignores notable events generated for the same alert condition. Enable suppression to minimize the number of duplicate notable events sent to Episode Review. |
| Trigger if | Generate notable events based on trigger conditions that occur simultaneously.  
  ◊ **Count** - If the number of occurrences of the alert condition meets or exceeds the value over time that you specify, the alert triggers.  
  ◊ **Consecutive Count** - If the number of consecutive occurrences of the alert condition meets or exceeds the value that you specify, the alert triggers. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

7. Click **Next** and specify how the alert should appear in Episode Review.
8. Click **Save**.

ITSI saves the correlation search in **Configure > Correlation Searches** and **Settings > searches, reports, and alerts**. For more information, see Create correlation searches in ITSI in this manual.

For an end-to-end troubleshooting scenario that involves creating a multi-KPI alert, see Troubleshoot an outage in ITSI in the *Splunk IT Service Intelligence Use Cases* manual.
Maintenance

Create maintenance windows in ITSI

Splunk IT Service Intelligence (ITSI) lets you put services and entities into maintenance mode for a specific time period. You can use maintenance mode to prevent ITSI from triggering alerts from machines and other devices that are undergoing maintenance operations or do not require active monitoring for any reason.

For example, if your ticketing system experiences a failure and you start to receive a cascade of identical notable events, you can put the service that is monitoring the ticketing system into maintenance mode to stop ITSI from generating alerts until the issue is resolved.

Create a new maintenance window

Maintenance windows apply to services and entities.

Prerequisites

By default, only users assigned the `itoa_admin` or `itoa_team_admin` role can create a maintenance window.

Steps

1. From the ITSI top menu bar, click **Configure > Maintenance Windows.**
2. Click **Create Maintenance Window.**
3. Provide a title for the maintenance window. For example, "DB entity maintenance window."
4. Set the start time, duration, and end time for the maintenance window.
5. Select the **Objects** for which you want to create a maintenance window: Entities or Services.

   If you do not have write access to the Global team, you will not be able to select Entities to put into maintenance.

6. Click **Next.**
7. Select the specific services or entities that you want to place in maintenance mode for the duration of the maintenance window. You can only select services or entities for which you have write access.
8. Click **Create**.
The selected entities or services enter maintenance mode according to the defined schedule.

When viewing a scheduled maintenance window, a user only sees the services included in the maintenance window for which the user has read access. If a maintenance window only contains services for which the user does not have read access, the user cannot view the maintenance window. A maintenance window that contains only entities can be viewed by all ITSI users.

If using the Delete bulk action to delete all or several maintenance windows, only the maintenance windows that contain services or entities for which the user has write access will be deleted.

**Impact of maintenance windows**

Maintenance windows can have an impact on associated KPIs, service health score calculations, and other ITSI features.

Consider the following when you put a service into maintenance mode:

- When you put a service into maintenance mode, all KPIs associated with that service are automatically put into maintenance mode.
- Search results from KPIs in maintenance mode are ignored for the purpose of service health score calculation for the duration of the maintenance window.

Consider the following when you put an entity into maintenance mode:

- If that entity has no KPIs running searches against it, there is no impact on service health scores.
- If that entity has one or more KPIs that are running searches against it, all search results from all KPIs running against that entity are ignored for the purpose of service health score calculation.
- If a KPI is split by entity (for example if the same KPI is running against two different entities, and one entity is in maintenance mode and one is not), search results generated by the KPI running against the entity in maintenance mode are ignored for the purpose of health score calculation. Search results generated by the same KPI running against the entity that is not in maintenance mode are included as usual in the service health score calculation.
 Entities can be placed in full or partial maintenance mode without being explicitly placed in maintenance mode, if a service that contains the entity is placed in maintenance mode.

**Impact on ITSI features**

Services, entities, and KPIs that are fully or partially impacted by a maintenance window appear in a dark gray color on pages that display health scores, including service analyzers, service and entity details pages, glass tables, multi-KPI alerts, and deep dives.

**View impacted KPIs**

You can view the impact of a maintenance window on associated KPIs.

1. Select a maintenance window from the Maintenance Windows lister page. The maintenance window details page opens, showing the specific services or entities impacted by the maintenance window.
2. Click **Impacted KPIs** to see a list of KPIs impacted by the maintenance window. KPIs that are split by entity, and thus are currently running searches against other entities that are not in maintenance mode, are listed as "Partially" impacted. KPIs that are not split by entity are listed as "Fully" impacted.
When to schedule maintenance windows

It is a best practice to schedule maintenance windows with a 15- to 30-minute time buffer before and after you start and stop your maintenance work. This gives the system an opportunity to catch up with the maintenance state and reduces the chances of ITSI generating false positives during maintenance operations.

For example, if a server will be shut down for maintenance at 1:00PM and restarted at 5:00PM, the ideal maintenance window is 12:30PM to 5:30PM.

The 15- to 30-minute time buffer is a rough estimate based on 15 minutes being the time period over which most KPIs are configured to search data and identify alert triggers.

Back up and restore ITSI KV store data

Back up the KV store and restore it from backup. You can perform both full backups and partial backups. Taking regular backups from a healthy environment enables you to restore from a backup in the event of a disaster, or if you add a search head to a cluster. Make sure to be familiar with the standard backup and restore tools and procedures used by your organization.

When you run a backup job, ITSI saves your data to a set of JSON files compressed into a single .zip file located in

$SPLUNK_HOME/var/itsi/backups/<_key.zip> on the search head. ITSI detects and preserves the application version from which it creates a backup. When you restore from a backup, ITSI detects the correct version of the backup and performs the required migration.

Splunk Cloud customers must use the Backup/Restore page in the ITSI user interface. Others can perform backups and restores from the command line using the kvstore_to_json.py script. See Backup and restore operations (mode 1) for information.

The following table describes the functionality available in each backup and restore method.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backup/Restore UI</th>
<th>Command line script</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full backup</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

221
<table>
<thead>
<tr>
<th>Method</th>
<th>Backup/Restore</th>
<th>Command line script</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partial backup</td>
<td>X</td>
<td>X</td>
<td>Dependent objects are not backed up when performing a partial backup using the command line script.</td>
</tr>
<tr>
<td>Partial restore</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Merge changes</td>
<td>X</td>
<td>X</td>
<td>Merges objects in the backup with existing KV store objects.</td>
</tr>
<tr>
<td>during restore</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clean restore</td>
<td></td>
<td>X</td>
<td>Replaces existing KV store objects with objects in the backup.</td>
</tr>
</tbody>
</table>

**What gets backed up in an ITSI backup**

The following table describes the types of data included and not included in an ITSI backup.

<table>
<thead>
<tr>
<th>Data</th>
<th>Example</th>
<th>Included in backup</th>
<th>Not included in backup</th>
</tr>
</thead>
<tbody>
<tr>
<td>KV store</td>
<td>Services, service templates, entities, KPIs, KPI base searches, teams,</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>objects</td>
<td>glass tables, service analyzers, deep dives</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configuration</td>
<td>deep_dive_drilldowns.conf, itsi_settings.conf</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>files</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indexed data</td>
<td>ITSI summary index, notable events</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

To back up indexed data, use the same approach you use to back up other Splunk indexes. For information, see Back up indexed data in the Splunk Enterprise *Managing Indexers and Clusters of Indexers* manual.

To preserve custom configurations contained in .conf files, make an archive or copy of `$SPLUNK_HOME/etc/`.
Default scheduled backup

The default scheduled backup is a full backup that runs daily at 1:00 AM (01:00) in the server’s local time zone. The time of the backup job is displayed on the Backup/Restore page in the Splunk user’s local time zone. Click Edit in the Actions column to change the frequency and time of the scheduled backup job or disable it from the Backup/Restore Jobs page. You cannot delete the scheduled backup job.

There is only one scheduled backup job. You cannot create additional scheduled backup jobs. Only the most recent backup is retained from the scheduled backup.

Create a full backup

Create a full backup to make a copy of all your ITSI configuration information.

Prerequisites

You must have the itoa_admin role, or be granted the write_itsi_backup_restore capability, to create a backup/restore job.

Steps

1. Click Configure > Backup/Restore.
2. Click Create Job > Create Backup Job.
3. Select Full Backup.
4. Provide a name and description of the backup job.
5. Click Create.

The backup job appears on the Backup/Restore Jobs lister page with the status "Queued" until the job runs. When the backup job finishes, the status changes to "Completed" and a confirmation message appears in the Messages drop-down in Splunk Web.

You can run any completed backup job again by clicking Edit > Start Backup in the Actions column. You can also modify the completed backup job before running it again.

Create a partial backup

Create a partial backup if you want to back up one or more KV store objects
instead of all KV store objects. You can back up services, service templates, teams, and glass tables. When selecting one of these object types, dependent objects are automatically selected to preserve the functionality of the objects after they are restored. In some cases, you can choose whether or not to include dependent objects in the backup.

Prerequisites

You must have the `itoa_admin` role, or be granted the `write_itsi_backup_restore` capability, to create a backup/restore job.

Steps

1. Click **Configure > Backup/Restore**.
2. Click **Create Job > Create Backup Job**.
3. Select **Partial Backup**.
4. Provide a name and description of the backup.
5. Click **Next**.
6. On the partial backup page, select the objects to include in the backup.
   - Selections made in one tab can cause selections to be made in another tab if there are dependencies between the objects.
7. (Optional) Click **Change Settings** to change the objects that are selected when you select a service. By default, dependent services are selected.
   - The KPI base searches, threshold templates, and team associated with a service are always included in the backup.
   - Although entities are not listed in the partial backup page, you can include them in the backup file by selecting **Entities** in the Settings dialog.
8. After making your selections, review every tab to confirm the objects that are selected.
9. Click **Save and Backup**.

The backup job appears in the Backup/Restore jobs lister page with the status "Queued" until the job runs. When the backup job finishes, the status changes to "Completed" and a confirmation message appears in the Messages drop-down in Splunk web.

You can edit any partial backup job before it starts. If the backup job started, you will see a read-only view that lists the objects contained in the partial backup.

You can run any completed backup job again by clicking **Edit > Start Backup** in the Actions column. You can also modify the completed backup job before running it again.
**Back up a service**

When you select a service to back up, ITSI also backs up the following objects:

- KPI base searches
- Threshold templates
- Teams

When you select a service to back up, you can choose whether or not to back up the following objects:

- Dependent services
- Entities that match service entity rules
- Linked service template

The dependency between the service and these objects breaks if you do not choose to back up the associated object:

- If you back up a service that has dependent services without also backing up the dependent services, the service will no longer be dependent on the services after restoring.
- If you back up a service without backing up the entities that match the service entity rules, the entities will no longer be associated with the service if the entities do not exist in the restored environment.
- If you back up a service that is linked to a service template without also backing up the service template, the service will no longer be linked to the service template after restoring.

**Back up a service template**

When you back up a service template, all the services linked to the service template are also added to the backup. If you deselect a service, it will not exist in the restored environment.

**Back up teams**

When you select to back up a team, all the services associated with that team are added to the backup. You can deselect any services you do not want to back up.
**Back up a glass table**

When you select to back up a glass table, all of the services associated with that glass table are also added to the backup. If you choose not to back up a service that the glass table depends on, any widgets that use KPIs from the service will no longer function if the service does not exist in the restored environment.

Glass table images and ACLs are always included in the backup when you back up a glass table.

**Restore a full or partial backup**

When you restore from a backup listed in the Backup/Restore Jobs page, ITSI merges the JSON data contained in the backup ZIP file with your existing KV store data. New objects added since the backup are added, existing objects that match an object in the backup are replaced, and all other existing objects are preserved. If you want to delete all existing KV store objects in an ITSI instance and replace them with the objects in the backup for a clean restore, use the command line script.

A restore from the Backup/Restore UI restores all of the data in the backup file. If you want to selectively restore files in a backup, use the command line script.

If you are restoring from a previous version of ITSI to version 3.0 or later, all services and service-related objects such as entities, KPI templates, KPI base searches, and KPI threshold templates are placed in the Global team. Backups and subsequent restores on ITSI version 3.0 or later retain team information for services and service-related objects. See ITSI service-level permissions for information on teams.

When restoring a backup taken on an ITSI 3.0 or later system to another ITSI 3.0 or later system, team ACLs are retained when the teams are restored. Therefore, the roles assigned to the teams must exist on the system that the backup is restored to. For example, a restore creates teams called "HR" and "Finance", which have read/write access for the hr_admin and finance_admin roles, respectively. If the current system does not have these roles, these teams are only accessible to the itoa_admin role. If the roles assigned to the teams don't already exist on the system, you can create them either before or after restoring.

Before restoring a backup, make sure no service templates are syncing. Check the sync status of service templates in the Service Template viewer by selecting Configure > Service Templates from the top menu.
If you restart Splunk software while a backup or restore job is in progress, the job
resumes after Splunk software restarts. Queued jobs automatically time out if
they are not completed within 12 hours for any reason. You can change the
default timeout duration by updating the value of job_queue_timeout in the
[backup_restore] stanza in itsi_settings_conf.

**Restore from a backup**

You can restore from a backup that you created.

1. On the Backup/Restore Jobs lister page, find the backup from which you
   want to restore. Click **Edit > Restore Backup**.
2. Click **Start Restore**.
   "Restore from" prepends the backup name in the jobs list. A message
   stating that the restore job completed successfully appears in the
   Message drop-down in Splunk Web.

If you restore from a backup that was generated while the Default notable event
aggregation policy was running in Smart Mode, you must restart the Splunk
platform.

**Download and restore from a backup ZIP file**

You can download any backup ZIP file that is created when you run a backup job
in the UI and then restore from that backup ZIP file using the UI. The maximum
file size supported for uploading is 500 MB.

To download a backup ZIP file:

1. On the Backup/Restore jobs lister page, find the backup file that you want
to download.
2. Click **Edit > Download Backup**. The backup ZIP file downloads to your
   local machine.

To restore from a downloaded backup ZIP file:

1. Click **Create Job > Create Restore Job**.
2. Provide a name and optional description of the backup.
3. Click **Choose File** and select the previously downloaded backup ZIP file
   from which you want to restore.
4. Click **Create**.
   ITS1 uploads the backup ZIP file and the new restore job appears in the
   Backup/Restore Jobs list. A message stating that the restore job has
completed successfully appears in the Message drop-down in Splunk web.

**Restore from a backup created using the command line**

If you created a backup of ITSI using the `kvstore_to_json.py` command line option and you want to restore that data using the Backup/Restore Jobs page, the backup JSON files must be contained in a folder named `backup` and compressed into a ZIP file.

**Backup and restore in a search head cluster environment**

You can run backup and restore jobs from the Backup/Restore page in search head cluster environments. You can create a backup on any cluster member and then later restore data from that backup on any cluster member, regardless of where the backup was initiated.

For example, if your search head cluster has three cluster members, sh-01, sh-02, and sh-03, and you create a backup on sh-01, you can later restore from that backup on sh-01, sh-02, or sh-03.

When you create a backup on any search head cluster member the configuration data from all cluster members is backed up. Likewise, when you restore from a backup on any cluster member, configuration data is restored across all cluster members.

In a search head cluster environment, the scheduled backup runs only on the captain. However, you can perform a restore of a scheduled backup from any cluster member. If you choose to download the scheduled backup, make sure to download it from the captain because the captain contains the latest backup.

**kvstore_to_json.py operations in ITSI**

ITSI provides a `kvstore_to_json.py` script that lets you backup/restore ITSI configuration data, perform bulk service KPI operations, apply time zone offsets for ITSI objects, and regenerate KPI search schedules.

**Usage options**

The `kvstore_to_json.py` script is located in

```
$SPLUNK_HOME/etc/apps/SA-ITOA/bin/
```
The `kvstore_to_json.py` script has these 4 modes:

- **Mode 1:** Backup and restore operations
- **Mode 2:** Bulk service KPI operations.
- **Mode 3:** Time zone offset operations.
- **Mode 4:** Regenerate KPI search schedules.

To view all `kvstore_to_json.py` usage options, specify the `-h` option.

```
[root@myserver splunk]# ./bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -h
```

Usage: kvstore_to_json.py [options]

Options:
- `-h`, `--help` show this help message and exit
- `-s` `SPLUNKDPORT`, `--splunkdport=SPLUNKDPORT` splunkd port. If no option is provided, we will default to '8089'
- `-u` `USERNAME`, `--username=USERNAME` Splunk username
- `-p` `PASSWORD`, `--password=PASSWORD` Splunk password
- `-n`, `--no-prompt` Use this option when you want to disable the prompt
- `-v`, `--verbose` Use this option for verbose logging
- `-f` `FILE_PATH`, `--filepath=FILE_PATH` The full path of a directory. Usage depends on mode.
  - When importing backed up data of version 1.2.0, this could be a file or a set of files. When working with service KPIs, this is a directory containing input.json on entry and output.json on exit.
- `-m` `MODE`, `--mode=MODE` Specify the mode of operation – what kind of operations to perform. Mode is set to:
  - 1 - for backup/restore operations.
  - 2 - for service KPI operations.
- `-i`, `--importData` Use this option when you want to perform backup/restore operations.
- `-f` `FILE_PATH`, `--filepath=FILE_PATH` The full path of a directory. Usage depends on mode.

Backup and restore operations. This is mode 1.:
Use this option when you want to perform backup/restore operations.

Use this option when you want to upload data to the KV
you can use filepath as wildcard to upload data from more than one file. However, filepath must be within quotes if it is being used as a wildcard.

- **--persist-data**
  Use this option when you want to persist existing configuration in KV Store during import. NOTE: Applicable only if importData option is used.

- **--dry-run**
  Use this option when you want only to list objects for import or backup.

- **--base-version=BR_VERSION**
  The original ITSI application version user intends to backup/restore from.

- **--rule_file_path=RULE_FILE_PATH**
  The full path of a file which has rules defined for backing up data or importing data.

- **--dupname-tag=DUPNAME_TAG**
  Automatically rename all the duplicated service or entity names from restoring with a tag. If this option is not set, the restoring will halt if duplicate names are detected. The default tag is: _dup_from_restore_<epoch_timestamp>

**Service KPI operations. This is mode 2.**

Use this option when you want to get/create/update/delete KPIs for existing services.

- **--get**
  For input, specify a list of service keys with the keys of KPIs to retrieve. Expected format: [[_key: <service key>, kpis: [[_key: <KPI key>]]]. Specify [] to get all KPIs from all services. Specify [[]] to get all KPIs from a service. Assumes input is available in file_path/input.json

- **--create**
  For input, specify a non-empty list of service keys with their KPIs list. Expected format: [[_key: <service key>, kpis: [[_key: <KPI key>, <rest of KPI]]]...
Structure>}}. Note that only existing services could be updated with new KPIs only with this option. Assumes input is available in

```
file_path/input.json
```

-\(t\), --update

For input, specify a non-empty list of service keys with their KPIs list. Expected format: 

```
[{_key: <service key>, kpis: [{_key: <KPI key>, <rest structure>}}]}
```

Note that only existing services and existing KPIs could be updated using this option. Assumes input is available in

```
file_path/input.json
```

-\(r\), --delete

For input, specify a list of service keys with the keys for the KPIs to delete. Expected format:

```
[{_key: <service key>, kpis: [{_key: <KPI key>}]}
```

Assumes input is available in file_path/input.json

**Timezone offset operations. This is mode 3.**

Use this option when you want to adjust timezone settings for time sensitive fields on object configuration.

-\(q\) IS_GET, --is_get=IS_GET

For input, specify if you are trying to read objects or update their timezone offsets.

-\(o\) OBJECT_TYPE, --object_type=OBJECT_TYPE

For input, specify a valid object type that contains time sensitive configuration. This option will apply offset to all objects on this type unless scoped to a specific object using object_key parameter. Supported object types are: "maintenance_calendar" for maintenance windows, "service" for Services/KPIs (threshold policies)

-\(k\) OBJECT_TITLE, --object_title=OBJECT_TITLE

For input, specify an optional object title of object type that contains time sensitive configuration. Using this option will cause the offset change to
to that object.

- \( z \) OFFSET\_IN\_SEC, \(--\text{offset\_seconds=OFFSET\_IN\_SEC} \)
For input, specify the offset to apply in seconds as a positive or negative number. This offset should be the number of seconds that you want to add or subtract from the current value.

Backup and restore operations (mode 1)

Use \texttt{kvstore\_to\_json.py} mode 1 to back up and restore your ITSI configuration data to and from JSON files. You can use the \texttt{kvstore\_to\_json} script to do full or partial backups, full or partial restores, and merged or clean restores.

See Backup and restore ITSI configuration data for a summary of the types of data included and not included in an ITSI backup.

You can run the \texttt{kvstore\_to\_json} script in either interactive mode or non-interactive mode. In interactive mode, which is the default, the script prompts you to enter required information. To run the script in non-interactive mode, you must enter the \texttt{-n} option to disable the default interactive mode. You can schedule backups through cron by calling the script with all necessary flags for full automation.

Before performing a backup or restore through the CLI, make sure that there are no service template syncs in progress. Check the sync status of service templates in the Service Template viewer by selecting \textbf{Configure > Service Templates} from the top menu.

Perform a full backup

\textbf{Perform a full backup using interactive mode}

1. Run \texttt{kvstore\_to\_json.py}, where \(-s\) is the default \texttt{splunkd} port. If you are running \texttt{splunkd} on any other port besides 8089, you must specify the port. For example:

\begin{verbatim}
cd $SPLUNK\_HOME
bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -s 8089 -u admin -p changeme -m 1 -f /root/backup
\end{verbatim}

2. Enter the requested information at the prompts. For example:

\begin{verbatim}
>> Enter the splunkd port number OR press the enter key to use [8089]
\end{verbatim}
Do you wish to backup data from KV Store OR restore to KV store.
Press [y|yes|enter] to backup, [n|no] to restore? > y
You have indicated backup of data from current ITSI version 4.0.3. Proceeding.

The script copies your KV store data to a series of JSON files in the root/backup directory. If the root/backup directory does not exist, the script creates it.

3. Go to root/backup to confirm that your backup JSON files have been created. Backup file names follow the specific format: `<COLLECTION_NAME>___<OBJECT_TYPE>___<ROLLOVER_NUMBER>`. For example:

* itsi_migration___migration___0.json
* itsi_pages___glass_table___0.json
* itsi_services___entity___0.json
* itsi_services___kpi_threshold_template___0.json
* itsi_pages___deep_dive___0.json
* itsi_service_analyzer___home_view___0.json
* itsi_services___kpi_template___0.json
* itsi_services___service___0.json
* itsi_services___correlation_search___0.json

Do not change the JSON file name. If you change the file name you cannot restore the data.

**Perform a full backup using non-interactive mode**

1. Run `kvstore_to_json.py`, using the appropriate backup options. Enter `-n` to disable the default interactive mode. Use the `-b` option to specify the ITSI version that you want to back up. If you do not specify `-b`, the current ITSI version is used. For example:

```bash
cd $SPLUNK_HOME
bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -s 8089 -u admin -p changeme -m 1 -f /root/backup -n -b "4.0.3"
```

2. Go to root/backup and confirm that your backup JSON files have been created.

**Restore from a backup**

ITSI supports restore migration from the current backup or an earlier version of the backup. You can migrate backup JSON files based on an earlier ITSI version to the most recent ITSI version.

You can restore all the files in a backup or selected files in a backup.
When restoring a backup taken on an ITSI 3.0 or later system to another ITSI 3.0 or later system, team ACLs are retained when the teams are restored. Therefore, the roles assigned to the teams must exist on the system the backup is restored to. For example, a restore creates teams called HR and Finance which have read/write access for the roles hr_admin and finance_admin, respectively. If the current system does not have the hr_admin and finance_admin roles, these teams will only be accessible to the itoa_admin role. If the roles assigned to the teams don't already exist on the system, they can be created either prior to restoring or after restoring.

**Restore from a backup using interactive mode**

To restore all files in a backup using interactive mode, do the following:

1. Run `kvstore_to_json.py`, using the `-i --importData` option to upload data from your JSON backup files to the KV store. For example:

   ```
   cd $SPLUNK_HOME
   bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -s 8089 -u admin -p changeme -m 1 -f /root/backup -i
   ```

2. Enter the requested information at the prompt. For example:

   ```
   >> Do you wish to backup data from KV Store OR restore to KV store.
   Press [y|yes|enter] to backup, [n|no] to restore? > n
   You would like to restore data from disk to KV Store. Proceeding.
   >> Enter the previous ITSI version from which you want the data to be restored. Ex: '2.2.0', '2.2.1', '2.2.2' ? >
   >> Do you wish to persist the existing data in KV Store during the import [y|n]: > y
   ```

   **Caution:** The `-i` option deletes all existing data from the KV store. If you do not want to delete all data from the KV store on import, enter `y` at the prompt to use `-d --persist data` option. This option appends data to the existing data in the KV store.

3. Open the ITSI app and verify that your configuration data has been restored.

**Restore from a backup using non-interactive mode**

To restore data using non-interactive mode, you must use the `-b` option to specify the ITSI version of the backup file you plan to restore.

To restore all files in a backup using non-interactive mode, do the following:
1. Run `kvstore_to_json.py`, using the `-i --importData` option to upload data from your JSON backup files to the KV store. Use the `-b` option to specify the ITSI version of the backup file. Make sure to include the `-n` option to disable interactive mode. For example:

   ```
   cd $SPLUNK_HOME
   bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -s 8089 -u admin -p changeme -m 1 -f /root/backup -i -n -b "4.0.3" -d
   ```

   The `-i` option deletes all data from the KV store. If you do not want to delete all data from the KV store on import, enter the `-d --persist data` option. This option appends data to the existing data in the KV store.

2. Open the ITSI app and verify that your configuration data has been restored.

    **Use a wildcard to specify groups of files**

    You can use a wildcard with the `-f` option to specify a group of files. For example, to import the files `itsi_export.json` `itsi_export_0.json` `itsi_export_1.json`

    you could enter the `-f` path as follows:

    ```
    cd $SPLUNK_HOME
    bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -s 8089 -u admin -p changeme -m 1 -f "/root/itsi_export*" -i
    ```

    The value of `-f` must be in quotes.

    **Perform a partial backup or restore**

    You can use the `kvstore_to_json.py` script to perform a partial backup or restore on a subset of your ITSI configuration data.

    Use the Backup/Restore page in the ITSI app to do partial backups of supported objects because dependent objects are also backed up using this method. Dependent objects are not backed up when doing a partial backup using the script which can result in a failed restore. For information on using the Backup/Restore page to do partial backups, see About partial backups.

    To perform a partial backup or restore of ITSI data, you must first create a JSON file that defines rules for the data restore. The rules file has the following schema:
For example, to restore a single object, such as a service:

1. In any directory, create a JSON rules file. For example:

   touch rules.json

2. Edit rules.json to define rules for importing the object. For example:

   [  
     {  
       "object_type": "service",  
       "title_list": "^Database Service$"  
     }  
   ]

**Perform a partial backup**

Performing a partial backup using the script does not back up dependent objects which can result in a failed restore. Perform partial backups of supported objects from the Backup/Restore page in order to preserve object dependencies.

To back up a single object, such as a service:

1. In any directory, create a JSON file to use as your rules file. For example:

   touch rules.json

236
2. Edit the rules.json file as shown in the schema to define a rule for backing up data. For example, to backup a single service called "Database Service" in rules.json you might specify:

```json
[
  {
    "object_type": "service",
    "title_list": "^Database Service$"
  }
]
```

3. Run the kvstore_to_json.py command using the dry run (-y) option.

```bash
./bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -u admin -p changeme -f /root/backup_folder_2 -y -l /root/partial_back_up/rules.json -n
```

4. Verify that the list of objects to backup is correct, before you perform the actual backup. For example:

```plaintext
>>> This is a dry run. No actual backup or importing will happen, only a list of objects will be displayed. To perform the actual operation, re-run again without the flag for dry run ...

<<<<<<<<<<<<<<<<<<< Object Type = service >>>>>>>>>>>>>>>>>>>>>>>>
Title = Database Service (_key = 97dc7a6a-5706-4d67-864b-b0e7a72e326c)

<<<<<<<<<<<<<<<<<<< Object Type = entity >>>>>>>>>>>>>>>>>>>>>>>>

<<<<<<<<<<<<<<<<<<< Object Type = kpi_template >>>>>>>>>>>>>>>>>>>>>>>>

<<<<<<<<<<<<<<<<<<< Object Type = kpi_threshold_template >>>>>>>>>>>>>>>>>>>>>>>>

<<<<<<<<<<<<<<<<<<< Object Type = glass_table >>>>>>>>>>>>>>>>>>>>>>>>

<<<<<<<<<<<<<<<<<<< Object Type = deep_dive >>>>>>>>>>>>>>>>>>>>>>>>

<<<<<<<<<<<<<<<<<<< Object Type = home_view >>>>>>>>>>>>>>>>>>>>>>>>

<<<<<<<<<<<<<<<<<<< Object Type = migration >>>>>>>>>>>>>>>>>>>>>>>>

5. Run the command without the dry run (-y) option to perform the actual backup.

**Perform a partial backup of a shared deep dive or service analyzer**

If performing a partial backup of a shared deep dive or service analyzer (home view) using the method described in the previous section, the deep dive or service analyzer may not be restored due to the fact that the ACL JSON file is not backed up and restored along with the shared object. To work around this issue, change the permissions of the object to Private before backing it up. Then change the permissions on the object back to Shared in App after restoring it.
Perform a partial backup of a glass table

Use the Backup/Restore UI to back up glass tables. When you do a partial backup of a glass table from the Backup/Restore UI, the glass table images, ACLs, and associated services are backed up with the glass table. See Backing up glass tables.

When you perform a partial backup of a glass table using the script, images in the glass table may not display after it has been restored. This is because the JSON file containing external images for the glass table needs to be backed up and restored along with the glass table. If it is a shared glass table, it may not be restored due to the fact that the ACL JSON file is not backed up and restored along with the shared object.

To use the script to back up the images that go with the glass table, as well as the ACLs necessary to restore a shared glass table, do the following:

1. Create a full backup without using rules.json.
2. Create a partial backup of the glass table using the rules.json file.
3. Create a backup_partial directory with the following files:
   - app_acl___app_acl___0.json
     From the full backup in step 1. This file contains the information needed to restore any glass tables that have "Shared in App" permissions.
   - SA-ITOA_files___glass_table_images___0.json
     From the full backup in step 1. This file contains the glass table images.
   - itsi_pages___glass_table___0.json
     From the partial backup of the glass table in step 2.
   - app_info.json
     From either the full or partial backup in steps 1 or 2.
4. Perform a full restore using the backup directory created in step 3. For example:
   cd $SPLUNK_HOME
   bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -s 8089 -u admin -p changeme -m 1 -f /root/backup_partial -i
   The glass table will be restored with its images.

Perform a partial restore

Use partial restore when you have a full backup of your ITSI configuration data and you want to restore only a few objects, instead of all objects.
For example, to restore a single object, such as a service:

1. In any directory, create a JSON rules file. For example:
   ```
touch rules.json
   ```

2. Edit `rules.json` to define rules for importing the object. For example:
   ```
   {
     "object_type": "service",
     "title_list": "Database Service$"
   }
   ```

3. Run the `kvstore_to_json.py` command using the dry run (-y) option, where `-f` is the path to the folder containing your full data backup, `-l` is the path to the `rules.json` file, `-i` is the option to import data, and `-d` is the option to persist data (to avoid deletion of existing objects). For example:
   ```
   ./bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py
   -u admin -p changeme -f /root/full_backup_folder -l
   /root/partial_back_up/rules.json -i -d -y -n
   ```

4. After you verify that the list of objects to import in the dry run is correct, run the command without the dry run (-y) option to perform the actual restore operation.

**Use replacement options**

The partial restore rules schema provides replacement options, which let you change the name of an object when you run a partial backup/restore operation. Replacement options are useful for renaming objects when moving from a test environment to a production environment.

For example, to backup a service called `test_database_service`, but change the name to `database_service`; and to backup a deep dive called `test_database_deep_dive`, and change the name to `database_deep_dive`, you would create a `rules.json` file that contains the following:

```
[
  {
    "object_type": "service",
    "title_list": "test_database_service$",
    "replacement_rules": [{
      ?replacement_key": "title",
      ?replacement_type": "replace",
      ?replacement_string": "database_service",
      ?replacement_pattern": "^test_database_service$"
    }]
  },
  {
    ...
  }
]
```
Service KPI operations (mode 2)

`kvstore_to_json.py` mode 2 options let you run bulk operations on KPIs, including get (-g), create (-c), update (-t), and delete (-r). Use these options to replicate, edit, and copy KPIs to multiple services, for example, when moving your ITSI deployment from a test environment to a production environment.

All service KPI options require you to specify the mode 2 parameter `--m 2`. You must also specify the file path `--f` parameter as the full path to the directory containing the `input.json` file.

Before you can run service KPI operations, you must create an `input.json` file in the destination directory. The script accepts data input from `input.json` and sends data output to an `output.json` file that the script creates in the same directory.

All service KPI operations, except get `-g`, require you to specify service and/or KPI keys. You can retrieve these keys using the `-g` option in `output.json`.

**Note:** See the `kvstore_to_json.py help -h` option for proper `input.json` and command syntax.

**Get service and KPI keys**

Use the get `-g` option to retrieve service and KPI data in JSON format, including service and KPI keys.

1. Create an `input.json` file in the destination directory.

   ```
   mkdir <directory_containing_input.json>
   touch input.json
   ```

240
2. **Edit input.json**: Add `[]` to the file to retrieve JSON data for all services and KPIs, or add specific service and kpi keys to the file to retrieve JSON data for those services and KPIs only. For example:

```json
[{{?_key?: ?<service_key>?, ?kpis?: [{?_key?: ?<kpi_key>?}] }} ]
```

3. **Run the kvstore_to_json script** using the `get -g` option. Name the full path to the directory containing the `input.json` file as the file path `-f` parameter. For example:

```
cd $SPLUNK_HOME
bin/splunk cmd python kvstore_to_json.py -u admin -p changeme -m 2
-g -f <directory_containing_input.json> -n
```

4. **Review the contents of output.json** to identify service and KPI keys. For example:

```
[
  {
    "_key": "669c5cec-a492-419d-8659-95a185b4dc5c",
    "kpis": [
      {
        "_key": "f017cc7b2e67f2b3b9152146",
        ...}
    ]
  }
]
```

### Create KPIs

**Use the `-c` option to create new KPIs.**

1. **Edit input.json** to specify the service key of the service for which you want to create the KPI.

2. **Add KPI keys** for the KPIs that you want to add to the service and any key-value pairs belonging to the KPI that you want to include in the KPI definition. Leave the key field for each KPI empty for ITSI to auto generate it. For example:

```
[
  {
    "_key": "<service_key>",
    "kpis": [
      {
        "title": "<title_of_kpi_to_create>",
        ...
      }
    ]
  }
]
```
Update KPIs

Use the `-t` option to update KPIs.

In `input.json` specify the service and KPI key for each KPI, and any other key/value pair data that you want to update for the KPI.

Delete KPIs

Use the `-r` option to delete KPIs.

In `input.json`, specify service and kpi keys for all KPIs that you want to delete.

Caution: Make sure to properly validate your JSON input. While the `kvstore_to_json` script does provide some schema validation, incorrect JSON formatting can cause errors.

Time zone offset operations (mode 3)

The `kvstore_to_json.py` mode 3 option lets you apply a time offset for time-sensitive fields in object configurations. You can use this option to correct time zone discrepancies for the following object types:

- `maintenance_calendar`: Sets an offset for maintenance window start and end times.
- `service`: Sets an offset for the KPI threshold time policies within a service. See Apply time zone offsets in this manual.
- `kpi_threshold_template`: Sets an offset for a KPI threshold template. After running the command to set an offset for a KPI threshold template, you must run the command again for each service that uses the KPI threshold template and set the same offset so that they are in sync.

Apply time zone offset

Run the following command to set an offset for one of the supported object types.

If you use the command to set an offset for a KPI threshold template, you must run the command again for each service that uses the KPI threshold template and set the same offset so that they are in sync.
1. Run `kvstore_to_json.py`, where `-o` is the object type, `-k` is the title of the specific object, and `-z` is the specific time zone offset in seconds. For example:

```
cd $SPLUNK_HOME
bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -m 3 -o service -k "Database Service" -z 1800
```

2. Enter the requested information at the prompts (default interactive mode only). For example:

```
>> Enter the splunkd port number OR press the enter key to use [8089] > 8089
>> Enter splunk username or press enter to use "admin" > admin
>> Enter splunk password for "admin" >
```

3. The script applies the time zone offset to the specified object. For example:

```
1 object(s) match request
Applying timezone change on requested object(s): [u'Database Service']
Timezone offset has been applied on the objects requested.
```

ITSI time-sensitive configurations are normalized to UTC.

**Regenerate KPI search schedules (mode 4)**

The `kvstore_to_json.py` mode 4 option regenerates the search schedules for your KPIs. Use this command if you have set your KPI saved search schedules to run at the same time in itsi_settings.conf. Run this command to reset the search schedules of all your KPIs to use the new search schedule. See Schedule KPI searches to update at the same time for more information.

1. Run `kvstore_to_json.py` in mode 4.
   For example:

   ```
cd $SPLUNK_HOME
bin/splunk cmd python etc/apps/SA-ITOA/bin/kvstore_to_json.py -m 4
```

2. Enter the requested information at the prompts (default interactive mode only).

3. You will see the following message after the KPI search schedules have been reset:
   Retrieving KPIs to reset their saved search scheduling
   Saving updated KPI scheduling
   Done.
Administer Splunk ITSI with configuration files

About configuration files

Splunk IT Service Intelligence configuration information is stored in configuration files. These files are identified by the .conf extension and hold the information for different aspects of your ITSI configurations. These aspects include:

- System settings
- Authentication and authorization information
- KPI, glass table, and deep dive configurations
- Notable event configurations
- Module settings

A single Splunk instance typically has multiple versions of configuration files across several directories. You can have configuration files with the same name in your default, local, and app directories. This creates a layering effect that allows Splunk to determine configuration priorities based on factors such as the current user and the current app.

For a list of ITSI configuration files and an overview of the area each file covers, see List of ITSI configuration files in this manual.

Most configuration files come packaged with your ITSI software in the $SPLUNK_HOME/etc/apps/ directory.

Editing a configuration file

You can edit the configuration files directly for any ITSI setting. Never change, copy, or move the configuration files in the default directory. Default files must remain intact and in their original location. To change settings for a particular configuration file, you must first create a new version of the file in a non-default directory and then add the settings that you want to change. When you first create this new version of the file, start with an empty file. Do not start from a copy of the file in the default directory.

Before you change any configuration files:
• Learn about how the default configuration files work, and where to put the files that you edit. See Configuration file directories.
• Learn about the structure of the stanzas that comprise configuration files and how the attributes you want to edit are set up. See Configuration file structure.
• Learn how different versions of the same configuration files in different directories are layered and combined. See Configuration file precedence.
• Consult the .spec and .example files for the configuration file. These files reside in the file system in $SPLUNK_HOME/etc/apps/SA-ITOA/README or $SPLUNK_HOME/etc/apps/itsi/README

After you are familiar with the configuration file content and directory structure, and understand how to leverage configuration file precedence, see How to edit a configuration file to learn how to safely modify your files.

**List of ITSI configuration files**

The following is a list of ITSI configuration files. All files are located under $SPLUNK_HOME/etc/apps/. Most .conf files have accompanying spec and example files located in the README folder that list all supporting attributes. Contact Support before editing a conf file that does not have an accompanying spec or example file.

If you are using Splunk Cloud, you can't edit a .conf file directly. For any task that requires editing a .conf file, submit a ticket using the Support Portal and Splunk Support will work with you to arrange a maintenance window.

**Caution**: Never change or copy the configuration files in the default directory. Default files must remain intact and in their original location. The upgrade process overwrites the default directory, so any changes that you make in the default directory are lost on upgrade. Create and edit your files in a local directory, for example $SPLUNK_HOME/etc/apps/<app_name>/local. Local directories are not overwritten during upgrades. For more information, see Configuration file directories.

<table>
<thead>
<tr>
<th>File</th>
<th>Purpose</th>
<th>ITSI Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert_actions.conf</td>
<td>Generate ITSI notable events and configure episode actions.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>File</td>
<td>Purpose</td>
<td>ITSI Location</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>alert_actions.conf</td>
<td>Summarizing KPI searches into the ITSI summary index.</td>
<td>/itsi/default</td>
</tr>
<tr>
<td>app_common_flags.conf</td>
<td>Used to enable or disable certain ITSI features. CAUTION: Do not edit</td>
<td>/itsi/default</td>
</tr>
<tr>
<td>authorize.conf</td>
<td>Configure ITSI-specific roles and capabilities, including role-based</td>
<td>/itsi/default</td>
</tr>
<tr>
<td></td>
<td>access controls. Always use /itsi/default. For more information, see</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Enable/disable ITSI capabilities for a role.</td>
<td></td>
</tr>
<tr>
<td>collections.conf</td>
<td>Configure KV store collections for ITSI.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>commands.conf</td>
<td>Connect search commands to any custom search script.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>datamodels.conf</td>
<td>Attribute/value pairs for configuring data models.</td>
<td>/DA-ITSI-APPSERVER/default</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/DA-ITSI-LB/default</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/DA-ITSI-VIRTUALIZATION/default</td>
</tr>
<tr>
<td>deep_dive_drilldowns.conf</td>
<td>Configure deep dive drilldowns, add new drilldowns.</td>
<td>/itsi/default</td>
</tr>
<tr>
<td>File</td>
<td>Purpose</td>
<td>ITSI Location</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>distsearch.conf</td>
<td>Specify behavior for distributed search. Group search peers to facilitate searching on a subset of peers.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>drilldownsearch_offset.conf</td>
<td>Configure time range picker presets for correlation search drilldown offsets.</td>
<td>/itsi/default</td>
</tr>
<tr>
<td>fields.conf</td>
<td>Create multivalue fields and add search capability for indexed fields.</td>
<td>/itsi/default</td>
</tr>
<tr>
<td>glasstable_icon_library.conf</td>
<td>Add and remove icons from the glass table icon library.</td>
<td>/itsi/default</td>
</tr>
<tr>
<td>inputs.conf</td>
<td>Set up data inputs.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_da.conf</td>
<td>(Deprecated) Configure an app to export entity searches and service templates for use within ITSI.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_deep_dive.conf</td>
<td>Upload deep dives to the KV store.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_glass_table.conf</td>
<td>Upload glass tables to the KV store.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>File</td>
<td>Purpose</td>
<td>ITSI Location</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>itsi_kpi_base_search.conf</td>
<td>Upload KPI base searches to the KV store.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_kpi_template.conf</td>
<td>Upload KPI templates to the KV store.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_kpi_threshold_template.conf</td>
<td>Upload KPI threshold templates to the KV store.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_notable_event_retention.conf</td>
<td>Define how long notable events are retained before they move to the index. Default is</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>File</td>
<td>Purpose</td>
<td>ITSI Location</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>itsi_notable_event_severity.conf</td>
<td>Configure the colors associated with different severity levels in Episode Review.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_notable_event_status.conf</td>
<td>Configure label descriptions and event status in Episode Review.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_service_analyzer.conf</td>
<td>Configure auto-refresh interval, or disable auto-refresh.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_service_template.conf</td>
<td>Configure an app to export service templates for use within ITSI.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_settings.conf</td>
<td>Configure ITSI.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>itsi_team.conf</td>
<td>Upload sample ITSI teams to the KV store.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>limits.conf</td>
<td>Set various limits (such as maximum result size or concurrent real-time searches) for search commands.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>macros.conf</td>
<td></td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>File</td>
<td>Purpose</td>
<td>ITSI Location</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>mad.conf</td>
<td>Define search macros in Settings.</td>
<td></td>
</tr>
<tr>
<td>notable_event_actions.conf</td>
<td>Configure actions to take on groups in Episode Review.</td>
<td>/SA-ITSI-MetricAD/default</td>
</tr>
<tr>
<td>notable_event_commonality.conf</td>
<td>Define fields to include or exclude from the Common Fields tab of Episode Review.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>notable_event_correlation.conf</td>
<td>Set threshold values and limits for Smart Mode event correlation.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>props.conf</td>
<td>Set indexing property configurations, including timezone offset, custom source type rules, and pattern collision priorities. Also, map transforms to event properties.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>restmap.conf</td>
<td>Create custom REST endpoints.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>File</td>
<td>Purpose</td>
<td>ITSI Location</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>savedsearches.conf</td>
<td>Define ordinary reports, scheduled reports, and alerts.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>searchbnf.conf</td>
<td>Configure the search assistant.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td></td>
<td>Configure settings for severity-level thresholds. Change the label, color, threshold level, health weight, minimum and maximum health score, and score contribution.</td>
<td>/itsi/default</td>
</tr>
<tr>
<td>threshold_labels.conf</td>
<td></td>
<td>/itsi/default</td>
</tr>
<tr>
<td>threshold_periods.conf</td>
<td>Deprecated. Do not edit.</td>
<td>/itsi/default</td>
</tr>
<tr>
<td>transforms.conf</td>
<td>Configure regex transformations to perform on data inputs. Use in tandem with props.conf.</td>
<td>/SA-ITOA/default</td>
</tr>
<tr>
<td>ui-tour.conf</td>
<td>Customize the ITSI product tour.</td>
<td>/itsi/default</td>
</tr>
<tr>
<td>visualizations.conf</td>
<td>Declare common visualizations that other modules can use.</td>
<td>/SA-ITSI-CustomModuleViz/default</td>
</tr>
<tr>
<td>File</td>
<td>Purpose</td>
<td>ITSI Location</td>
</tr>
<tr>
<td>----------</td>
<td>----------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>web.conf</td>
<td>Configure Splunk Web, enable HTTPS.</td>
<td>/SA-ITOA/default</td>
</tr>
</tbody>
</table>
Configuration file reference

alert_actions.conf

The following are the spec and example files for alert_actions.conf.

alert_actions.conf.spec

# This file contains possible attributes and values for generating ITSI notable events and configuring episode actions.
#
# There is an alert_actions.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/default/.
# To set custom configurations, place an alert_actions.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/local/. You must restart Splunk to enable
# configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the top
#   of the file.
# * Each .conf file should have at most one default stanza. If there are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in the
#   file wins.
# * If an attribute is defined at both the global level and in a specific
#   stanza, the value in the specific stanza takes precedence.

ttl = <integer> [p]
* The minimum time to live (TTL), in seconds, of the search artifacts if this action is triggered.
* If $p$ follows the integer, then the integer is the number of scheduled periods.
* Default: 600 (10 minutes)

maxtime = <integer> [m|s|h|d]
* The maximum amount of time that the execution of an action is allowed to take before the action is aborted.
* Use the d, h, m and s suffixes to define the period of time:
  d = day, h = hour, m = minute and s = second.
  For example: 5d means 5 days.
* If you do not include a suffix, the time defaults to seconds.
* Default: 600 (10 minutes)

maxresults = <integer>
* The maximum number of search results sent via the alert.
* Defaults to 10000.

is_custom = <boolean>
* Specifies whether the alert action is based on the custom alert actions framework and is supposed to be listed in the search UI.
* Default: 1

label = <string>
* Defines the label shown in the UI. If not specified, the stanza name is used instead.
* Default: "ITSI Alert Generator"

description = <string>
* Defines the description shown in the UI.
* Default: Send ITSI alert to summary index

payload_format = [xml|json]
* The format in which the alert script receives the configuration via STDIN.
* Default: json

[itsi_event_generator]

* Generate notable events under this stanza name.
* ITSI sends notable events to the ITSI summary index.
* Follow this stanza name with any number of the following attribute/value pairs.
* If you do not specify an entry for each attribute, Splunk will use the default value.

param.http_token_name = <string>
* The HTTP token name.
* Optional.
* If you do not provide a token name, ITSI obtains one
token using the index and sourcetype parameters below.

param.index = <string>
* The index name.
* This setting is required if you do not provide an HTTP token for 'param.http_token_name'.
* Default: itsi_tracked_alerts

param.sourcetype = <string>
* The sourcetype.
* This setting is used if you do not provide an HTTP token for the 'param.http_token_name' setting.
* Default: itsi_notable: event

param.event_identifier_fields = <comma-separated list>
* A list of fields that are used to identify event duplication.
* Default: source

param.search_type = <string>
* The search type.
* Default: custom

param.is_use_event_time = <boolean>
* If "1", ITSI uses the actual event time.
* If "0", ITSI uses the time the event was indexed.
* Default: 0

[itsi_sample_event_action_ping]

* Ping a host in one or more ITSI episodes under this stanza name.
* Follow this stanza name with any number of the following attribute/value pairs.
* If you do not specify an entry for each attribute, Splunk will use the default value.

param.host_to_ping = <string>
* The field from the episode representing the host to ping.
* If your event contains the field 'server', set to '%server%'.
* When ITSI executes the alert action, it extracts the value corresponding to the token value from event data and tries to ping it.
* If you set a value that does not begin and end with '%', ITSI considers this to be the value to ping. No extractions are done in this case.
* Default: %orig_host%
[itsi_event_action_link_ticket]

* Set options to associate an episode with a ticket from an external ticketing system under this stanza name.
* Follow this stanza name with any number of the following attribute/value pairs.
* If you do not specify an entry for each attribute, Splunk will use the default value.

param.ticket_system = <string>
* The name of the external ticketing system.
* This setting is required to create/update/delete a ticket.
* There is no default.

param.ticket_id = <string>
* The ID of the specific ticket to link to.
* This setting is required to create/update/delete a ticket.
* There is no default.

param.ticket_url = <string>
* The drilldown link to the ticket in the external ticketing system.
* This setting is required to create/update a ticket.
* There is no default.

param.operation = <upsert|delete>
* Specifies the type of action to take on the ticket.
* If "upsert", ITSI inserts or updates existing fields.
* If "delete", ITSI deletes the ticket.
* There is no default.

param.kwargs = <dict>
* A dictionary of additional fields to pass to the ticket.
* Optional.
* There is no default.

alert_actions.conf.example

No example

app_common_flags.conf

The following are the spec and example files for app_common_flags.conf.
This file contains attributes and values for disabling (feature flagging) certain ITSI features. There is an app_common_flags.conf in $SPLUNK_HOME/etc/apps/itsi/default. To set custom configurations, place an app_common_flags.conf in $SPLUNK_HOME/etc/apps/itsi/local/. You must restart Splunk software to enable configurations. To learn more about configuration files (including precedence) please see the documentation located at http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles. CAUTION: This is an internal configuration file used to turn off certain ITSI features that are incomplete. Do NOT edit or remove this file.

GLOBAL SETTINGS

Use the [default] stanza to define any global settings. You can also define global settings outside of any stanza, at the top of the file. Each .conf file should have at most one default stanza. If there are multiple default stanzas, attributes are combined. In the case of multiple definitions of the same attribute, the last definition in the file wins. If an attribute is defined at both the global level and in a specific stanza, the value in the specific stanza takes precedence.

<app_common_flag>

Each stanza represents a feature within Splunk IT Service Intelligence (ITSI). If the feature is disabled, it is currently incomplete and should NOT be enabled.
feature = <string>
* The name of the feature.

description = <string>
* A description of what the feature does.

disabled = <boolean>
* Whether the feature is enabled or disabled.
* If "1", the feature is disabled.
* If "0", the feature is enabled.

app_common_flags.conf.example

No example

authorize.conf

The following are the spec and example files for authorize.conf.

authorize.conf.spec

#   Version 7.2.0
#   This file contains possible attribute/value pairs for creating roles
#   in
#   authorize.conf. You can configure roles and granular access controls
#   by
#   creating your own authorize.conf.

#   There is an authorize.conf in $SPLUNK_HOME/etc/system/default/. To
#   set
#   custom configurations, place an authorize.conf in
#   $SPLUNK_HOME/etc/system/local/. For examples, see
#   authorize.conf.example.
#   You must restart Splunk to enable configurations.
#   
#   To learn more about configuration files (including precedence) please
#   see
#   the documentation located at
#   http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
#   * You can also define global settings outside of any stanza, at the
top
#     of the file.
#   * Each conf file should have at most one default stanza. If there
are
#     multiple default stanzas, attributes are combined. In the case of
#     multiple definitions of the same attribute, the last definition in
#     the file wins.
#   * If an attribute is defined at both the global level and in a
specific
#     stanza, the value in the specific stanza takes precedence.

[default]

srchFilterSelecting = <boolean>
* Determines whether a role's search filters will be used for selecting
or
   eliminating during role inheritance.
* Selecting will join the search filters with an OR when combining the
   filters.
* Eliminating will join the search filters with an AND when combining
   the
   filters.
   * All roles will default to true (in other words, selecting).
   * Example:
     * role1 srchFilter = sourcetype!=ex1 with selecting=true
     * role2 srchFilter = sourcetype=ex2 with selecting = false
     * role3 srchFilter = sourcetype!=ex3 AND index=main with selecting =
true
     * role3 inherits from role2 and role 2 inherits from role1
     * Resulting srchFilter = ((sourcetype!=ex1) OR
       (sourcetype!=ex3 AND index=main)) AND ((sourcetype=ex2))

[capability::<capability>]

* DO NOT edit, remove, or add capability stanzas. The existing
capabilities
   are the full set of Splunk system capabilities.
* Splunk adds all of its capabilities this way
* For the default list of capabilities and assignments, see
authorize.conf
   under the 'default' directory
* Only alphanumeric characters and "_" (underscore) are allowed in
capability names.
Examples:
* edit_visualizations
* view_license
* Descriptions of specific capabilities are listed below.

[role_<roleName>]

<capability> = <enabled>
* A capability that is enabled for this role.
* You can list many of these.
* Note that 'enabled' is the only accepted value here, as capabilities are
disabled by default.
* Roles inherit all capabilities from imported roles, and inherited capabilities cannot be disabled.
* Role names cannot have uppercase characters. User names, however, are case-insensitive.

importRoles = <string>
* Semicolon delimited list of other roles and their associated capabilities
  that should be imported.
* Importing other roles also imports the other aspects of that role, such as
  allowed indexes to search.
* By default a role imports no other roles.

grantableRoles = <string>
* Semicolon delimited list of roles that can be granted when edit_user
capability is present.
* By default, a role with 'edit_user' capability can create/edit a user and
  assign any role to them. Roles assigned to users can be restricted by assigning
  'edit_grantable_role' capability and specifying the roles in 'grantableRoles'.
  When you set 'grantableRoles', the roles that can be assigned will be
  restricted to the ones whose capabilities are a proper subset of those in the
  roles provided.
* For a role that has no edit_user capability, grantableRoles has no effect.
* NOTE: A role that has been assigned 'grantableRoles' can list only the users
  whose capabilities are a subset of all capabilities of the roles assigned to
  'grantableRoles'.
* Example:
  Consider a Splunk instance where role1-4 are assigned the following
capabilities:
  role1: c1, c2, c3
  role2: c4, c5, c6
  role3: c1, c6
  role4: c4, c8

Users user1-4 are assigned the following roles:
user1: role1
user2: role2
user3: role3
user4: role4

grantableRoles is defined as follows for the admin role:

[role_admin]

  grantableRoles = role1;role2

  For the above configuration, the admin user can list/edit only user1, user2
  and user3 and can only assign roles role1, role2, and role3 to those
  users.
  * Defaults to not present.

srchFilter = <string>
* Semicolon delimited list of search filters for this Role.
* By default we perform no search filtering.
* To override any search filters from imported roles, set this to '*',
  as
  the 'admin' role does.

srchTimeWin = <number>
* Maximum time span of a search, in seconds.
  * This time window limit is applied backwards from the latest time
    specified in a search.
  * By default, searches are not limited to any specific time window.
  * To override any search time windows from imported roles, set this to
    '0'
    (infinite), as the 'admin' role does.
* -1 is a special value that implies no search window has been set for
  this role
  * This is equivalent to not setting srchTimeWin at all, which means it
    can be easily overridden by an imported role

srchDiskQuota = <number>
* Maximum amount of disk space (MB) that can be used by search jobs of a
  user that belongs to this role
* In search head clustering environments, this setting takes effect on a
  per-member basis. There is no cluster-wide accounting.
* The dispatch manager checks the quota at the dispatch time of a search
  and additionally the search process will check at intervals that are
  defined
  in the 'disk_usage_update_period' setting in limits.conf as long as
  the
  search is active.
* The quota can be exceeded at times, since the search process does not
  check
  the quota constantly.
* Exceeding this quota causes the search to be auto-finalized
  immediately,
  even if there are results that have not yet been returned.
* Defaults to '100', for 100 MB.

srchJobsQuota = <number>
* Maximum number of concurrently running historical searches a member of
  this role can have.
* This excludes real-time searches, see rtSrchJobsQuota.
* Defaults to 3.

rtSrchJobsQuota = <number>
* Maximum number of concurrently running real-time searches a member of
  this
  role can have.
* Defaults to 3.

srchMaxTime = <number><unit>
* Maximum amount of time that searches of users from this role will be
  allowed to run.
* Once the search has been ran for this amount of time it will be auto
  finalized, If the role
* Inherits from other roles, the maximum srchMaxTime value specified in
  the
  included roles.
* This maximum does not apply to real-time searches.
* Examples: 1h, 10m, 2hours, 2h, 2hrs, 100s
* Defaults to 100days

srchIndexesDefault = <string>
* A semicolon-delimited list of indexes to search when no index is
  specified.
* These indexes can be wild-carded ("**"), with the exception that '**'
  does not
  match internal indexes.
* To match internal indexes, start with '_'. All internal indexes are
  represented by '_*'.
* The wildcard character '*' is limited to match either all the
  non-internal
  indexes or all the internal indexes, but not both at once.
* If you make any changes in the "Indexes searched by default" Settings
  panel
  for a role in Splunk Web, those values take precedence, and any
wildcards
you specify in this setting are lost.
* Defaults to none.

srchIndexesAllowed = <string>
* Semicolon delimited list of indexes this role is allowed to search
* Follows the same wildcarding semantics as srchIndexesDefault
* If you make any changes in the "Indexes" Settings panel
  for a role in Splunk Web, those values take precedence, and any
  wildcards
  you specify in this setting are lost.
* Defaults to none.

deleteIndexesAllowed = <string>
* Semicolon delimited list of indexes this role is allowed to delete
* This setting must be used in conjunction with the delete_by_keyword
capability
* Follows the same wildcarding semantics as srchIndexesDefault
* Defaults to none

cumulativeSrchJobsQuota = <number>
* Maximum number of concurrently running historical searches in total
  across all members of this role
* Requires enable_cumulative_quota = true in limits.conf to take
effect.
* If a user belongs to multiple roles, the user's searches count against
  the role with the largest cumulative search quota. Once the quota for
  that role is consumed, the user's searches count against the role with
  the next largest quota, and so on.
* In search head clustering environments, this setting takes effect on a
  per-member basis. There is no cluster-wide accounting.

cumulativeRTSrchJobsQuota = <number>
* Maximum number of concurrently running real-time searches in total
  across all members of this role
* Requires enable_cumulative_quota = true in limits.conf to take
effect.
* If a user belongs to multiple roles, the user's searches count against
  the role with the largest cumulative search quota. Once the quota for
  that role is consumed, the user's searches count against the role with
  the next largest quota, and so on.
* In search head clustering environments, this setting takes effect
  on a per-member basis. There is no cluster-wide accounting.

### Descriptions of Splunk system capabilities. Capabilities are added
to roles,
to which users are then assigned. When a user is assigned a role, they
acquire
the capabilities added to that role.
[capability::accelerate_datamodel]

* Lets a user enable or disable datamodel acceleration.

[capability::accelerate_search]

* Lets a user enable or disable acceleration for reports.
* The assigned role must also be granted the schedule_search capability.

[capability::run_multi_phased_searches]

* Lets a user in a distributed search environment run searches with three or more map-reduce phases
* Allows users to take advantage of the search performance gains related to parallel reduce functionality.
* Multiphased searches can lead to higher resource utilization on indexers, but they can also reduce resource utilization on search heads.

[capability::admin_all_objects]

* Lets a user access all objects in the system, such as user objects and knowledge objects.
* Lets a user bypasses any ACL restrictions, much the way root access in a *nix environment does.
* Splunk checks this capability when accessing manager pages and objects.

[capability::change_authentication]

* Lets a user change authentication settings through the authentication endpoints.
* Lets the user reload authentication.

[capability::change_own_password]

* Lets a user change their own password. You can remove this capability to control the password for a user.
[capability::delete_by_keyword]

* Lets a user use the "delete" search operator. Note that this does not actually delete the raw data on disk, instead it masks the data (via the index) from showing up in search results.

[capability::dispatch_rest_to_indexers]

* Lets a user dispatch the REST search command to indexers.

[capability::edit_deployment_client]

* Lets a user edit the deployment client.
* Lets a user edit a deployment client admin endpoint.

[capability::edit_deployment_server]

* Lets a user edit the deployment server.
* Lets a user edit a deployment server admin endpoint.
* Lets a user change or create remote inputs that are pushed to the forwarders and other deployment clients.

[capability::edit_dist_peer]

* Lets a user add and edit peers for distributed search.

[capability::edit_encryption_key_provider]

* Lets a user view and edit keyprovider properties when using the Server-Side Encryption (SSE) feature for a remote storage volume.

[capability::request_pstacks]

* Lets a user trigger pstacks generation of the main splunkd process using a REST endpoint.

[capability::edit_forwarders]

* Lets a user edit settings for forwarding data, including settings
for SSL, backoff schemes, etc.
* Also used by TCP and Syslog output admin handlers.

[capability::edit_health]

* Lets a user disable or enable health reporting for a feature in the splunkd health status tree through the server/health-config/{feature_name} endpoint.

[capability::edit_httpauths]

* Lets a user edit and end user sessions through the httpauth-tokens endpoint.

[capability::edit_indexer_cluster]

* Lets a user edit or manage indexer clusters.

[capability::edit_indexerdiscovery]

* Lets a user edit settings for indexer discovery, including settings for master_uri, pass4SymmKey, etc.
* Also used by Indexer Discovery admin handlers.

[capability::edit_input_defaults]

* Lets a user change the default hostname for input data through the server settings endpoint.

[capability::edit_monitor]

* Lets a user add inputs and edit settings for monitoring files.
* Also used by the standard inputs endpoint as well as the one-shot input endpoint.
[capability::edit_modinput_winhostmon]

* Lets a user add and edit inputs for monitoring Windows host data.

[capability::edit_modinput_winnetmon]

* Lets a user add and edit inputs for monitoring Windows network data.

[capability::edit_modinput_winprintmon]

* Lets a user add and edit inputs for monitoring Windows printer data.

[capability::edit_modinput_perfmon]

* Lets a user add and edit inputs for monitoring Windows performance.

[capability::edit_modinput_admon]

* Lets a user add and edit inputs for monitoring Splunk's Active Directory.

[capability::edit_roles]

* Lets a user edit roles.
* Lets a user change the mappings from users to roles.
* Used by both the user and role endpoint.

[capability::edit_roles_grantable]

* Lets the user edit roles and change user-to-role mappings for a limited set of roles.
* To limit this ability, also assign the edit_roles_grantable capability and configure grantableRoles in authorize.conf. For example: grantableRoles = role1;role2;role3. This lets user create roles using the subset of capabilities that the user has in their grantable_roles configuration.
[capability::edit_scripted]

* Lets a user create and edit scripted inputs.

[capability::edit_search_head_clustering]

* Lets a user edit and manage search head clustering.

[capability::edit_search_scheduler]

* Lets the user disable and enable the search scheduler.

[capability::edit_search_schedule_priority]

* Lets a user assign a search a higher-than-normal schedule priority.

[capability::edit_search_schedule_window]

* Lets a user edit a search schedule window.

[capability::edit_search_server]

* Lets a user edit general distributed search settings like timeouts, heartbeats, and blacklists.

[capability::edit_server]

* Lets the user edit general server and introspection settings, such as the server name, log levels, etc.
  * This capability also inherits the ability to read general server and introspection settings.

[capability::edit_server_crl]

* Lets a user reload Certificate Revocation List within Splunk.
[capability::edit_sourcetypes]

* Lets a user create and edit sourcetypes.

[capability::edit_splunktcp]

* Lets a user change settings for receiving TCP input from another Splunk instance.

[capability::edit_splunktcp_ssl]

* Lets a user view and edit SSL-specific settings for Splunk TCP input.

[capability::edit_splunktcp_token]

* Lets a user view or edit splunktcptokens. The tokens can be used on a receiving system to only accept data from forwarders that have been configured with the same token.

[capability::edit_tcp]

* Lets a user change settings for receiving general TCP inputs.

[capability::edit_telemetry_settings]

* Lets a user change settings to opt-in and send telemetry data.

[capability::edit_token_http]

* Lets a user create, edit, display, and remove settings for HTTP token input.
  * Enables the HTTP Events Collector feature.

[capability::edit_udp]

* Lets a user change settings for UDP inputs.
[capability::edit_user]

* Lets a user create, edit, or remove other users. To limit this ability,
  assign the edit_roles_grantable capability and configure grantableRoles
  in authorize.conf. For example: grantableRoles = role1;role2;role3.
* Also lets a user manage certificates for distributed search.

[capability::edit_view_html]

* Lets a user create, edit, or otherwise modify HTML-based views.

[capability::edit_web_settings]

* Lets a user change the settings for web.conf through the system settings
endpoint.

[capability::export_results_is_visible]

* Lets a user show or hide the Export button in Splunk Web.
* Disable this setting to hide the Export button and prevent users with this role from exporting search results.

[capability::get_diag]

* Lets the user get remote diag from an instance through the /streams/diag endpoint.

[capability::get_metadata]

* Lets a user use the "metadata" search processor.

[capability::get_typeahead]

* Enables typeahead for a user, both the typeahead endpoint and the 'typeahead' search processor.
[capability::indexes_edit]

* Lets a user change any index settings such as file size and memory limits.

[capability::input_file]

* Lets a user add a file as an input through inputcsv (except for dispatch=t mode) and inputlookup.

[capability::license_tab]

* (Deprecated) Lets a user access and change the license.

[capability::license_edit]

* Lets a user access and change the license.

[capability::license_view_warnings]

* Lets a user see if they are exceeding limits or reaching the expiration date of their license.
* License warnings are displayed on the system banner.

[capability::list_deployment_client]

* Lets a user list the deployment clients.

[capability::list_deployment_server]

* Lets a user list the deployment servers.

[capability::list_forwarders]

* Lets a user list settings for data forwarding.
* Used by TCP and Syslog output admin handlers.
[capability::list_health]

* Lets a user monitor the health of various Splunk features (such as inputs, outputs, clustering, etc) through REST endpoints.

[capability::list_httpauths]

* Lets a user list user sessions through the httpauth-tokens endpoint.

[capability::list_indexer_cluster]

* Lets a user list indexer cluster objects such as buckets, peers, etc.

[capability::list_indexerdisclosure]

* Lets a user view settings for indexer discovery.
* Used by Indexer Discovery handlers.

[capability::list_inputs]

* Lets a user view the list of inputs, including files, TCP, UDP, Scripts, etc.

[capability::list_introspection]

* Lets a user read introspection settings and statistics for indexers, search, processors, queues, etc.

[capability::list_search_head_clustering]

* Lets a user list search head clustering objects such as artifacts, delegated jobs, members, captain, etc.

[capability::list_search_scheduler]

* Lets a user list search scheduler settings.
[capability::list_settings]

* Lets a user list general server and introspection settings such as the server name, log levels, etc.

[capability::list_metrics_catalog]

* Lets a user list metrics catalog information such as the metric names, dimensions, and dimension values.

[capability::list_storage_passwords]

* Lets a user access the /storage/passwords endpoint.
* Lets the user perform GETs.
* The admin_all_objects capability must added to the role in order for the user to perform POSTs to the /storage/passwords endpoint.

[capability::never_lockout]

* Lets a user account never lockout.

[capability::never_expire]

* Lets a user account never expire.

[capability::output_file]

* Lets a user create file outputs, including outputcsv (except for dispatch=t mode) and outputlookup.

[capability::request_remote_tok]

* Lets a user get a remote authentication token.
* Used for distributing search to old 4.0.x Splunk instances.
* Also used for some distributed peer management and bundle replication.
[capability::rest_apps_management]

* Lets a user edit settings for entries and categories in the python remote apps handler.
* See restmap.conf for more information.

[capability::rest_apps_view]

* Lets a user list various properties in the python remote apps handler.
* See restmap.conf for more info

[capability::rest_properties_get]

* Lets a user get information from the services/properties endpoint.

[capability::rest_properties_set]

* Lets a user edit the services/properties endpoint.

[capability::restart_splunkd]

* Lets a user restart Splunk through the server control handler.

[capability::rtsearch]

* Lets a user run realtime searches.

[capability::run_collect]

* Lets a user run the collect command.

[capability::run_mcollect]

* Lets a user run the mcollect and meventcollect commands.
[capability::run_debug_commands]

* Lets a user run debugging commands, for example "summarize".

[capability::schedule_rtsearch]

* Lets a user schedule real time saved searches. The scheduled_search
  scheduled_search and rtsearch capabilities must be enabled for the role.

[capability::schedule_search]

* Lets a user schedule saved searches, create and update alerts, and
  review triggered alert information.

[capability::search]

* Lets a user run a search.

[capability::search_process_config_refresh]

* Lets a user manually flush idle search processes through the
  "refresh search-process-config"CLI command.

[capability::use_file_operator]

* Lets a user use the "file" search operator. The "file" search operator
  is DEPRECATED.

[capability::web_debug]

* Lets a user access /_bump and /debug/** web debug endpoints.

[capability::edit_statsd_transforms]

* Lets a user define regular expressions to extract munged dimensions
  out of
  metric_name in statsd metric data using
  services/data/transforms/statsdextractions
  endpoint.
[capability::edit_metric_schema]

* Lets a user define the schema of the log data which needs to be converted into metric format using services/data/metric-transforms/schema endpoint.

[capability::list_workload_pools]

* Lets a user list and view workload pool and workload status information through the workloads endpoint.

[capability::edit_workload_pools]

* Lets a user create and edit workload pool and workload config information (except workload rule) through the workloads endpoint.

[capability::select_workload_pools]

* Lets a user select a workload pool for a scheduled or an ad-hoc search.

[capability::list_workload_rules]

* Lets a user list and view workload rule information from the workload/rules endpoint.

[capability::edit_workload_rules]

* Lets a user create and edit workload rules through the workloads/rules endpoint.

authorize.conf.example

#  Version 7.2.0
#
# This is an example authorize.conf. Use this file to configure roles and capabilities.

# To use one or more of these configurations, copy the configuration block into authorize.conf in $SPLUNK_HOME/etc/system/local/. You must reload auth or restart Splunk to enable configurations.

# To learn more about configuration files (including precedence) please see the documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

[role_ninja]
rtsearch = enabled
importRoles = user
srcrFilter = host=foo
srcrIndexesAllowed = *
srchIndexesDefault = mail;main
srcrJobsQuota = 8
rtSrcrJobsQuota = 8
srcrDiskQuota = 500

# This creates the role 'ninja', which inherits capabilities from the 'user' role. ninja has almost the same capabilities as power, except cannot schedule searches.

# The search filter limits ninja to searching on host=foo.

# ninja is allowed to search all public indexes (those that do not start with underscore), and will search the indexes mail and main if no index is specified in the search.

# ninja is allowed to run 8 search jobs and 8 real time search jobs concurrently (these counts are independent).

# ninja is allowed to take up 500 megabytes total on disk for all their jobs.

commands.conf

The following are the spec and example files for commands.conf.
commands.conf.spec

#   Version 7.2.0
#   This file contains possible attribute/value pairs for creating search
#   commands for any custom search scripts created. Add your custom search
#   script to $SPLUNK_HOME/etc/searchscripts/ or
#   $SPLUNK_HOME/etc/apps/MY_APP/bin/. For the latter, put a custom
#   commands.conf in $SPLUNK_HOME/etc/apps/MY_APP. For the former, put
#   your
#   custom commands.conf in $SPLUNK_HOME/etc/system/local/.

# There is a commands.conf in $SPLUNK_HOME/etc/system/default//. For examples,
# see commands.conf.example. You must restart Splunk to enable configurations.

# To learn more about configuration files (including precedence) please see the
# documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
#   * You can also define global settings outside of any stanza, at the top of
#     the file.
#   * Each conf file should have at most one default stanza. If there are
#     multiple default stanzas, attributes are combined. In the case of
#     multiple definitions of the same attribute, the last definition in the
#     file wins.
#   * If an attribute is defined at both the global level and in a
#     specific
#     stanza, the value in the specific stanza takes precedence.

[<STANZA_NAME>]

* Each stanza represents a search command; the command is the stanza name.
* The stanza name invokes the command in the search language.
* Set the following attributes/values for the command. Otherwise, Splunk uses the defaults.
* If the filename attribute is not specified, Splunk searches for an external program by appending extensions (e.g. ".py", ".pl") to the stanza name.
* If chunked = true, in addition to ".py" and ".pl" as above, Splunk searches using the extensions ".exe", ".bat", ".cmd", ".sh", ".js", and no extension (to find extensionless binaries).
* See the filename attribute for more information about how Splunk searches for external programs.

```
type = <string>
* Type of script: python, perl
  * Defaults to python.

filename = <string>
* Optionally specify the program to be executed when the search command is used.
* Splunk looks for the given filename in the app's bin directory.
* The filename attribute can not reference any file outside of the app's bin directory.
* If the filename ends in ".py", Splunk's python interpreter is used to invoke the external script.
* If chunked = true, Splunk looks for the given filename in 
  $SPLUNK_HOME/etc/apps/MY_APP/<PLATFORM>/bin before searching $SPLUNK_HOME/etc/apps/MY_APP/bin, where <PLATFORM> is one of  "linux_x86_64", "linux_x86", "windows_x86_64", "windows_x86",  "darwin_x86_64" (depending on the platform on which Splunk is running on).
* If chunked = true and if a path pointer file (*.path) is specified, the contents of the file are read and the result is used as the command to be run. Environment variables in the path pointer file are substituted. Path pointer files can be used to reference system binaries (e.g. /usr/bin/python).

command.arg.<N> = <string>
* Additional command-line arguments to use when invoking this program. Environment variables will be substituted (e.g. $SPLUNK_HOME).
  * Only available if chunked = true.

local = [true|false]
* If true, specifies that the command should be run on the search head only
  * Defaults to false

perf_warn_limit = <integer>
* Issue a performance warning message if more than this many input events are passed to this external command (0 = never)
  * Defaults to 0 (disabled)
streaming = [true|false]
  * Specify whether the command is streamable.
  * Defaults to false.

maxinputs = <integer>
  * Maximum number of events that can be passed to the command for each
    invocation.
  * This limit cannot exceed the value of maxresultrows in limits.conf.
  * 0 for no limit.
  * Defaults to 50000.

passauth = [true|false]
  * If set to true, splunkd passes several authentication-related facts
    at the start of input, as part of the header (see enableheader).
  * The following headers are sent
    * authString: psuedo-xml string that resembles
      <auth><userId>username</userId><username>username</username><authToken>auth_token</authToken></auth>
      where the username is passed twice, and the authToken may be used
      to contact splunkd during the script run.
    * sessionKey: the session key again.
    * owner: the user portion of the search context
    * namespace: the app portion of the search context
  * Requires enableheader = true; if enableheader = false, this flag will
    be treated as false as well.
  * Defaults to false.
  * If chunked = true, this attribute is ignored. An authentication
    token is always passed to commands using the chunked custom search
    command protocol.

run_in_preview = [true|false]
  * Specify whether to run this command if generating results just for
    preview
    rather than final output.
  * Defaults to true

enableheader = [true|false]
  * Indicate whether or not your script is expecting header information or
    not.
  * Currently, the only thing in the header information is an auth token.
  * If set to true it will expect as input a head section + '
\n' then the
    csv input
  * NOTE: Should be set to true if you use splunk.Intersplunk
  * Defaults to true.

retainsevents = [true|false]
  * Specify whether the command retains events (the way the
    sort/dedup/cluster
    commands do) or whether it transforms them (the way the stats command
    does).
  * Defaults to false.
generating = [true|false]
* Specify whether your command generates new events. If no events are passed to
  the command, will it generate events?
* Defaults to false.

generates_timeorder = [true|false]
* If generating = true, does command generate events in descending time order
  (latest first)
* Defaults to false.

overrides_timeorder = [true|false]
* If generating = false and streaming=true, does command change the order of
  events with respect to time?
* Defaults to false.

requires_preop = [true|false]
* Specify whether the command sequence specified by the 'streaming_preop' key
  is required for proper execution or is it an optimization only
* Default is false (streaming_preop not required)

streaming_preop = <string>
* A string that denotes the requested pre-streaming search string.

required_fields = <string>
* A comma separated list of fields that this command may use.
* Informs previous commands that they should retain/extract these fields if
  possible. No error is generated if a field specified is missing.
* Defaults to '*'

supports_multivalues = [true|false]
* Specify whether the command supports multivalues.
  * If true, multivalues will be treated as python lists of strings,
    instead of a
    * flat string (when using Intersplunk to interpret stdin/stdout).
  * If the list only contains one element, the value of that element will be
    returned, rather than a list
    * (for example, isinstance(val, basestring) == True).

supports_getinfo = [true|false]
* Specifies whether the command supports dynamic probing for settings
  (first argument invoked == __GETINFO__ or __EXECUTE__).

supports_rawargs = [true|false]
* Specifies whether the command supports raw arguments being passed to it or if
  it prefers parsed arguments (where quotes are stripped).
* If unspecified, the default is false

undo_scheduler_escaping = [true|false]
* Specifies whether the commands raw arguments need to be unescaped.
* This is particularly applies to the commands being invoked by the scheduler.
* This applies only if the command supports raw arguments(supports_rawargs).
* If unspecified, the default is false

requires_srinfo = [true|false]
* Specifies if the command requires information stored in SearchResultsInfo.
* If true, requires that enableheader be set to true, and the full pathname of the info file (a csv file) will be emitted in the header under the key 'infoPath'
* If unspecified, the default is false

needs_empty_results = [true|false]
* Specifies whether or not this search command needs to be called with intermediate empty search results
* If unspecified, the default is true

changes_colorder = [true|false]
* Specify whether the script output should be used to change the column ordering of the fields.
* Default is true

outputheader = <true/false>
* If set to true, output of script should be a header section + blank line + csv output
* If false, script output should be pure csv only
* Default is false

clear_required_fields = [true|false]
* If true, required_fields represents the *only* fields required.
* If false, required_fields are additive to any fields that may be required by subsequent commands.
* In most cases, false is appropriate for streaming commands and true for reporting commands
* Default is false

stderr_dest = [log|message|none]
* What do to with the stderr output from the script
* 'log' means to write the output to the job's search.log.
* 'message' means to write each line as an search info message. The message level can be set to adding that level (in ALL CAPS) to the start of
the line, e.g. "WARN my warning message."
* 'none' means to discard the stderr output
* Defaults to log

is_order_sensitive = [true|false]
* Specify whether the command requires ordered input.
* Defaults to false.

is_risky = [true|false]
* Searches using Splunk Web are flagged to warn users when they unknowingly run a search that contains commands that might be a security risk. This warning appears when users click a link or type a URL that loads a search that contains risky commands. This warning does not appear when users create ad hoc searches.
* This flag is used to determine whether the command is risky.
* Defaults to false.
* - Specific commands that ship with the product have their own defaults

chunked = [true|false]
* If true, this command supports the new "chunked" custom search command protocol.
* If true, the only other commands.conf attributes supported are is_risky, maxwait, maxchunksize, filename, and command.arg.<N>.
* If false, this command uses the legacy custom search command protocol supported by Intersplunk.py.
* Default is false

maxwait = <integer>
* Only available if chunked = true.
* Not supported in Windows.
* The value of maxwait is the maximum number of seconds the custom search command can pause before producing output.
* If set to 0, the command can pause forever.
* Default is 0

maxchunksize = <integer>
* Only available if chunked = true.
* The value of maxchunksize is maximum size chunk (size of metadata plus size of body) the external command may produce. If the command tries to produce a larger chunk, the command is terminated.
* If set to 0, the command may send any size chunk.
* Default is 0

commands.conf.example

# Version 7.2.0
#
# This is an example commands.conf. Use this file to configure settings
# for external search commands.
#
# To use one or more of these configurations, copy the configuration block
# into commands.conf in $SPLUNK_HOME/etc/system/local/. You must restart
# Splunk to enable configurations.
#
# To learn more about configuration files (including precedence)
# see the documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
#
# Note: These are examples. Replace the values with your own
# customizations.

###################
# defaults for all external commands, exceptions are below in
# individual stanzas

# type of script: 'python', 'perl'
TYPE = python
# default ?filename? would be <stanza-name>.py for python,
# <stanza-name>.pl for perl, and
# <stanza-name> otherwise

# is command streamable?
streaming = false

# maximum data that can be passed to command (0 = no limit)
maxinputs = 50000

# end defaults
###################

[crawl]
filename = crawl.py

[createrss]
filename = createrss.py

[diff]
filename = diff.py

[gentimes]
filename = gentimes.py

[head]
filename = head.py

[loglady]
The following are the spec and example files for collections.conf.

collections.conf.spec

#   Version 7.2.0
#
#   This file configures the KV Store collections for a given app in Splunk.
#   To learn more about configuration files (including precedence) please see
#   the documentation located at
#   http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
enforceTypes = true|false
* Indicates whether to enforce data types when inserting data into the collection.
* When set to true, invalid insert operations fail.
* When set to false, invalid insert operations drop only the invalid field.
* Defaults to false.

field.<name> = number|bool|string|time
* Field type for a field called <name>.
* If the data type is not provided, it is inferred from the provided JSON data type.

accelerated_fields.<name> = <json>
* Acceleration definition for an acceleration called <name>.
* Must be a valid JSON document (invalid JSON is ignored).
* Example: 'acceleration.foo={"a":1, "b":-1}' is a compound acceleration that first sorts 'a' in ascending order and then 'b' in descending order.
* There are restrictions in compound acceleration. A compound acceleration must not have more than one field in an array. If it does, KV Store does not start or work correctly.
* If multiple accelerations with the same definition are in the same collection, the duplicates are skipped.
* If the data within a field is too large for acceleration, you will see a warning when you try to create an accelerated field and the acceleration will not be created.
* An acceleration is always created on the _key.
* The order of accelerations is important. For example, an acceleration of { "a":1, "b":1 } speeds queries on "a" and "a" + "b", but not on "b" lone.
* Multiple separate accelerations also speed up queries. For example, separate accelerations { "a": 1 } and { "b": 1 } will speed up queries on "a" + "b", but not as well as a combined acceleration { "a":1, "b":1 }.
* Defaults to nothing (no acceleration).

profilingEnabled = true|false
* Indicates whether to enable logging of slow-running operations, as defined
profilingThresholdMs = <zero or positive integer>
* The threshold for logging a slow-running operation, in milliseconds.
* When set to 0, all operations are logged.
* This setting is only used when 'profilingEnabled' is true.
* This setting impacts the performance of the collection.
* Defaults to 1000.

replicate = true|false
* Indicates whether to replicate this collection on indexers. When
false,
  this collection is not replicated, and lookups that depend on this
  collection will not be available (although if you run a lookup command
  with 'local=true', local lookups will still be available). When true,
  this collection is replicated on indexers.
* Defaults to false.

replication_dump_strategy = one_file|auto
* Indicates how to store dump files. When set to one_file, dump files
  are
  stored in a single file. When set to auto, dumps are stored in
  multiple
  files when the size of the collection exceeds the value of
  'replication_dump_maximum_file_size'.
* Defaults to auto.

replication_dump_maximum_file_size = <unsigned integer>
* Specifies the maximum file size (in KB) for each dump file when
  'replication_dump_strategy=auto'.
* If this value is larger than 'concerningReplicatedFileSize', which is set
in distsearch.conf, the value of 'concerningReplicatedFileSize' will be
  used instead.
* KV Store does not pre-calculate the size of the records that will be
  written
  to disk, so the size of the resulting files can be affected by the
  'max_rows_in_memory_per_dump' setting from 'limits.conf'.
* Defaults to 10240KB.

type = internal_cache|undefined
* Indicates the type of data that this collection holds.
* When set to 'internal_cache', changing the configuration of the current
  instance between search head cluster, search head pool, or standalone
  will erase the data in the collection.
* Defaults to 'undefined'.
* For internal use only.
collections.conf.example

#   Version 7.2.0
#
#   The following is an example collections.conf configuration.
#   To use one or more of these configurations, copy the configuration block
#   into collections.conf in $SPLUNK_HOME/etc/system/local/. You must restart
#   Splunk to enable configurations.
#
#   To learn more about configuration files (including precedence) please see
#   the documentation located at
#   http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
#   Note this example uses a compound acceleration. Please check
#   collections.conf.spec
#   for restrictions on compound acceleration.

[mycollection]

field.foo = number
field.bar = string
accelerated_fields.myacceleration = {"foo": 1, "bar": -1}

datamodels.conf

The following are the spec and example files for datamodels.conf.

datamodels.conf.spec

#   Version 7.2.0
#
#   This file contains possible attribute/value pairs for configuring data models. To configure a datamodel for an app, put your custom
#   datamodels.conf in $SPLUNK_HOME/etc/apps/MY_APP/local/
#
#   For examples, see datamodels.conf.example. You must restart Splunk to
#   enable configurations.
#
#   To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
#   * You can also define global settings outside of any stanza, at the top
#   of the file.
#   * Each conf file should have at most one default stanza. If there are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in the
#   file wins.
#   * If an attribute is defined at both the global level and in a specific
#     stanza, the value in the specific stanza takes precedence.

[<datamodel_name>]

* Each stanza represents a data model. The data model name is the stanza name.

acceleration = <bool>
* Set acceleration to true to enable automatic acceleration of this data model.
* Automatic acceleration creates auxiliary column stores for the fields
  and values in the events for this datamodel on a per-bucket basis.
* These column stores take additional space on disk, so be sure you have the
  proper amount of disk space. Additional space required depends on the
  number of events, fields, and distinct field values in the data.
* The Splunk software creates and maintains these column stores on a
  schedule
  you can specify with 'acceleration.cron_schedule.' You can query
  them with the 'tstats' command.

acceleration.earliest_time = <relative-time-str>
* Specifies how far back in time the Splunk software should keep these column
  stores (and create if acceleration.backfill_time is not set).
* Specified by a relative time string. For example, '-7d' means
  'accelerate
  data within the last 7 days.'
acceleration.backfill_time = <relative-time-str>
* ADVANCED: Specifies how far back in time the Splunk software should create its column stores.
* ONLY set this parameter if you want to backfill less data than the retention period set by 'acceleration.earliest_time'. You may want to use this parameter to limit your time window for column store creation in a large environment where initial creation of a large set of column stores is an expensive operation.
* WARNING: Do not set 'acceleration.backfill_time' to a narrow time window. If one of your indexers is down for a period longer than this backfill time, you may miss accelerating a window of your incoming data.
* MUST be set to a more recent time than 'acceleration.earliest_time'. For example, if you set 'acceleration.earliest_time' to '-1y' to retain your column stores for a one year window, you could set 'acceleration.backfill_time' to '-20d' to create column stores that only cover the last 20 days. However, you cannot set 'acceleration.backfill_time' to '-2y', because that goes farther back in time than the 'acceleration.earliest_time' setting of '-1y'.
* Defaults to empty string (unset). When 'acceleration.backfill_time' is unset, the Splunk software always backfills fully to 'acceleration.earliest_time'.

acceleration.max_time = <unsigned int>
* The maximum amount of time that the column store creation search is allowed to run (in seconds).
* Note that this is an approximate time.
* Defaults to: 3600
* An 'acceleration.max_time' setting of '0' indicates that there is no time limit.

acceleration.poll_buckets_until_maxtime = <bool>
* In a distributed environment that consist of heterogenous machines, summarizations might complete sooner on machines with less data and faster resources. After the summarization search is finished with all of the buckets, the search ends. However, the overall search runtime is
determined by the slowest machine in the environment.
* When set to "true": All of the machines run for "max_time" (approximately).
The buckets are polled repeatedly for new data to summarize
* Set this to true if your data model is sensitive to summarization latency delays.
* When this setting is enabled, the summarization search is counted against the number of concurrent searches you can run until "max_time" is reached.
* Default: false

acceleration.cron_schedule = <cron-string>
* Cron schedule to be used to probe/generate the column stores for this data model.
* Defaults to: */5 * * * *

acceleration.manual_rebuilds = <bool>
* ADVANCED: When set to 'true,' this setting prevents outdated summaries from being rebuilt by the 'summarize' command.
* Normally, during the creation phase, the 'summarize' command automatically rebuilds summaries that are considered to be out-of-date, such as when the configuration backing the data model changes.
* The Splunk software considers a summary to be outdated when:
  * The data model search stored in its metadata no longer matches its current data model search.
  * The search stored in its metadata cannot be parsed.
* NOTE: If the Splunk software finds a partial summary be outdated, it always rebuilds that summary so that a bucket summary only has results corresponding to one datamodel search.
* Defaults to: false

acceleration.max_concurrent = <unsigned int>
* The maximum number of concurrent acceleration instances for this data model that the scheduler is allowed to run.
* Defaults to: 3

acceleration.allow_skew = <percentage>|<duration-specifier>
* Allows the search scheduler to randomly distribute scheduled searches more evenly over their periods.
* When set to non-zero for searches with the following cron_schedule values, the search scheduler randomly "skews" the second, minute, and hour that the search actually runs on:
* * * *  Every minute.
*/M * * * *  Every M minutes (M > 0).
0 * * * *  Every hour.
0 */H * * *  Every H hours (H > 0).
0 0 * * *  Every day (at midnight).
* When set to non-zero for a search that has any other cron_schedule setting,
the search scheduler can only randomly "skew" the second that the search runs
on.
* The amount of skew for a specific search remains constant between edits of
the search.
* An integer value followed by '%' (percent) specifies the maximum amount of
time to skew as a percentage of the scheduled search period.
* Otherwise, use <int><unit> to specify a maximum duration. Relevant units
are: m, min, minute, mins, minutes, h, hr, hour, hrs, hours, d, day, days.
(The <unit> may be omitted only when <int> is 0.)
* Examples:
  100% (for an every-5-minute search) = 5 minutes maximum
  50% (for an every-minute search) = 30 seconds maximum
  5m = 5 minutes maximum
  1h = 1 hour maximum
* A value of 0 disallows skew.
* Default is 0.

acceleration.schedule_priority = default | higher | highest
* Raises the scheduling priority of a search:
  + "default": No scheduling priority increase.
  + "higher": Scheduling priority is higher than other data model searches.
  + "highest": Scheduling priority is higher than other searches regardless of
    scheduling tier except real-time-scheduled searches with priority = highest
    always have priority over all other searches.
  + Hence, the high-to-low order (where RTSS = real-time-scheduled search, CSS =
    continuous-scheduled search, DMAS = data-model-accelerated search, d =
    default, h = higher, H = highest) is:
    RTSS(H) > DMAS(H) > CSS(H)
    > RTSS(h) > RTSS(d) > CSS(h) > CSS(d)
    > DMAS(h) > DMAS(d)
* The scheduler honors a non-default priority only when the search owner has
  the 'edit_search_schedule_priority' capability.
* Defaults to: default
* WARNING: Having too many searches with a non-default priority will
impede the ability of the scheduler to minimize search starvation. Use this setting only for mission-critical searches.

acceleration.hunk.compression_codec = <string>
* Applicable only to Hunk Data models. Specifies the compression codec to be used for the accelerated orc/parquet files.

acceleration.hunk.dfs_block_size = <unsigned int>
* Applicable only to Hunk data models. Specifies the block size in bytes for the compression files.

acceleration.hunk.file_format = <string>
* Applicable only to Hunk data models. Valid options are "orc" and "parquet"

********** Dataset Related Attributes ******
# These attributes affect your interactions with datasets in Splunk Web and should not be changed under normal conditions. Do not modify them unless you are sure you know what you are doing.

dataset.description = <string>
* User-entered description of the dataset entity.

dataset.type = [datamodel|table]
* The type of dataset:
  + "datamodel": An individual data model dataset.
  + "table": A special root data model dataset with a search where the dataset is defined by the dataset.commands attribute.
* Default: datamodel

dataset.commands = [<object>(, <object>)*]
* When the dataset.type = "table" this stringified JSON payload is created by the table editor and defines the dataset.

dataset.fields = [<string>(, <string>)*]
* Automatically generated JSON payload when dataset.type = "table" and the root data model dataset's search is updated.

dataset.display.diversity = [latest|random|diverse|rare]
* The user-selected diversity for previewing events contained by the dataset:
  + "latest": search a subset of the latest events
+ "random": search a random sampling of events
+ "diverse": search a diverse sampling of events
+ "rare": search a rare sampling of events based on clustering
* Default: latest

dataset.display.sample_ratio = <int>
* The integer value used to calculate the sample ratio for the dataset
diversity.
   The formula is 1 / <int>.
* The sample ratio specifies the likelihood of any event being includedin the
sample.
* For example, if sample_ratio = 500 each event has a 1/500 chance ofbeing
   included in the sample result set.
* Default: 1

dataset.display.limiting = <int>
* The limit of events to search over when previewing the dataset.
* Default: 100000

dataset.display.currentCommand = <int>
* The currently selected command the user is on while editing the
dataset.

dataset.display.mode = [table|datasummary]
* The type of preview to use when editing the dataset:
   + "table": show individual events/results as rows.
   + "datasummary": show field values as columns.
* Default: table

dataset.display.datasummary.earliestTime = <time-str>
* The earliest time used for the search that powers the datasummary view of
   the dataset.

dataset.display.datasummary.latestTime = <time-str>
* The latest time used for the search that powers the datasummary view of
   the dataset.

tags_whitelist = <list-of-tags>
* A comma-separated list of tag fields that the data model requires
   for its search result sets.
* This is a search performance setting. Apply it only to data models
   that use a significant number of tag field attributes in their
definitions. Data models without tag fields cannot use this setting.
   This setting does not recognize tags used in constraint searches.
* Only the tag fields identified by tag_whitelist (and the event types
   tagged by them) are loaded when searches are performed with this
data model.
* When you update tags_whitelist for an accelerated data model,
the Splunk software rebuilds the data model unless you have enabled acceleration.manual_rebuild for it.
* If tags_whitelist is empty, the Splunk software attempts to optimize out unnecessary tag fields when searches are performed with this data model.
* Defaults to empty.

**datamodules.conf.example**

```bash
#   Version 7.2.0
#
# Configuration for example datamodels
#
# An example of accelerating data for the 'mymodel' datamodel for the past five days, generating and checking the column stores every 10 minutes
[mymodel]
acceleration = true
acceleration.earliest_time = -5d
acceleration.poll_buckets_until_maxtime = true
acceleration.cron_schedule = */10 * * * *
acceleration.hunk.compression_codec = snappy
acceleration.hunk.dfs_block_size = 134217728
acceleration.hunk.file_format = orc
```

**deep_dive_drilldowns.conf**

The following are the spec and example files for deep_dive_drilldowns.conf.

**deep_dive_drilldowns.conf.spec**

```bash
# Copyright (C) 2005-2018 Splunk Inc. All Rights Reserved.
# This file contains all possible attribute/value pairs for configuring # drilldown options for deep dive lanes.
#
# A unique drilldown options is represented by a stanza in this file.
# The name of the stanza is the name that will appear in the UI.
# Default values are provided for most settings and are defined in
# the [default] stanza of the configuration file.
#
# Other more complex drilldown options are not defined in this file
# because they are only represented in the deep dive code and cannot
```
# be disabled.
#
# There is a deep_dive_drilldowns.conf in
# $SPLUNK_HOME/etc/apps/itsi/default.
# To set custom configurations, place a deep_dive_drilldowns.conf in
# $SPLUNK_HOME/etc/apps/itsi/local/. You must restart Splunk software
to
# enable configurations.
#
# To learn more about configuration files (including precedence) please
# see the documentation located at
# https://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
#   * You can also define global settings outside of any stanza, at the
top of
#     the file.
#   * Each conf file should have at most one default stanza. If there
are
#     multiple default stanzas, settings are combined. In the case of
#     multiple definitions of the same setting, the last definition in
the
#     file wins.
#   * If a setting is defined at both the global level and in a specific
#     stanza, the value in the specific stanza takes precedence.

<name>

* Each stanza represents a unique drilldown option. Use these settings
to
  configure properties for all types of drilldowns.

type = uri|search
* Represents whether this drilldown is meant to redirect to a new
  URI or open a Splunk search.
* Required.

replace_tokens = true|false
* Enables token replacement in the search string or URI.
* Optional.
* If "true", the search or URI is token replaced by properties of the
drilldown.
* Token replacement is similar to token replacement in simpleXML. Tokens
are
represented in tokenized strings as a sub-string key surrounded by
'\$'.
* For example, search=index=_internal | stats count | where
count>$value$
* The following tokens are available for replacement by default:
  * lane_title - the title of the lane
  * lane_subtitle - the subtitle of the lane
  * lane_search - the search that powered the primary graph in the lane
  * earliest - the earliest epoch time stamp of the entire lane
  * latest - the latest epoch time stamp of the entire lane
  * bucket_earliest - the earliest epoch time stamp of the time bucket
  * bucket_latest - the latest epoch time stamp of the time bucket
* The following tokens are available for KPI lanes only:
  * kpi.service_id - the ID of the service to which the KPI belongs
  * kpi.service_title - the title of the service to which the KPI belongs
  * kpi.kpi_id - the ID of the KPI represented in the lane
  * kpi.kpi_title - the title of the KPI represented in the lane
  * kpi.single_value_search - the raw data alert search for the KPI
  * kpi.timeseries_search - the raw data time series search for the KPI
  * kpi.base_search - the event gathering/filtering search for the KPI
  * Default: false
metric_lane_enabled = true|false
* Whether to enable drilldowns on metric lanes.
* Optional.
  * If "true", drilldown is available on metric lanes.
  * If "false", drilldown is unavailable on metric lanes.
  * Default: false
kpi_lane_enabled = true|false
* Whether to enable drilldowns on KPI lanes.
* Optional.
  * If "true", drilldown is available on KPI lanes.
  * If "false", drilldown is unavailable on KPI lanes.
  * Default: false
event_lane_enabled = true|false
* Whether to enable drilldowns on event lanes.
* Optional.
  * If "true", drilldown is available on event lanes.
  * If "false", drilldown is unavailable on event lanes.
  * Default: false

**Entity-based features**

# Entity-based features are only available on KPI lanes because KPI
lanes are the only
# lanes that understand entities. Note that KPIs must have 'Split by Entity' enabled.

entity_level_only = true|false
* Whether to enable drilldowns only on lanes that surface entity-level information.
* Optional.
* If "true", drilldown is only available on lanes that surface entity-level information.
* If "false", drilldown is available on all lanes.
* Entity-level drilldowns make additional tokens and information available based
  on the entities clicked. See the 'entity_tokens' setting for more details.
* Default: false

entity_tokens = <csv>
* A CSV file of entity attributes to include on a drilldown.
* Optional.
* Only defined entities will be available on entity-level drilldowns. Pseudo-entities are ignored.
* If the 'replace_tokens' setting is "true", this setting will generate additional token replacements.
* Attributes can be either info fields or aliases.
* If the 'uri_payload_type' setting is set to "json", these entity attributes
  are added to the JSON payload per entity.
* Tokens from the first entity are replaced. If there are multiple entities,
  they all appear in a JSON payload.
* Tokens have the format "entity.<attribute name>".
* If any entity tokens are set to "all" (required to make drilldown work),
  entity.id and entity.title will always be available as tokens.

entity_activation_rules = <JSON blob of entity rules>|all
* Determines which entities to consider for drilldown.
* Optional.
* If "all", all entities are considered valid for drilldown.
* If set to a JSON blob of entity rules, entities will be tested for compliance with those rules. If no entities match, drilldown will not be available. If some or all all entities match, only those matching will be passed down to drilldown.

Properties for search type drilldowns

search = <tokenized search string>
* The search to use in the new lane or on the search page.
* Required for search type drilldowns.
* If the 'replace_tokens' setting is "true", the search is token
add_lane_enabled = true|false
* Whether users can activate the drilldown as a search.
* Required for search type drilldowns
* If "true", users can activate the drilldown as a search.
* If "false", users cannot activate the drilldown as a search.
* Default: false

use_bucket_timerange = true|false
* Whether to use only the time range of the selected bucket
  when redirected to a Splunk search.
* Optional.
* If "true", the drilldown search uses only the time range from which
  the user clicked in the deep dive.
* If "false", the drilldown search uses the entire search timerange.
* Default: true

new_lane_settings = <tokenized JSON for lane settings properties>
* A tokenized JSON string that represents a model to use for new lanes.
* Required for search type drilldowns with the 'add_lane_enabled'
  setting set to "true".
* The "search" setting is overridden by the search property in this
  stanza.
* If the 'replace_tokens' setting is "true", the string is token
  replaced
  by properties from the drilldown itself.
* Default lane settings are applied if you do not specify any values.

**Properties for URI type drilldowns**

uri = <str>
* The URI to redirect to on the drilldown.
* Required for URI type drilldowns.
* If the 'replace_tokens' setting is "true" and the 'uri_payload_type'
  setting is "simple", the URI string is replaced by tokens.
* Follows the format of an href:
  * A leading protocol allows a change in domain.
  * A leading slash changes the full path on the same domain.
  * Any other string only replaces the last segment of the URI with that
    string.

uri_payload_type = simple|json
* If "simple", token replacement is performed on the URI as if it were a
  search.
* If "json", no token replacement is performed and a query string
  parameter
  'drilldown_payload' is appended to the URI with a JSON representation
of the context of a drilldown. This payload will always contain the context portion of the JSON blob, which contains the basic properties. If it is entity level and the entity properties of the drilldown are specified, the entities portion will exist and consist of the entity ID and title as well as all attributes specified in as 'entity_tokens'. A JSON payload format will look like the following (assumes 'entity_tokens' was host,family):
{
    "context": {
        "earliest": <earliest time of full lane>,
        "latest": <latest time of full lane>,
        "bucket_earliest": <earliest time of bucket clicked>,
        "bucket_latest": <latest time of the bucket clicked>,
        "return_url": <URI of the current deep dive>,
        "service_id": "158bdaf4-6b0c-433e-9c24-c3a36c0e8eea",
        "kpi_id": "65ec30c5e1dd5046ac5416f5",
        "service_title": "Production Webservers",
        "kpi_title": "Total Request Latency (ms)"
    },
    "entities": [
    {
        "id": "5303377f-162c-45cc-809a-d1e3254ea4a1",
        "title": "Host Title 1",
        "host": "Host1",
        "family": "Linux"
    },
    {
        "id": "7aefd044-0f46-4ba4-ab13-f31e5797a3bf",
        "title": "Host Title 2",
        "host": "Host2",
        "family": "Linux"
    }
    ]
},

* Default: simple

deep_dive_drilldowns.conf.example

# This is an example deep_dive_drilldowns.conf. Use this file to configure custom drilldowns.
# To use one or more of these configurations, copy the configuration block
# into deep_dive_drilldowns.conf in $SPLUNK_HOME/etc/apps/itsi/local.
# You must restart Splunk to enable configurations.

300
# To learn more about configuration files (including precedence) please see the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

# This example alert includes showing raw events at selected time buckets,
# showing raw events from a displayed time range, showing KPI events for
# a host, and showing all events for a host.

[Show raw events at selected time bucket]
type = uri
kpi_lane_enabled = true
entity_level_only = false
uri = /app/itsi/search?q=search$kpi.base_search$&earliest=$bucket_earliest$&latest=$bucket_latest$&display.page.search.mode=smart&dispatch.sample_ratio=1
replace_tokens = false

[Show raw events from displayed time range]
type = uri
kpi_lane_enabled = true
entity_level_only = false
uri = /app/itsi/search?q=search$kpi.base_search$&earliest=$earliest$&latest=$latest$&display.page.search.mode=smart&dispatch.sample_ratio=1
replace_tokens = false

[Show kpi events for this host]
type = uri
kpi_lane_enabled = true
entity_level_only = true
replace_tokens = true
entity_tokens = host
uri = /app/itsi/search?q=search$kpi.base_search$ AND host=$entity.host$&earliest=$earliest$&latest=$latest$&display.page.search.mode=smart&dispatch.sample_ratio=1
entity_activation_rules = [ 
  { 
    "rule_condition": "AND", 
    "rule_items": [ 
      { 
        "field": "host", 
        "field_type": "alias", 
        "rule_type": "not", 
        "value": "" 
      } 
    ] 
  } 
]

[Show ALL events for this host]
distsearch.conf

The following are the spec and example files for distsearch.conf.

distsearch.conf.spec

#   Version 7.2.0
#
# This file contains possible attributes and values you can use to configure
# distributed search.
#
# To set custom configurations, place a distsearch.conf in
# $SPLUNK_HOME/etc/system/local/. For examples, see
# distsearch.conf.example.
# You must restart Splunk to enable configurations.
# To learn more about configuration files (including precedence) please
# see the
# documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
#
These attributes are all configured on the search head, with the exception of
the optional attributes listed under the SEARCH HEAD BUNDLE MOUNTING OPTIONS
heading, which are configured on the search peers.

GLOBAL SETTINGS

Use the [default] stanza to define any global settings.
* You can also define global settings outside of any stanza, at the
top of
the file.
* Each conf file should have at most one default stanza. If there
are
multiple default stanzas, attributes are combined. In the case of
multiple definitions of the same attribute, the last definition in
the
file wins.
* If an attribute is defined at both the global level and in a
specific
stanza, the value in the specific stanza takes precedence.

[distributedSearch]
* Set distributed search configuration options under this stanza name.
* Follow this stanza name with any number of the following
attribute/value
pairs.
* If you do not set any attribute, Splunk uses the default value (if
there
is one listed).

disabled = [true|false]
* Toggle distributed search off (true) and on (false).
* Defaults to false (your distributed search stanza is enabled by
default).

heartbeatMcastAddr = <IP address>
* This setting is deprecated

heartbeatPort = <port>
* This setting is deprecated

ttl = <integer>
* This setting is deprecated

heartbeatFrequency = <int, in seconds>
* This setting is deprecated
statusTimeout = <int, in seconds>  
* Set connection timeout when gathering a search peer's basic  
  info (/services/server/info).  
* Note: Read/write timeouts are automatically set to twice this value.  
  * Defaults to 10.

removedTimedOutServers = [true|false]  
* This setting is no longer supported, and will be ignored.

checkTimedOutServersFrequency = <integer, in seconds>  
* This setting is no longer supported, and will be ignored.

autoAddServers = [true|false]  
* This setting is deprecated

bestEffortSearch = [true|false]  
* Whether to remove a peer from search when it does not have any of our  
  bundles.  
* If set to true searches will never block on bundle replication, even  
  when a  
  peer is first added - the peers that don't have any common bundles  
  will  
  simply not be searched.  
* Defaults to false

skipOurselves = [true|false]  
* This setting is deprecated

servers = <comma separated list of servers>  
* Initial list of servers.  
* Each member of this list must be a valid uri in the format of  
  scheme://hostname:port

disabled_servers = <comma separated list of servers>  
* A list of disabled search peers. Peers in this list are not monitored  
  or searched.  
* Each member of this list must be a valid uri in the format of  
  scheme://hostname:port

quarantined_servers = <comma separated list of servers>  
* A list of quarantined search peers.  
* Each member of this list must be a valid uri in the format of  
  scheme://hostname:port  
* The admin may quarantine peers that seem unhealthy and are degrading  
  search  
  performance of the whole deployment.  
* Quarantined peers are monitored but not searched by default.  
* A user may use the splunk_server arguments to target a search to  
  quarantined peers  
  at the risk of slowing the search.  
* When a peer is quarantined, running realtime searches will NOT be  
  restarted. Running

304
realtime searches will continue to return results from the quarantined peers. Any
realtime searches started after the peer has been quarantined will not contact the peer.
* Whenever a quarantined peer is excluded from search, appropriate warnings will be displayed
  in the search.log and Job Inspector

useDisabledListAsBlacklist = <boolean>
* Whether or not the search head treats the ?disabled_servers? setting as a blacklist.
* If set to ?true?, search peers that appear in both the ?servers? and
  ?disabled_servers? lists are disabled and do not participate in search.
* If set to ?false?, search peers that appear in both lists are treated as enabled, despite
  being in the ?disabled_servers? list. These search peers do participate in search.
* Default: false

shareBundles = [true|false]
* Indicates whether this server will use bundle replication to share search
time configuration with search peers.
* If set to false, the search head assumes that all the search peers can access
  the correct bundles via share storage and have configured the options listed
  under "SEARCH HEAD BUNDLE MOUNTING OPTIONS".
* Defaults to true.

useSHPBundleReplication = <bool>|always
* Relevant only in search head pooling environments. Whether the search heads
  in the pool should compete with each other to decide which one should handle
  the bundle replication (every time bundle replication needs to happen)
  or whether each of them should individually replicate the bundles.
* When set to always and bundle mounting is being used then use the search head
  pool guid rather than each individual server name to identify bundles (and
  search heads to the remote peers).
* Defaults to true

trySSLFirst = <bool>
* This setting is no longer supported, and will be ignored.

peerResolutionThreads = <int>
* This setting is no longer supported, and will be ignored.
defaultUriScheme = [http|https]
* When a new peer is added without specifying a scheme for the uri to its management port we will use this scheme by default.
* Defaults to https

serverTimeout = <int, in seconds>
* REMOVED, this setting is now ignored and has been replaced by connectionTimeout, sendTimeout, receiveTimeout

connectionTimeout = <int, in seconds>
* Amount of time in seconds to use as a timeout during search peer connection establishment.

sendTimeout = <int, in seconds>
* Amount of time in seconds to use as a timeout while trying to write/send data to a search peer.

receiveTimeout = <int, in seconds>
* Amount of time in seconds to use as a timeout while trying to read/receive data from a search peer.

authTokenConnectionTimeout = <number, in seconds>
* Maximum number of seconds to connect to a remote search peer, when getting its auth token
* Fractional seconds are allowed
* Default is 5

authTokenSendTimeout = <number, in seconds>
* Maximum number of seconds to send a request to the remote peer, when getting its auth token
* Fractional seconds are allowed
* Default is 10

authTokenReceiveTimeout = <number, in seconds>
* Maximum number of seconds to receive a response from a remote peer, when getting its auth token
* Fractional seconds are allowed
* Default is 10

DISTRIBUTED SEARCH KEY PAIR GENERATION OPTIONS
[tokenExchKeys]

certDir = <directory>
* This directory contains the local Splunk instance's distributed search key
  pair.
* This directory also contains the public keys of servers that distribute
  searches to this Splunk instance.

publicKey = <filename>
* Name of public key file for this Splunk instance.

privateKey = <filename>
* Name of private key file for this Splunk instance.

genKeyScript = <command>
* Command used to generate the two files above.

REPLICATION SETTING OPTIONS

[replicationSettings]

connectionTimeout = <int, in seconds>
* The maximum number of seconds to wait before timing out on initial
  connection
to a peer.

sendRcvTimeout = <int, in seconds>
* The maximum number of seconds to wait for the sending of a full
  replication
to a peer.

replicationThreads = <positive int>|auto
* The maximum number of threads to use when performing bundle
  replication to peers.
* If you configure this setting to "auto", the peer autotunes the number
  of threads it uses for bundle replication.
** If the peer has less than 4 CPUs, it allocates 2 threads.
** If the peer has 4 or more, but less than 8 CPUs, it allocates up to
  '# of CPUs - 2' threads.
** If the peer has 8 or more, but less than 16 CPUs, it allocates up to
  '# of CPUs - 3' threads.
** If the peer has 16 or more CPUs, it allocates up to '# of CPUs - 4'
  threads.
* Defaults to 5.
maxMemoryBundleSize = <int>
* The maximum size (in MB) of bundles to hold in memory. If the bundle is
  larger than this the bundles will be read and encoded on the fly for each
  peer the replication is taking place.
* Defaults to 10

maxBundleSize = <int>
* The maximum size (in MB) of the bundle for which replication can occur. If
  the bundle is larger than this bundle replication will not occur and an
  error message will be logged.
* Defaults to: 2048 (2GB)

cconcerningReplicatedFileSize = <int>
* Any individual file within a bundle that is larger than this value (in MB)
  will trigger a splunkd.log message.
* Where possible, avoid replicating such files, e.g. by customizing your
  blacklists.
* Defaults to: 500

excludeReplicatedLookupSize = <int>
* Any lookup file larger than this value (in MB) will be excluded from the
  knowledge bundle that the search head replicates to its search peers.
* When this value is set to 0, this feature is disabled.
* Defaults to 0

allowStreamUpload = auto | true | false
* Whether to enable streaming bundle replication for peers.
* If set to auto, streaming bundle replication will be used when connecting to
  peers with a complete implementation of this feature (Splunk 6.0 or higher).
* If set to true, streaming bundle replication will be used when connecting to
  peers with a complete or experimental implementation of this feature
  (Splunk 4.2.3 or higher).
* If set to false, streaming bundle replication will never be used.
  Whatever the value of this setting, streaming bundle replication will not
  be used for peers that completely lack support for this feature.
* Defaults to: auto

allowSkipEncoding = <bool>
* Whether to avoid URL-encoding bundle data on upload.
* Defaults to: true
allowDeltaUpload = <bool>
* Whether to enable delta-based bundle replication.
* Defaults to: true

sanitizeMetaFiles = <bool>
* Whether to sanitize or filter *.meta files before replication.
* This feature can be used to avoid unnecessary replications triggered by
  writes to *.meta files that have no real effect on search behavior.
* The types of stanzas that "survive" filtering are configured via the
  replicationSettings:refineConf stanza.
* The filtering process removes comments and cosmetic whitespace.
* Defaults to: true

**RFS (AKA S3 / REMOTE FILE SYSTEM) REPLICATION SPECIFIC SETTINGS**

enableRFSReplication = <bool>
* Currently not supported. This setting is related to a feature that is
  still under development.
* Required on search heads.
* When search heads generate bundles, these bundles are uploaded to
  the configured remote file system.
* When search heads delete their old bundles, they subsequently
  attempt to delete the bundle from the configured remote file system.
* If set to true, remote file system bundle replication is enabled.
* Default: false.

enableRFSMonitoring = <bool>
* Currently not supported. This setting is related to a feature that is
  still under development.
* Required on search peers.
* Search peers periodically monitor the configured remote file system
  and download any bundles that they do not have on disk.
* If set to true, remote file system bundle monitoring is enabled.
* Default: false.

rfsMonitoringPeriod = <unsigned int>
* Currently not supported. This setting is related to a feature that is
  still under development.
* The amount of time, in seconds, that a search peer waits between
  polling attempts. You must also set this attribute on search heads, whether or
  not 'enableRFSMonitoring' is enabled on them.
* For search heads when 'rfsSyncReplicationTimeout' is set to 'auto',
  setting this will automatically adapt the 'rfsSyncReplicationTimeout'
  parameter to the monitoring frequency of the search peers.
* If you set this parameter to less than 60, it is automatically
  reset to 60.
rfsSyncReplicationTimeout = <unsigned int>
* Currently not supported. This setting is related to a feature that is
  still under development.
* The amount of time, in seconds, that a search head waits for
  synchronous
  replication to complete. Only applies to RFS bundle replication.
* Default value is computed from 'rfsMonitoringPeriod', i.e.
  \((rfsMonitoringPeriod + 60)) \times 5\), where 60 is the non-configurable
  search
  head to search peer polling interval, and 5 is arbitrary multiplier.
  If 'rfsMonitoringPeriod' is not modified, default value is 600.
* Default: auto.

path = <path on server>
* Currently not supported. This setting is related to a feature that is
  still under development.
* Required.
* The path attribute points to the remote storage location where bundles
  reside.
* The format for this attribute is:
  \(<\text{scheme}>://<\text{remote-location-specifier}>\)
  * The "scheme" identifies a supported external storage system type.
  * The "remote-location-specifier" is an external system-specific
    string
    for identifying a location inside the storage system.
* These external systems are supported:
  - Object stores that support AWS's S3 protocol. These use the scheme
    "s3".
    Example: "path=s3://mybucket/some/path"
  - POSIX file system, potentially a remote filesystem mounted over NFS.
    These use the scheme "file".
    Example: "path=file:///mnt/cheap-storage/some/path"

remote.s3.endpoint = <URL>
* Currently not supported. This setting is related to a feature that is
  still under development.
* The URL of the remote storage system supporting the S3 API.
* The protocol, http or https, can be used to enable or disable SSL
  connectivity with the endpoint.
* If not specified and the indexer is running on EC2, the endpoint will
  be
  constructed automatically based on the EC2 region of the instance
  where
  the indexer is running, as follows: https://s3-<region>.amazonaws.com
* Example: https://s3-us-west-2.amazonaws.com

remote.s3.encryption = sse-s3 | none
* Currently not supported. This setting is related to a feature that is
  still under development.
* Optional.
* Specifies the schema to use for Server-side Encryption (SSE) for data at rest.
  * none: Server-side encryption is disabled. Data is stored unencrypted on the remote storage.
  * Default: none

[replicationSettings:refineConf]

replicate.<conf_file_name> = <bool>
* Controls whether Splunk replicates a particular type of *.conf file, along with any associated permissions in *.meta files.
  * These settings on their own do not cause files to be replicated. A file must still be whitelisted (via replicationWhitelist) to be eligible for inclusion via these settings.
  * In a sense, these settings constitute another level of filtering that applies specifically to *.conf files and stanzas with *.meta files.
  * Defaults to: false

**REPLICATION WHITELIST OPTIONS**

[replicationWhitelist]

<name> = <whitelist_pattern>
* Controls Splunk's search-time conf replication from search heads to search nodes.
  * Only files that match a whitelist entry will be replicated.
  * Conversely, files which are not matched by any whitelist will not be replicated.
  * Only files located under $SPLUNK_HOME/etc will ever be replicated in this way.
  * The regex will be matched against the filename, relative to $SPLUNK_HOME/etc.
    Example: for a file "$SPLUNK_HOME/etc/apps/fancy_app/default/inputs.conf"
    this whitelist should match "apps/fancy_app/default/inputs.conf"
  * Similarly, the etc/system files are available as system/...
    user-specific files are available as users/username/appname/...
* The 'name' element is generally just descriptive, with one exception: if <name> begins with "refine.", files whitelisted by the given pattern will also go through another level of filtering configured in the replicationSettings:refineConf stanza.
* The whitelist_pattern is the Splunk-style pattern matching, which is primarily regex-based with special local behavior for '...' and '*'. ... matches anything, while * matches anything besides directory separators.
  * See props.conf.spec for more detail on these.
* Note '.' will match a literal dot, not any character.
* Note that these lists are applied globally across all conf data, not to any particular app, regardless of where they are defined. Be careful to pull in only your intended files.

**REPLICATION BLACKLIST OPTIONS**

[replicationBlacklist]

<name> = <blacklist_pattern>
* All comments from the replication whitelist notes above also apply here.
* Replication blacklist takes precedence over the whitelist, meaning that a file that matches both the whitelist and the blacklist will NOT be replicated.
* This can be used to prevent unwanted bundle replication in two common scenarios:
  * Very large files, which part of an app may not want to be replicated,
    especially if they are not needed on search nodes.
  * Frequently updated files (for example, some lookups) will trigger retransmission of all search head data.
* Note that these lists are applied globally across all conf data. Especially for blacklisting, be careful to constrain your blacklist to match only data your application will not need.

**BUNDLE ENFORCER WHITELIST OPTIONS**
[bundleEnforcerWhitelist]

<name> = <whitelist_pattern>
* Peers uses this to make sure knowledge bundle sent by search heads and masters do not contain alien files.
* If this stanza is empty, the receiver accepts the bundle unless it contains files matching the rules specified in [bundleEnforcerBlacklist].
  Hence, if both [bundleEnforcerWhitelist] and [bundleEnforcerBlacklist] are empty (which is the default), then the receiver accepts all bundles.
* If this stanza is not empty, the receiver accepts the bundle only if it contains only files that match the rules specified here but not those in [bundleEnforcerBlacklist].
* All rules are regexs.
* This stanza is empty by default.

BUNDLE ENFORCER BLACKLIST OPTIONS

[bundleEnforcerBlacklist]

<name> = <blacklist_pattern>
* Peers uses this to make sure knowledge bundle sent by search heads and masters do not contain alien files.
* This list overrides [bundleEnforcerWhitelist] above. That means the receiver rejects (i.e. removes) the bundle if it contains any file that matches the rules specified here even if that file is allowed by [bundleEnforcerWhitelist].
* If this stanza is empty, then only [bundleEnforcerWhitelist] matters.
* This stanza is empty by default.

SEARCH HEAD BUNDLE MOUNTING OPTIONS

# You set these attributes on the search peers only, and only if you also set # shareBundles=false in [distributedSearch] on the search head. Use them to
# achieve replication-less bundle access. The search peers use a shared storage
# mountpoint to access the search head bundles ($SPLUNK_HOME/etc).
#******************************************************************************

[searchhead:<searchhead-splunk-server-name>]
* <searchhead-splunk-server-name> is the name of the related searchhead installation.
* This setting is located in server.conf, serverName = <name>

mounted_bundles = [true|false]
* Determines whether the bundles belong to the search head specified in the
  stanza name are mounted.
* You must set this to "true" to use mounted bundles.
  Default is "false".

bundles_location = <path_to_bundles>
* The path to where the search head's bundles are mounted. This must be the
  mountpoint on the search peer, not on the search head. This should point to
  a directory that is equivalent to $SPLUNK_HOME/etc/. It must contain at least
  the following subdirectories: system, apps, users.

**DISTRIBUTED SEARCH GROUP DEFINITIONS**

# These are the definitions of the distributed search groups. A search group is
# a set of search peers as identified by thier host:management-port. A search
# may be directed to a search group using the splunk_server_group argument.
The # search will be dispatched to only the members of the group.
#******************************************************************************

[distributedSearch:<splunk-server-group-name>]
* <splunk-server-group-name> is the name of the splunk-server-group that is
defined in this stanza

servers = <comma separated list of servers>
* List of search peers that are members of this group. Comma separated list
  of peer identifiers i.e. hostname:port
default = [true|false]
* Will set this as the default group of peers against which all searches are run unless a server-group is not explicitly specified.

**distsearch.conf.example**

```
# Version 7.2.0
#
# These are example configurations for distsearch.conf. Use this file to configure distributed search. For all available attribute/value pairs, see
# distsearch.conf.spec.
#
# There is NO DEFAULT distsearch.conf.
#
# To use one or more of these configurations, copy the configuration block into
# distsearch.conf in $SPLUNK_HOME/etc/system/local/. You must restart Splunk
# to enable configurations.
#
# To learn more about configuration files (including precedence) please see the
# documentation located at
#
http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

[distributedSearch]

# This entry distributes searches to 192.168.1.1:8059,192.168.1.2:8059. These machines will be contacted on port 8059 using https
# Attributes not set here will use the defaults listed in distsearch.conf.spec.

# this stanza controls the timing settings for connecting to a remote peer and
# the send timeout
[replicationSettings]
connectionTimeout = 10
sendRcvTimeout = 60

# this stanza controls what files are replicated to the other peer each is a
# regex
[replicationWhitelist]
allConf = *.conf
```
# Mounted bundles example.
# This example shows two distsearch.conf configurations, one for the
# search
# head and another for each of the search head's search peers. It shows
# only
# the attributes necessary to implement mounted bundles.

# On a search head whose Splunk server name is "searcher01":
[distributedSearch]
...
shareBundles = false

# On each search peer:
[searchhead:searcher01]
mounted_bundles = true
bundles_location = /opt/shared_bundles/searcher01

---

drilldownsearch_offset.conf

The following are the spec and example files for drilldownsearch_offset.conf.

---

drilldownsearch_offset.conf.spec

# This file contains attributes and values for configuring time range
# picker
# presets for correlation search drilldown offsets.
#
# There is a drilldownsearch_offset.conf in
# $SPLUNK_HOME/etc/apps/itsi/default/.
# To set custom configurations, place a drilldownsearch_offset.conf in
# $SPLUNK_HOME/etc/apps/itsi/local/. You must restart Splunk to enable
# configurations.
# To learn more about configuration files (including precedence) please
see
# the documentation located at
#
http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

---

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
#  * You can also define global settings outside of any stanza, at the

316
# Each .conf file should have at most one default stanza. If there
# are multiple default stanzas, attributes are combined. In the case of
# multiple definitions of the same attribute, the last definition in
# the file wins.
# If an attribute is defined at both the global level and in a
# specific stanza, the value in the specific stanza takes precedence.

[<offset-period-number>]

timeInSecs = <integer>
* The offset time, in seconds.
* Required.

description = <string>
* The description that is shown in the UI for the earliest and latest offset
dropdown. The earliest offset prepends "Last" to the description and the
latest offset prepends "Next" to the description.
* Required if the 'earliest_description' and 'latest_description' settings
are not defined below.

earliest_description = <string>
* A description for the earliest offset dropdown.
* Optional.

latest_description = <string>
* A description for the latest offset dropdown.
* Optional.

**drilldownsearch_offset.conf.example**

No example

**fields.conf**

The following are the spec and example files for fields.conf.
GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the
# top of
#   the file.
# * Each conf file should have at most one default stanza. If there
# are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in
# the
#   file wins.
# * If an attribute is defined at both the global level and in a
# specific
#   stanza, the value in the specific stanza takes precedence.

[field name>]

* Name of the field you're configuring.
* Follow this stanza name with any number of the following
attribute/value pairs.
* Field names can only contain a-z, A-Z, 0-9, and _, but cannot begin with a number or _

# TOKENIZER indicates that your configured field's value is a smaller part of a token. For example, your field's value is "123" but it occurs as "foo123" in your event.
TOKENIZER = <regular expression>
* Use this setting to configure multivalue fields (refer to the online documentation for multivalue fields).
* A regular expression that indicates how the field can take on multiple values at the same time.
* If empty, the field can only take on a single value.
* Otherwise, the first group is taken from each match to form the set of values.
* This setting is used by the "search" and "where" commands, the summary and XML outputs of the asynchronous search API, and by the top, timeline and stats commands.
* Tokenization of indexed fields (INDEXED = true) is not supported so this attribute is ignored for indexed fields.
* Default to empty.

INDEXED = [true|false]
* Indicate whether a field is indexed or not.
* Set to true if the field is indexed.
* Set to false for fields extracted at search time (the majority of fields).
* Defaults to false.

INDEXED_VALUE = [true|false|<sed-cmd>|<simple-substitution-string>]
* Set this to true if the value is in the raw text of the event.
* Set this to false if the value is not in the raw text of the event.
* Setting this to true expands any search for key=value into a search of value AND key=value (since value is indexed).
* For advanced customization, this setting supports sed style substitution.
  For example, 'INDEXED_VALUE=s/foo/bar/g' would take the value of the field, replace all instances of 'foo' with 'bar,' and use that new value as the value to search in the index.
* This setting also supports a simple substitution based on looking for the literal string '<VALUE>' (including the '<' and '>' characters).
For example, 'INDEXED_VALUE=source::*<VALUE>*' would take a search for 'myfield=myvalue' and search for 'source::*myvalue*' in the index as a single term.
* For both substitution constructs, if the resulting string starts with a '!', Splunk interprets the string as a Splunk LISPY expression. For example, 'INDEXED_VALUE=[OR <VALUE> source::*<VALUE>]’ would turn 'myfield=myvalue' into applying the LISPY expression '[OR myvalue source::*myvalue]' meaning it matches either 'myvalue' or 'source::*myvalue' terms).
* Defaults to true.
* NOTE: You only need to set indexed_value if indexed = false.

fields.conf.example

# Version 7.2.0
#
# This file contains an example fields.conf. Use this file to configure dynamic field extractions.
#
# To use one or more of these configurations, copy the configuration block into fields.conf in $SPLUNK_HOME/etc/system/local/. You must restart Splunk to enable configurations.
# To learn more about configuration files (including precedence) please see the documentation located at http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
#
# These tokenizers result in the values of To, From and Cc treated as a list, where each list element is an email address found in the raw string of data.

[To]
TOKENIZER = (\w[\w\.-]*@[\w\.-]*\w)

[From]
TOKENIZER = (\w[\w\.-]*@[\w\.-]*\w)

[Cc]
TOKENIZER = (\w[\w\.-]*@[\w\.-]*\w)
glasstable_icon_library.conf

The following are the spec and example files for glasstable_icon_library.conf.

glasstable_icon_library.conf.spec

# This file contains possible attributes and values for adding
# and removing icons from the glass table icon library.
#
# There is a glasstable_icon_library.conf in
# $SPLUNK_HOME/etc/apps/itsi/default/.
# To set custom configurations, place a glasstable_icon_library.conf in
# $SPLUNK_HOME/etc/apps/itsi/local/. You must restart Splunk to enable
# configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
#
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the top
#   of the file.
# * Each .conf file should have at most one default stanza. If there are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in the
#   file wins.
# * If an attribute is defined at both the global level and in a
#   specific
#   stanza, the value in the specific stanza takes precedence.

[default]

iconThumbnailSrc = <string>
* The file path of the icon.
  * Required.

```
[<stanza name>]
```

iconId = <string>
* An internal unique identifier for the icon.
  * Required.

iconLabel = <string>
* The name or label for the icon that appears in the UI.

iconThumbnailSrc = <string>
* The file path of the icon.
  * Required.

iconCategory = ['Application'|'Splunk'|'Network'|'General']
* The assigned category for the icon.
  * Required.

svgPath = <string>
* The SVG path for the icon.
  * The same path used for the icon library thumbnail.

defaultWidth = <positive integer>
* The initial width of the icon.

defaultHeight = <positive integer>
* The initial height of the icon.

glasstable_icon_library.conf.example

No example

inputs.conf

The following are the spec and example files for inputs.conf.

inputs.conf.spec

# This file contains possible settings you can use to configure ITSI inputs, register
# user access roles, and import services and entities from CSV files or
search strings.
#
# There is an inputs.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/default. To set
# configurations, place an inputs.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/local.
# You must restart ITSI to enable new configurations.
#
# To learn more about configuration files (including precedence), see
# the
# documentation located at
#
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the
#   top of
#   the file.
# * Each conf file should have at most one default stanza. If there
#   are
#   multiple default stanzas, settings are combined. In the case of
#   multiple definitions of the same setting, the last definition in
#   the
#   file wins.
# * If a setting is defined at both the global level and in a specific
#   stanza, the value in the specific stanza takes precedence.

# log_level = <DEBUG|INFO|WARN|ERROR>
# * This setting sets the logging level of each modular input.
# * Logging levels are in order of most to least verbose.
# * The logging level describes the type and/or quantity of output
#   that an application writes to a log file.
# * Set the logging verbosity of each modular input to specify how
#   much and what kind of information it writes to the log file.
# * Setting a log level gets you messages at that level and higher,
#   so default settings are typically INFO or WARN.

[itsi_user_access_init://<name>]

* A modular input that runs once during startup (or at the user's request)
  to register user access roles and capabilities with the SA-UserAccess
  module.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: WARN

app_name = <name>
* The Splunk application that has the user access roles and capabilities.
* Default: itsi

registered_capabilities = [true|false]
* Indicates whether or not capabilities have already been registered with ITSI.
* If true, the 'itsi_user_access_init' input does not re-register capabilities.
* If false, 'itsi_user_access_init' registers ITSI capabilities again.
* Default: false

[configure_itsi://<name>]

* A configuration input that runs once (or at the user's request) to pull entities from the configuration file system into the App Key Value (KV) Store.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: WARN

is_configured = ""
* Left it for backwards compatibility.

[itsci_csv_import://<string>]

* A modular input that periodically uploads CSV data into the KV Store.
* The CSV file must contain headers for the import to work properly.
* This input runs every 4 hours or after a Splunk software restart.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: WARN

import_from_search = <boolean>
* Indicates whether to import data from a CSV file or a Splunk search.
* If ?true?, this input imports data from the search specified by 'search_string'.
* If ?false?, this input imports CSV data from the path specified by 'csv_location'.
* This setting is required, and the input does not run if the setting is not present.
* There is no default.
The location on disk of the CSV file to import.
* NOTE: The disk must be local to the search head. Cloud storage is unacceptable.
* This setting is required if you import data from a CSV file
  (if you set 'import_from_search' to "false").
* There is no default.

The Splunk search string that generates the data to import.
* This setting is required if you import from a search string
  (if you set 'import_from_search' to "true").
* There is no default.

The ITSI team that the imported services belong to.
* Use teams to group services by department, organization, or type of service and control access to the services.
* This setting is required, and the input does not run if the setting is not present.
* There is no default.

Specify the earliest _indextime, in minutes, for the time range of your search.
* This setting is required if you import from a search string
  (if you set 'import_from_search' to "true").
* Default: -15m

Specify the latest _indextime, in minutes, for the time range of your search.
* This setting is required if you import from a search string
  (if you set 'import_from_search' to "true").
* Default: now

The column name in the CSV file, or the field in the search, to import the entity title from.
* This field serves as the informal identifier of the entity.
* There is no default.

A dictionary of key:value pairs that specifies how 'entity_title_field' associates with other fields and in what relationship.
* NOTE: This setting is unused.
* For example,
  "hosts": ["vm1", "vm2"], "hostedBy": "host_id"

For a record that has values for fields: vm1, vm2, host_id,
<'entity_title_field' value>, three relationships are extracted:
<value for 'entity_title_field'> hosts <value for vm1>
<value for 'entity_title_field'> hosts <value for vm2>
<value for 'entity_title_field'> hostedBy <value for host_id>
* There is no default.

selected_services = <comma-separated list>
* A list of existing services to associate the imported entities with.
* DEPRECATED.
* There is no default.

service_rel = <comma-separated list>
* A list of existing service relationships.
* DEPRECATED.
* Use this setting to represent service dependencies in ITSI.
* There is no default.

service_dependents = <comma-separated list>
* A list of child columns in the CSV file, or child fields in the search,
  that indicate service dependencies.
* There is no default.

entity_service_columns = <comma-separated list>
* A list of services found in the CSV file or search that are to be associated with the entity for the row.
* DEPRECATED.
* There is no default.

entity_identifier_fields = <comma-separated list>
* A list of columns found in the CSV file or fields in the search that identify the entities (entity aliases).
* There is no default.

entity_description_column = <comma-separated list>
* A list of columns found in the CSV file or fields in the search that describe the entities.
* There is no default.

entity_informational_fields = <comma-separated list>
* A list of informational columns in the CSV file or fields in the search.
* These are non-identifying fields for the entities.
* There is no default.

entity_field_mapping = <key-value pairs>
* A key-value mapping of fields to re-map to other fields in your data.
* Follows a <CSV field> = <Splunk search field> format.
* For example, ip1 = dest, ip2 = dest, storage_type = volume
* Use this setting to rename a field or column to an alias or info value.
* There is no default.

326
service_title_field = <string>
* The field to import the service title from.
* This field is the informal identifier of the service.
* There is no default.
* This setting is required if you import services.

service_description_column = <comma-separated list>
* A list of columns in the CSV file or fields in the search
  that describe the services.
* There is no default.

service_enabled = <boolean>
* Whether or not imported services are enabled.
* Default: false

service_template_field = <string>
* This setting determines which service template a service is linked to.
* There is no default.

template = <dict>
* A dictionary of key:value pairs that maps entity rules to service
  templates.
* For example,
  
  ```
  {
  "test_template_2":{"entity_rules":[
  {"rule_items":
  [{"rule_type":"matches","field_type":"alias","field":"whoa","value":"doe"}],
  "rule_condition":"AND"}],
  "test_template_1":{"entity_rules":[
  {"rule_items":
  [{"rule_type":"matches","field_type":"alias","field":"blah","value":"da"}],
  "rule_condition":"AND"}]}}
  ```
* CAUTION: Do not change this setting.
* There is no default.

backfill_enabled = <boolean>
* This setting determines whether to enable backfill on all
  Key Performance Indicators (KPIs) in linked service templates.
* Backfill is the process of getting historical KPI data.
* ITSI backfills the KPI summary index (itsi_summary). You must have
  indexed adequate raw data for the backfill period.
* There is no default.

update_type = <APPEND|UPSERT|REPLACE>
* The update/insertion method when uploading entities.
* This setting is required, and the input will not run if the setting is
  not present.
* APPEND: ITSI makes no attempt to identify commonalities between
  entities.
  All information is appended to the table.
* UPSERT: ITSI appends new entries. Existing entries (based on the
  value
  found in the title_field) have additional information appended
  to the existing record.
* REPLACE: ITSI appends new entries. Existing entries (based on the
value
  found in the title_field) are replaced by the new record value.
* There is no default.

interval = <integer>
* The interval, in seconds, that determines how often this input runs.
* There is no default.

[itsi_async_csv_loader://<name>]

* A modular input that periodically uploads CSV data into the KV store.
* The file must contain headers for the import to work properly.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: WARN

import_from_search = <boolean>
* Indicates whether to import data from a CSV file or a Splunk search.
* If ?true?, this input imports data from the search specified by
  'search_string'.
* If ?false?, this input imports CSV data from the path specified by
  'csv_location'.
* This setting is required, and the input does not run if the setting is
  not present.
* There is no default.

csv_location = <path>
* The location on disk of the CSV file to import.
* NOTE: The disk must be local to the search head. Cloud storage is
  unacceptable.
* This setting is required if you import data from a CSV file
  (if you set 'import_from_search' to "false").
* There is no default.

search_string = <string>
* The Splunk search string that generates the data to import.
* This setting is required if you import from a search string
  (if you set 'import_from_search' to "true").
* There is no default.

index_earliest = <integer>
* Specify the earliest _indextime, in minutes, for the time range of
  your search.
* This setting is required if you import from a search string
  (if you set 'import_from_search' to "true").
* Default: -15m

index_latest = <integer>
* Specify the latest _indextime, in minutes, for the time range of your
search.
* This setting is required if you import from a search string
  (if you set 'import_from_search' to "true").
* Default: now

entity_title_field = <string>
* The column name in the CSV file, or the field in the search, to import
  the entity title from.
* This field serves as the informal identifier of the entity.
* There is no default.

entity_relationship_spec = <dict>
* A dictionary of key:value pairs that specifies how
  'entity_title_field' associates with other fields and in what
  relationship.
* NOTE: This setting is unused.
* For example,
  
  {
    "hosts": ["vm1", "vm2"],
    "hostedBy": "host_id"}

* For a record that has values for fields: vm1, vm2, host_id,
  
  <value for 'entity_title_field'>
  hosts <value for vm1>

  <value for 'entity_title_field'>
  hosts <value for vm2>

  <value for 'entity_title_field'>
  hostedBy <value for host_id>
* There is no default.

selected_services = <comma-separated list>
* A list of existing services to associate the imported entities with.
  * DEPRECATED.
  * There is no default.

service_rel = <comma-separated list>
* A list of existing service relationships.
  * DEPRECATED.
  * Use this setting to represent service dependencies in ITSI.
  * There is no default.

service_dependents = <comma-separated list>
* A list of child columns in the CSV file, or child fields in the
  search,
  that indicate service dependencies.
* There is no default.

entity_service_columns = <comma-separated list>
* A list of services found in the CSV file or search that are to be
  associated with the entity for the row.
  * DEPRECATED.
  * There is no default.

entity_identifier_fields = <comma-separated list>
* A list of columns found in the CSV file or fields in the search
  that identify the entities (entity aliases).
entity_description_column = <comma-separated list>
* A list of columns found in the CSV file or fields in the search
  that describe the entities.
* There is no default.

entity_informational_fields = <comma-separated list>
* A list of informational columns in the CSV file or fields in the
  search.
* These are non-identifying fields for the entities.
* There is no default.

entity_field_mapping = <key-value pairs>
* A key-value mapping of fields to re-map to other fields in your data.
* Follows a <CSV field> = <Splunk search field> format.
* For example, ip1 = dest, ip2 = dest, storage_type = volume
* Use this setting to rename a field or column to an alias or info
  value.
* There is no default.

service_title_field = <string>
* The field to import the service title from.
* This field is the informal identifier of the service.
* There is no default.
* This setting is required if you import services.

service_description_column = <comma-separated list>
* A list of columns in the CSV file or fields in the search
  that describe the services.
* There is no default.

update_type = <APPEND|UPSERT|REPLACE>
* The update/insertion method when uploading entities.
* This setting is required, and the input will not run if the setting is
  not present.
* APPEND: ITSI makes no attempt to identify commonalities between
  entities.
  All information is appended to the table.
* UPSERT: ITSI appends new entries. Existing entries (based on the
  value
  found in the title_field) have additional information appended
  to the existing record.
* REPLACE: ITSI appends new entries. Existing entries (based on the
  value
  found in the title_field) are replaced by the new record value.
* There is no default.
[itsi_upgrade://<name>]

* A modular input that checks the ITSI version when Splunk Enterprise starts.
* If the version does not match, Splunk Enterprise locks the UI and performs a version-based migration.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: DEBUG

[itsi_refresher://<name>]

* A modular input that processes deferred methods using a single queue processor.
* Tracks relational objects and dependencies.
* This input detects conflicts and ensures consistency across ITSI.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO

[itsi_consumer://<name>]

* A modular input that processes deferred methods using multiple queues across the Splunk environment.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO

[itsi_backup_restore://<name>]

* A modular input that performs backup and restore operations by managing backup/restore jobs.
* If you restore ITSI from a backup of an older version of ITSI, migration begins during the restore process.
* The input runs runs every 5 seconds to check for the scheduled job.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO
[itsi_scheduled_backup_caller://<name>]

* A modular input that manages ITSI backup schedules.
* For example, you might use this input if you want to backup ITSI every night at 1 am.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO

[itsi_service_template_update_scheduler://<name>]

* A modular input that performs a scheduled sync from service templates to services every 15 minutes.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO

[itsi_backfill://<name>]

* A modular input that manages KPI backfill jobs.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO

[itsi_notable_event_archive://<name>]

* A modular input that moves notable events from the KV store to the index every hour.
* Splunk cannot read the modular name unless a parameter is specified. Therefore, ITSI passes 'owner = <string>'.

[maintenance_minder://<name>]

* A modular input that runs every 60 seconds and populates the operative maintenance log based on configured maintenance windows. This input is responsible for putting services into maintenance mode.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO
**[itsi_default_aggregation_policy_loader://<name>]**

* A modular input that loads the default aggregation policy.
* The default aggregation policy receives notable events that do not match the filtering criteria of any other aggregation policies.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO

**[itsi_default_correlation_search_acl_loader://<name>]**

* A modular input that loads the Access Control List (ACL) for the default correlation searches provided with ITSI: "Monitor Critical Service Based on HealthScore", "Splunk App for Infrastructure Alerts", and "Normalized Correlation Search".
* This input pulls ACL information from the KV store.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO

**[itsi_notable_event_hec_init://<name>]**

* A modular input that initializes HEC client on a search head by creating and showing pertinent HEC tokens.
* A new HEC token is acquired during a Splunk restart.
* The internal system populates the new HEC token automatically.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO

**[itsi_notable_event_actions_queue_consumer://name]**

* A modular input that acts as a consumer of the queue for executing notable event actions, such as pinging a host or running a script.
* This setting is primarily used by the rules engine.

exec_delay_time = <integer>
* The amount of time, in seconds, to delay execution of a notable event action.
* Default: 0
batch_size = <integer>
* The number of jobs to pick up in a single request from the
  notable event actions queue.
* Default: 5

timeout = <integer>
* The timeout period, in seconds, that ITSI uses when a
  user reclaims an expired job.
* Default: 7200 (2 hours)

[itsi_entity_exchange_consumer://name]

* A modular input that consumes entities from the entity exchange
  module.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of the modular input.
* Default: DEBUG

interval = <value>
* The interval, in seconds, at which the modular input should run.
* Optional
* Default: 300 (5 minutes)

[itsi_age_kpi_alert_value_cache://<name>]

* A modular input that cleans up the aged entries in the KPI summary
  cache.

retentionTimeInSec = <integer>
* Aging/retention time for entries present in the KPI summary cache.

log_level = <DEBUG|INFO|WARN|ERROR>
* The logging level of this input.
* Default: INFO

inputs.conf.example

No example
itsi_da.conf

The following are the spec and example files for itsi_da.conf.

itsi_da.conf.spec

# Copyright (C) 2005-2018 Splunk Inc. All Rights Reserved.
#
# This configuration file is DEPRECATED.
#
# For [entity_source_template://<string>], use
# inputs.conf/[itsi_csv_import://<name>] instead.
# For [service_template://<string>], use
# itsi_service_template.conf/[string] instead.
#
# This file contains possible settings you can use to configure an
# itsi_da.conf file. Use this
# file to configure an app to export entity searches and service
# templates for use within the
# IT Service Intelligence (ITSI) app.
#
# To learn more about configuration files (including precedence), see
# the
# documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
#
# CAUTION: You can drastically affect your Splunk installation by
# changing these settings.
# Consult technical support (http://www.splunk.com/page/submitIssue)
# if you are not sure how
# to configure this file.

[entity_source_template://<string>]

title = <string>
* The display name of the search.

description = <string>
* A human-readable description of this search.

saved_search = <string>
* The actual Splunk saved search that outputs a table. This will be
  enforced by
    client-side code.

title_field = <string>
* A single field that acts as the title for the entity.

description_fields = <comma-separated list>
* A list of fields that describe the entity.

identifier_fields = <comma-separated list>
* A list of fields that identify the entity.

informational_fields = <comma-separated list>
* A list of fields that act as additional entity metadata.

[service_template://<string>]

title = <string>
* A title for the service template.

description = <string>
* The full description of the service being created.

entity_source_templates = <comma-separated list>
* The list of entity searches that create entities that can be used with this service.
* The list is used to populate the list of entity searches in the combined entity-service creation workflow.

entity_rules = <string>
* A list of entity rules (rules specification) used to associate entities to service
  created from this template.
* This field is the same as the entity_rules field in itsi_service.conf.spec.

recommended_kpis = <comma-separated list>
* A list of KPIs that are automatically added when a service is created with this template.

informational_kpis = <comma-separated list>
* A list of informational (no threshold) KPIs that are automatically added when a service is created with this template.

optional_kpis = <comma-separated list>
* A list of KPIs that are available for this service (but not added automatically).
itsi_da.conf.example

No example

itsi_deep_dive.conf

The following are the spec and example files for itsi_deep_dive.conf.

itsi_deep_dive.conf.spec

# This file contains possible attributes and values for uploading sample deep dives to the KV store.
#
# To upload deep dives to the KV store, place an itsi_deep_dive.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/local.
###
# You must restart Splunk software to enable configurations, unless you are editing them through the Splunk manager.
#
# To learn more about configuration files (including precedence) please see the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
#
# WARNING: Manual editing of this file is not recommended. Contact Support before proceeding.

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the top of the file.
# * Each conf file should have at most one default stanza. If there are multiple default stanzas, attributes are combined. In the case of multiple definitions of the same attribute, the last definition in the file wins.
# * If an attribute is defined at both the global level and in a
specific
# stanza, the value in the specific stanza takes precedence.

[<name>]

* A name or primary identifier for the deep dive.

focus_id = <string>
* The ID of the entity or service that is in focus in the deep dive.
* When an entity or service has focus, you see a list of metrics
  (performance metrics, event counts) for that entity/service.
* Any particular deep dive can have a particular IT context
  in focus at any given time.
* You can change the IT context in focus at any time. However,
  changing the focus has implications for historical tracking
  if not in a named deep dive.

title = <string>
* The title of the deep dive that is displayed in the UI.

lane_settings_collection = <array>
* An array of lane settings specifying each lane's configuration.

acl = <value>
* The team Access Control List (ACL) settings.

mod_time = <value>
* The last time the 'acl' setting was modified.

description = <value>
* Optional. The description of the deep dive.

is_named = <true|false>
* Whether or not this deep dive is named.
* A deep dive is considered "named" if you save it in the UI
  and give it a name. You might name a deep dive if you find it
  particularly useful and want to save it for future use.
* A deep dive is considered "unnamed" if you dynamically generated
  it (for example, from a drilldown) and did not save it.

_owner = <string>
* The user's KV store account in which to store the deep dive.
* In nearly all cases this value is "nobody".

source_itsi_da = <string>
* Optional. The ITSI module that acts as the source to define
  the deep dive.
* This attribute is used by the domain add-ons.
itsi_deep_dive.conf.example

No example

itsi_glass_table.conf

The following are the spec and example files for itsi_glass_table.conf.

itsi_glass_table.conf.spec

# This file contains possible attributes and values for uploading sample
# glass tables to the KV store.
#
# To upload glass tables to the KV store, place an
# itsi_glass_table.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/local.
#
# You must restart Splunk software to enable configurations, unless you are
# editing them through the Splunk manager.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
#
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
#
# WARNING: Manual editing of this file is not recommended. Contact Support before proceeding.

[<name>]

svg_content = <value>
* The SVG content settings.

latest = <value>
* The latest time in the time range.

earliest = <value>
* The earliest time in the time range.

svg_coordinates = <value>
* The SVG coordinate settings.

title = <string>
* The user-defined title of the glass table.

description = <string>
* The user defined description of the glass table.

mod_time = <value>
* Last modified time.

acl = <value>
* Access control information.

_owner = <string>
* The user account this deep dive belongs to.

source_itsi_da = <string>
* The ITSI module which is the source defining this glass table.

**itsi_glass_table.conf.example**

No example

**itsi_kpi_base_search.conf**

The following are the spec and example files for `itsi_kpi_base_search.conf`.

**itsi_kpi_base_search.conf.spec**

```
# This file contains possible settings you can use to upload sample
# KPI base searches to the KV store.
#
# There is an itsi_kpi_base_search.conf in
# SPLUNK_HOME/etc/apps/SA-ITOA/default. To set custom
# configurations, place an itsi_kpi_base_search.conf in
# SPLUNK_HOME/etc/apps/SA-ITOA/local.
# You must restart ITSI to enable new configurations.
#
# To learn more about configuration files (including precedence), see
# the
# documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
```
description = <string>
* A description of the KPI base search.

title = <string>
* The title of the KPI base search

_owner = <string>
* The owner of this KPI base search
* Default: itsi

base_search = <search>
* The search to execute in the KPI base search. This search is the source for the fields defined in metrics.

metrics = <json>
* A JSON blob that specifies the array of metrics to be collected.
* Example item in the blob:
* `{  
  "unit": "%",  
  "title": "CPU Utilization: %",  
  "entity_statop": "avg",  
  "aggregate_statop": "avg",  
  "_key": "620b26a6f286a508fd356d94",  
  "threshold_field": "cpu_load_percent"  
}`
* The threshold_field in the item corresponds to a field from the base search.

is_entity_breakdown = <boolean>
* Whether the metrics should be broken down by entities for threshold calculations.
* If "1", metrics are broken down by entities.
* If "0", metrics are not broken down by entities.

is_service_entity_filter = <boolean>
* Whether metrics should filter out entities not in the service.
* If "1", entities that don't belong to the service are filtered out.
* IF "0", entities that don't belong to the service are still included.

entity_id_fields = <string>
* The field in the base search used to look up the corresponding entity to filter KPIs.
* For example, host, ip, and so on.
* This field is required if the 'is_service_entity_filter' setting is set to "true".

entity_breakdown_id_fields = <string>
* The field in the base search used to look up the corresponding entity to split KPIs.
* For example, host, ip, and so on.
* This field is required if the 'is_entity_breakdown' setting is set to "true".

entity_alias_filtering_fields = <comma-separated list>
* A list of alias attributes to be used to filter out entities not in the service.
* Optional.
* This field is required if the 'is_service_entity_filter' setting is set to "true".

alert_period = <integer>
* The frequency, in minutes, at which to run the search.

search_alert_earliest = <integer>
* The time window, in minutes, over which to evaluate the metrics.

alert_lag = <integer>
* The amount of time, in seconds, to push back the metric evaluation.
* This setting corresponds to the data indexing lag.
* Default: 30

metric_qualifier = <string>
* The field in the base search used to further split metrics.
* CAUTION: You cannot modify this setting in the UI.

source_itsi_da = <string>
* The ITSI module that is the source defining this KPI base search.

itsi_kpi_base_search.conf.example

No example

itsi_kpi_template.conf

The following are the spec and example files for itsi_kpi_template.conf.

itsi_kpi_template.conf.spec

# This file contains possible settings you can use to upload sample KPI templates to the KV store.
#
# There is an itsi_kpi_template.conf in
$SPLUNK_HOME/etc/apps/SA-ITOA/default. To set custom
# configurations, place an itsi_kpi_template.conf in
$SPLUNK_HOME/etc/apps/SA-ITOA/local.
# You must restart ITSI to enable new configurations.
#
# To learn more about configuration files (including precedence), see
the
# documentation located at
#
http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

[<name>]

description = <string>
* The description of the KPI template bundle.

title = <string>
* The title of the bundle.

_owner = <string>
* The owner of the bundle.

kpis = <json>
* A JSON blob that specifies the array of KPI definitions.

source_itsi_da = <string>
* The ITSI module that is the source defining this KPI template.

itsu_kpi_template.conf.example

No example

itsu_kpi_threshold_template.conf

The following are the spec and example files for
itsu_kpi_threshold_template.conf.

itsu_kpi_threshold_template.conf.spec

# This file contains possible settings you can use to upload sample
# KPI threshold templates to the KV store.
#
# There is an itsi_kpi_threshold_template.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/default. To set custom configurations, place an itsi_kpi_threshold_template.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/local.
# You must restart ITSI to enable new configurations.
#
# To learn more about configuration files (including precedence), see the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

**[kpi_threshold_template]**

* Each stanza represents a KPI threshold template.
* Use threshold templates to build your time policies.
* You can create different policies with different time block combinations, such as work hours, off hours, or weekends.

```plaintext
title = <string>
* The title of the KPI threshold template.

description = <string>
* A description of the KPI threshold template.

time_variate_thresholds_specification = <JSON>
* A JSON blob containing the detailed time variant threshold object.

acl = <JSON>
* A JSON blob containing the ACL information for the KPI threshold template.
* Use the following format:

```javascript
{
   "perms":{
      "read": [<LIST_OF_ROLES>],
      "write": [<LIST_OF_ROLES>]
   },
   "can_share_user": [true|false],
   "can_share_app": [true|false],
   "modifiable": [true|false],
   "sharing": ["app"|"global"],
   "can_change_perms": [true|false],
   "can_share_global": [true|false],
   "owner": <OWNER_NAME_STRING>,
   "can_write": [true|false]
}
```
time_variate_thresholds = [True|False]
* Whether to enable time-variate thresholds.
* Time-variate thresholds accommodate normal variations in usage across your services and improve the accuracy of KPI and service health scores.
* For example, a time-variate threshold might take into account higher levels of usage during work hours, and lower levels of usage during off-hours and weekends.
* Default: True

adaptive_thresholding_training_window = <-7|14|30|60>[d]
* The time window over which historical KPI data is analyzed for adaptive threshold updates.
* You must have 7 days of summary data in the summary index for adaptive thresholding to work properly.
* Default: -7 days

adaptive_thresholds_is_enabled = [True|False]
* Whether to enable adaptive thresholding for policies in time-variate thresholds.
* Adaptive thresholding lets you create time polices that generate thresholds dynamically and update daily based on changes in your data.
* If you set this value to "true", the 'time_variate_thresholds' setting must also be set to "true".
* Default: False

itsi_kpi_threshold_template.conf.example

No example

itsi_module_settings.conf

The following are the spec and example files for itsi_module_settings.conf.

itsi_module_settings.conf.spec

# This file contains a setting for determining whether a module is editable
# in the module lister page.
#
# There is an itsi_module_settings.conf in each individual module directory (for example, $SPLUNK_HOME/etc/apps/DA-ITSI-OS/default for the Operating System module). To change this setting for a specific module, place an itsi_module_settings.conf in $SPLUNK_HOME/etc/apps/<module>/local. You must restart Splunk software to enable configurations.

# To learn more about configuration files (including precedence) please see the documentation located at

http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

[settings://<app>]

* "app" is the ID for the app that contains this configuration file.

is_read_only = <boolean>
* Whether the module shows as editable in the module lister page.
* If "1", the module is not editable in the module lister page.
* If "0", the module is editable in the module lister page.
* Default: 1

itsi_module_settings.conf.example

No example

itsi_module_viz.conf

The following are the spec and example files for itsi_module_viz.conf.

itsi_module_viz.conf.spec

# This file contains possible attributes and values for changing tab names and panel titles in a module details dashboard.

# There is an itsi_module_viz.conf in each module-specific directory within ITSI (for example, $SPLUNK_HOME/etc/apps/DA-ITSI-OS/default for the Operating System module). To edit these configurations, place an itsi_module_viz.conf in
$SPLUNK_HOME/etc/apps/DA-ITSI-OS/local.
# You must restart Splunk software to enable configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
#
# WARNING: Manual editing of this file is not recommended. Contact Support before proceeding.

[<view_name>]

* The name of the deep dive drilldown view within the ITSI module.

tabs = <comma-separated list>
* A list of tab IDs that will be included in this drilldown view.

<tabId>.control_token = <string>
* Used to run all the panel searches in a given tab.
* When the tab is shown, a list of search tokens are retrieved, the search tokens for
  all inactive tabs are removed from the list, and the search token for the active tab
  is added to the list. This guarantees that only the shown tab's panels are displayed.

<tabId>.title = <string>
* The title of the tab that is displayed in the UI.

<tabId>.row.<int> = <comma-separated list>
* A list of panels that are displayed on each row on a tab.
* The panels are formatted as follows: `<module_name>:<panel_name>`.
* These settings start at 'row.0' and go up to any number of rows that is needed for a tab.
* Example:
  
  row.0 = DA-ITSI-OS:panel1,DA-ITSI-LB:panel2

<tabId>.extendable_tab = <boolean>
* Whether the tab is considered an extendable tab.
* This setting is for user-created tabs so that a delete button appears on the tab
  in the UI.
* Any tabs that ship with the module default to "false".

<tabId>.activation_rule = <comma-separated list>
* A list of KPI elements that are associated with a given tab so that context-aware drilldown is enabled based on the selected KPI from the deep dive.
* Each element here is defined as the content from the "target_field"
  parameter from each
  selected KPI from the file itsi_kpi_template.conf.

entity_search_filter = <JSON>
* A JSON blob of entity rules to use to filter entities for entity
dropdown.

requested_entity_tokens = <comma-separated list>
* A list of entity attributes that are submitted as tokens.

**itsi_module_viz.conf.example**

No example

**itsi_notable_event_retention.conf**

The following are the spec and example files for
itsi_notable_event_retention.conf.

**itsi_notable_event_retention.conf.spec**

```bash
# This file contains attributes and values for defining how long notable
# event metadata remains
# in the KV store before it moves to the 'itsi_notable_archive' index.
#
# There is an itsi_notable_event_retention.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/default/.
# To set custom configurations, place an
itsi_notable_event_retention.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/local/. You must restart Splunk to
enable
# configurations.
#
# To learn more about configuration files (including precedence) please
see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
```
GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the top
#   of the file.
# * Each .conf file should have at most one default stanza. If there are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in
#   the file wins.
# * If an attribute is defined at both the global level and in a specific
#   stanza, the value in the specific stanza takes precedence.

[<collection_name>]

retentionTimeInSec = <seconds>
* The amount of time, in seconds, to retain the notable event object type.
* Default: 15768000 (6 months)

disabled = 0|1
* Whether this stanza is enabled or disabled.
* If "1", the stanza is disabled.
* If "0", the stanza is enabled.

object_type = <string>
* The notable event object type to retain.
* For example, comments, tags, external tickets, and so on.
* Required.
* If 'object_type' is not specified, the entire stanza is ignored.

itsi_notable_event_retention.conf.example

No example

itsi_notable_event_severity.conf

The following are the spec and example files for
itsi_notable_event_severity.conf.
itsi_notable_event_severity.conf.spec

# This file contains attributes and values for defining the colors
# associated with
# different severity levels in Episode Review.
#
# There is an itsi_notable_event_severity.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/default/.
# To set custom configurations, place an
# itsi_notable_event_severity.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/local/. You must restart Splunk to
# enable
# configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the
top
#   of the file.
# * Each .conf file should have at most one default stanza. If there are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in
#   the
#   file wins.
# * If an attribute is defined at both the global level and in a
#   specific
#   stanza, the value in the specific stanza takes precedence.

<name>

color = <string>
* A valid color code to represent the color of an episode with this
  severity.
  * Required.

lightcolor = <string>
* A valid color code to represent the severity in Splunk Light.
* Required.

label = <string>
* The severity label.
* For example, Info, Medium, Critical.

default = 0|1
* Set this flag to indicate the default severity of an event or episode.

itsi_notable_event_severity.conf.example

No example

itsi_notable_event_status.conf

The following are the spec and example files for itsi_notable_event_status.conf.

itsi_notable_event_status.conf.spec

# This file contains attributes and values for configuring label descriptions
# and episode status in Episode Review.
#
# There is an itsi_notable_event_status.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/default/.
# To set custom configurations, place an itsi_notable_event_status.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/local/. You must restart Splunk to enable
# configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
#  * You can also define global settings outside of any stanza, at the
top
#    of the file.
#  * Each .conf file should have at most one default stanza. If there
are
#    multiple default stanzas, attributes are combined. In the case of
#    multiple definitions of the same attribute, the last definition in
the
#    file wins.
#  * If an attribute is defined at both the global level and in a
specific
#    stanza, the value in the specific stanza takes precedence.

[<id>]

label = <string>
* A valid label for the episode status.
* Required.

default = <boolean>
* Indicates the initial status of an episode when it is generated in
Episode Review.
* Set this value to "1" if this label is the default label.

description = <string>
* A description of the episode label.

description = <string>
* Indicates the last status in the episode review workflow.
* Set this value to "1" if this label is the end of the
episode management workflow.

itsi_notable_event_status.conf.example

No example

itsi_service_analyzer.conf

The following are the spec and example files for itsi_service_analyzer.conf.

itsi_service_analyzer.conf.spec

# This file contains attributes and values for configuring the
GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
#  * You can also define global settings outside of any stanza, at the top
#    of the file.
#  * Each .conf file should have at most one default stanza. If there are
#    multiple default stanzas, attributes are combined. In the case of
#    multiple definitions of the same attribute, the last definition in
#    the file wins.
#  * If an attribute is defined at both the global level and in a specific
#    stanza, the value in the specific stanza takes precedence.

[<stanza_name>]

* A setting that you want to enable for the Service Analyzer.
* Currently, 'auto_refresh' is the only supported setting.
* NOTE: Auto-refresh is automatically disabled in real-time search mode
  for the Service Analyzer.

disabled = 0|1
* Whether this setting is disabled for the Service Analyzer.
* Required.
* If "1", the setting is disabled.
* If "0", the setting is enabled.
* Default: 1

interval = <seconds>
* The interval, in seconds, at which auto-refresh occurs for Service Analyzer.
  * Required.
  * Default: 120 (2 minutes)

_itsi_service_analyzer.conf.example_

No example

_itsi_service_template.conf_

The following are the spec and example files for _itsi_service_template.conf_.

_itsi_service_template.conf.spec_

```bash
# Copyright (C) 2005-2018 Splunk Inc. All Rights Reserved.
#
# This file contains attributes and values for configuring an application
# to export service templates for use within ITSI.
#
# To set custom configurations, place an itsi_service_template.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/local/. You must restart Splunk software to
# enable configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
#
# CAUTION: You can drastically affect your Splunk installation by changing these settings.
# Consult technical support (http://www.splunk.com/page/submit_issue)
# if you are not sure how
# to configure this file.

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
```
# * You can also define global settings outside of any stanza, at the top
#    of the file.
# * Each .conf file should have at most one default stanza. If there are
#    multiple default stanzas, attributes are combined. In the case of
#    multiple definitions of the same attribute, the last definition in
#    the file wins.
# * If an attribute is defined at both the global level and in a specific
#    stanza, the value in the specific stanza takes precedence.

*[<stanza_name>]*

**title = <string>**  
* A title for the service template.

**description = <string>**  
* The full description of the service being created.

**entity_rules = <string>**  
* A list of entity rules (rules specification) used to associate entities
to services created from this template.
* This setting is the same as the 'entity_rules' setting in itsi_service.conf.spec.
* Example:

```
[

    "rule_condition": "AND", 
    "rule_items": [
        {"field": "app_title", 
         "field_type": "alias", 
         "rule_type": "not", 
         "value": ""
        }, 
        {"field": "itsi_role", 
         "field_type": "info", 
         "rule_type": "matches", 
         "value": "apm"
        }, 
        {"field": "type", 
         "field_type": "info", 
         "rule_type": "matches", 
         "value": "application"
        }
    ]
]
```
recommended_kpis = <comma-separated list>
* A list of KPIs that are automatically added when a service
  is created with this template.

optional_kpis = <comma-separated list>
* A list of KPIs that are available for this service but
  not added automatically.

**itsi_service_template.conf.example**

No example

**itsi_settings.conf**

The following are the spec and example files for **itsi_settings.conf**.

**itsi_settings.conf.spec**

```
# Copyright (C) 2005-2018 Splunk Inc. All Rights Reserved.
# This file contains attributes and values for configuring the IT
Service
# Intelligence (ITSI) app.
#
# There is an itsi_settings.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/default/.
# To set custom configurations, place an itsi_settings.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/local/. You must restart Splunk
software to enable
# configurations.
#
# To learn more about configuration files (including precedence) please
see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
# CAUTION: You can drastically affect your Splunk installation by
changing these settings.
# Consult technical support (http://www.splunk.com/page/submit_issue)
if you are not sure how
```
GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the top
#   of the file.
# * Each .conf file should have at most one default stanza. If there are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in the
#   file wins.
# * If an attribute is defined at both the global level and in a specific
#   stanza, the value in the specific stanza takes precedence.

[datamodels://<app>]

* 'app' is the ID for the app containing the datamodel.

blacklist = <datamodel_names_list>
* A pipe-separated list of data model external authentication interface
  (EAI) names (IDs) to blacklist.
* NOTE: Data model names do not contain pipe characters.
* The blacklisted data models will not be supported and remain hidden
  from the ITSI UI.

[cloud]

show_migration_message  = <boolean>
* Removes Cloud migration messages about deprecated files or apps from
  the logs because this process is done internally.

[backup_restore]

* Defines settings related to ITSI backup/restore.

job_queue_timeout = <seconds>
* The amount of time, in seconds, before the backup/restore job queue
times out if the node owning the job has been down for too long to
allow other jobs to proceed.
* The minimum supported timeout period is 3600 seconds (1 hour). The system sets the timeout to 3600 seconds when a value lower than this is set.
* Default: 43200 (12 hours)

[import]

* Defines limits for import behavior.

import_batch_size = <integer>
* The number of rows or objects that the importer should analyze before attempting a save to the KV store.
* Default: 750

preview_sample_limit = <integer>
* The maximum number of rows that are returned from a preview request for a pending import.
* Default: 100

asynchronous_processing_threshold = <integer>
* The number of rows after which the bulk importer reads and stores the inbound content so that it can be processed at a more convenient time, rather than processing it immediately.

[metric_backfill]

* Defines backfill settings.

pre_calculation_window = <seconds>
* The size, in seconds, of the pre-calculation window for metric backfill.
* The smallest accepted value is 1. Increasing this value makes the backfill search faster, but less accurate.
* Default: 1

[sai_integration]

* Defines Splunk App for Infrastructure (SAI) settings.

show_detection_modal = <boolean>
* Whether or not to show the Splunk App for Infrastructure integration modal when the Service Analyzer loads.
* If "1", ITSI displays the integration modal.
* If "0", ITSI does not display the integration modal.
* Default: 1
[synced_kpi_scheduling]

disabled = <boolean>
* Indicates whether KPI saved searches have a randomized schedule or the same schedule.
* If ?1?, KPI saved searches run at staggered times throughout the scheduled interval.
* If ?0?, KPI saved searches all run at the same time during each scheduled interval.
* CAUTION: Changing this value to ?0? can have a significant performance impact. KPI saved searches are designed to run at different times to prevent the search scheduler from becoming overloaded.  
* Default = 1

itsi_settings.conf.example

No example

itsi_team.conf

The following are the spec and example files for itsi_team.conf.

itsi_team.conf.spec

# This file contains attributes and values for uploading ITSI teams to the KV store. By default, only the Global team ships with ITSI.
# There is an itsi_team.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/default. To set custom configurations, place an itsi_team.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/local/. You must restart Splunk software to enable configurations.  
# To learn more about configuration files (including precedence) please see the documentation located at http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles  
# CAUTION: You can drastically affect your Splunk installation by changing these settings.
# Consult technical support (http://www.splunk.com/page/submit_issue) if you are not sure how
# to configure this file.

**GLOBAL SETTINGS**

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the top
#   of the file.
# * Each .conf file should have at most one default stanza. If there are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in
#   the
#   file wins.
# * If an attribute is defined at both the global level and in a specific
#   stanza, the value in the specific stanza takes precedence.

[default_itsi_security_group]

title = <value>
* The name of the team.
* Duplicate team names are allowed, but be aware of other team names
  and use naming conventions to avoid confusion.

description = <value>
* A meaningful description of the team.

_immutable = <boolean>
* Whether users can edit the name of the team from the UI.
* If "1", the team name cannot be edited.
* If "0", the team name can be edited.
* Default: 0

acl = <dictionary>
* An Access Control List (ACL) associating ITOA roles with permissions
  within that team.
* Assign read or write access to the listed ITOA roles as appropriate.
  If a role has write permissions for a team, a user with this role can
  create and modify services in the team. The user can't delete a
  service
  in the team unless the role has the delete capability for a service.
**[notable_event_review_security_group]**

`disabled = <boolean>`
* Use this setting to turn off Role-Based Access Control (RBAC) for Episode Review only.
* If you set this flag to "1", all users will be able to see all events within Episode Review, regardless of their team.
* If "1", RBAC is disabled for Episode Review.
* If "0", RBAC is enabled for Episode Review.
* Default: 0

**itsi_team.conf.example**

No example

**limits.conf**

The following are the spec and example files for `limits.conf`.

**limits.conf.spec**

```
#    Version 7.2.0
#

OVERVIEW

# This file contains descriptions of the settings that you can use to
# configure limitations for the search commands.
# # Each stanza controls different search commands settings.
# # There is a limits.conf file in the $SPLUNK_HOME/etc/system/default/
directory.
# Never change or copy the configuration files in the default directory.
# The files in the default directory must remain intact and in their
# original
# location.
#
# To set custom configurations, create a new file with the name
```
limits.conf in
# the $SPLUNK_HOME/etc/system/local/ directory. Then add the specific
# settings
# that you want to customize to the local configuration file.
# For examples, see limits.conf.example. You must restart the Splunk
instance
# to enable configuration changes.
#
# To learn more about configuration files (including file precedence)
see the
# documentation located at
#
http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
#
# About Distributed Search
# Unlike most settings which affect searches, limits.conf settings are
# not
# provided by the search head to be used by the search peers. This
means
# that if you need to alter search-affecting limits in a distributed
# environment, typically you will need to modify these settings on the
# relevant peers and search head for consistent results.
#
GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
#   * You can also define global settings outside of any stanza, at the
top of
#     the file.
#   * Each .conf file should have at most one default stanza. If there
are
#     multiple default stanzas, settings are combined. In the case of
multiple definitions of the same setting, the last definition in
the
#     file takes precedence.
#   * If a setting is defined at both the global level and in a specific
stanza, the value in the specific stanza takes precedence.
#
# CAUTION: Do not alter the settings in the limits.conf file unless you
know
# what you are doing. Improperly configured limits might result in
#
splunkd crashes, memory overuse, or both.
[default]

DelayArchiveProcessorShutdown = <bool>
* Specifies whether during splunk shutdown archive processor should finish processing archive file under process.
  * When set to ?false?: The archive processor abandons further processing of the archive file and will process again from start again.
  * When set to ?true?: The archive processor will complete processing of the archive file. Shutdown will be delayed.
  * Default: false

max_mem_usage_mb = <non-negative integer>
* Provides a limitation to the amount of RAM, in megabytes (MB), a batch of events or results will use in the memory of a search process.
* Operates on an estimation of memory use which is not exact. The estimation can deviate by an order of magnitude or so to both the smaller and larger sides.
* The limitation is applied in an unusual way; if the number of results or events exceeds maxresults, AND the estimated memory exceeds this limit, the data is spilled to disk.
* This means, as a general rule, lower limits will cause a search to use more disk I/O and less RAM, and be somewhat slower, but should cause the same results to typically come out of the search in the end.
* This limit is applied currently to a number, but not all search processors.
  However, more will likely be added as it proves necessary.
* The number is thus effectively a ceiling on batch size for many components of search for all searches run on this system.
* When set to ?0?: Specifies that the size is unbounded. Searches might be allowed to grow to arbitrary sizes.
* NOTE:
  * The mvexpand command uses the ?max_mem_usage_mb? value in a different way.
    * The mvexpand command has no combined logic with ?maxresults?.
    * If the memory limit is exceeded, output is truncated, not spilled to disk.
  * The stats command processor uses the ?max_mem_usage_mb? value in the following way.
    * If the estimated memory usage exceeds the specified limit, the results are
spilled to disk.
* If 0 is specified, the results are spilled to the disk when the number of results exceed the `?maxresultrows?` setting.
* The eventstats command processor uses the `?max_mem_usage_mb?` value in the following way.
* Both the `?max_mem_usage_mb?` and the `?maxresultrows?` settings are used to determine the maximum number of results to return. If the limit for one setting is reached, the eventstats processor continues to return results until the limit for the other setting is reached. When both limits are reached, the eventstats command stops adding the requested fields to the search results.
* If you set `?max_mem_usage_mb?` to 0, the eventstats command processor uses only the `?maxresultsrows?` setting as the threshold. When the number of results exceeds the `?maxresultsrows?` setting, the eventstats command processor stops adding the requested fields to the search results.
* Default: 200

`min_batch_size_bytes = <integer>`
* Specifies the size, in megabytes (MB), of the file/tar after which the file is handled by the batch reader instead of the trailing processor.
* Global parameter, cannot be configured per input.
* NOTE: Configuring this to a very small value could lead to backing up of jobs at the tailing processor.
* Default: 20

`regex_cpu_profiling = <bool>`
* Enable CPU time metrics for RegexProcessor. Output will be in the metrics.log file.
  Entries in metrics.log will appear per_host_regex_cpu,
  per_source_regex_cpu,
  per_sourcetype_regex_cpu, per_index_regex_cpu.
* Default: false

**[searchresults]**

* This stanza controls search results for a variety of Splunk search commands.

`compression_level = <integer>`
* Compression level to use when writing search results to .csv.gz files.
* Default: 1
maxresultrows = <integer>
* Configures the maximum number of events are generated by search commands
  which grow the size of your result set (such as multikv) or that create
  events. Other search commands are explicitly controlled in specific stanzas
  below.
* This limit should not exceed 50000.
* Default: 50000

tocsv_maxretry = <integer>
* Maximum number of times to retry the atomic write operation.
* When set to 1: Specifies that there will be no retries.
* Default: 5

tocsv_retryperiod_ms = <integer>
* Period of time to wait before each retry.
* Default: 500

* These setting control logging of error messages to the info.csv file. All messages will be logged to the search.log file regardless of these settings.

[search_info]

* This stanza controls logging of messages to the info.csv file.
* Messages logged to the info.csv file are available to REST API clients and Splunk Web. Limiting the messages added to info.csv will mean that these messages will not be available in the UI and/or the REST API.

filteredindexes_log_level = [DEBUG/INFO/WARN/ERROR]
* Log level of messages when search returns no results because user has no permissions to search on queried indexes.

infocsv_log_level = [DEBUG/INFO/WARN/ERROR]
* Limits the messages which are added to the info.csv file to the stated level and above.
* For example, if ?infocsv_log_level? is WARN, messages of type WARN and higher will be added to the info.csv file.

max_infocsv_messages  = <positive integer>
* If more than max_infocsv_messages log entries are generated, additional entries will not be logged in the info.csv file. All entries will still be logged in the search.log file.
show_warn_on_filtered_indexes = <boolean>
* Log warnings if search returns no results because user has no permissions to search on queried indexes.

[subsearch]

* This stanza controls subsearch results.
* NOTE: This stanza DOES NOT control subsearch results when a subsearch is called by commands such as join, append, or appendcols.
* Read more about subsearches in the online documentation:
  http://docs.splunk.com/Documentation/Splunk/latest/Search/Aboutsubsearches

maxout = <integer>
* Maximum number of results to return from a subsearch.
* This value cannot be greater than or equal to 10500.
* Default: 10000

maxtime = <integer>
* Maximum number of seconds to run a subsearch before finalizing
* Default: 60

ttl = <integer>
* The time to live (ttl), in seconds, of the cache for the results of a given subsearch.
* Do not set this below 120 seconds.
* See the definition in the [search] stanza under the ?TTL? section for more details on how the ttl is computed.
* Default: 300 (5 minutes)

SEARCH COMMAND

# This section contains the limitation settings for the search command.
# The settings are organized by type of setting.

[search]

# The settings under the [search] stanza are organized by type of setting.

# Batch search
# This section contains settings for batch search.

allow_batch_mode = <bool>
* Specifies whether or not to allow the use of batch mode which searches
  in disk based batches in a time insensitive manner.
* In distributed search environments, this setting is used on the search
  head.
* Default: true

batch_search_max_index_values = <int>
* When using batch mode, this limits the number of event entries read
  from the
  index file. These entries are small, approximately 72 bytes. However
  batch
  mode is more efficient when it can read more entries at one time.
* Setting this value to a smaller number can lead to slower search
  performance.
* A balance needs to be struck between more efficient searching in batch
  mode
* and running out of memory on the system with concurrently running
  searches.
* Default: 10000000

batch_search_max_pipeline = <int>
* Controls the number of search pipelines that are
  launched at the indexer during batch search.
* Increasing the number of search pipelines should help improve search
  performance, however there will be an increase in thread and memory
  usage.
* This setting applies only to searches that run on remote indexers.
* Default: 1

batch_search_max_results_aggregator_queue_size = <int>
* Controls the size, in MB, of the search results queue to which all
  the search pipelines dump the processed search results.
* Increasing the size can lead to search performance gains.
  Decreasing the size can reduce search performance.
* Do not specify zero for this setting.
* Default: 100

batch_search_max_serialized_results_queue_size = <int>
* Controls the size, in MB, of the serialized results queue from which
  the serialized search results are transmitted.
* Increasing the size can lead to search performance gains.
  Decreasing the size can reduce search performance.
* Do not specify zero for this setting.
* Default: 100

NOTE: The following batch search settings control the periodicity of
retries
to search peers in the event of failure (Connection errors, and
The interval exists between failure and first retry, as well as successive retries in the event of further failures.

`batch_retry_min_interval = <int>`
* When batch mode attempts to retry the search on a peer that failed, specifies the minimum time, in seconds, to wait to retry the search.
* Default: 5

`batch_retry_max_interval = <int>`
* When batch mode attempts to retry the search on a peer that failed, specifies the maximum time, in seconds, to wait to retry the search.
* Default: 300 (5 minutes)

`batch_retry_scaling = <double>`
* After a batch retry attempt fails, uses this scaling factor to increase the time to wait before trying the search again.
* The value should be > 1.0.
* Default: 1.5

# Bundles
This section contains settings for bundles and bundle replication.

`load_remote_bundles = <bool>`
* On a search peer, allow remote (search head) bundles to be loaded in splunkd.
* Default: false.

`replication_file_ttl = <int>`
* The time to live (ttl), in seconds, of bundle replication tarballs, for example: *.bundle files.
* Default: 600 (10 minutes)

`replication_period_sec = <int>`
* The minimum amount of time, in seconds, between two successive bundle replications.
* Default: 60

`sync_bundle_replication = [0|1|auto]`
* A flag that indicates whether configuration file replication blocks searches or is run asynchronously.
* When set to ?auto?: The Splunk software uses asynchronous replication only if all of the peers support asynchronous bundle replication. Otherwise synchronous replication is used.
* Default: auto

# Concurrency
# This section contains settings for search concurrency limits.

base_max_searches = <int>
* A constant to add to the maximum number of searches, computed as a multiplier of the CPUs.
* Default: 6

max_rt_search_multiplier = <decimal number>
* A number by which the maximum number of historical searches is multiplied to determine the maximum number of concurrent real-time searches.
* Note: The maximum number of real-time searches is computed as:
  max_rt_searches = max_rt_search_multiplier * max_hist_searches
* Default: 1

max_searches_per_cpu = <int>
* The maximum number of concurrent historical searches for each CPU.
  The system-wide limit of historical searches is computed as:
  max_hist_searches = max_searches_per_cpu * number_of_cpus + base_max_searches
* NOTE: The maximum number of real-time searches is computed as:
  max_rt_searches = max_rt_search_multiplier * max_hist_searches
* Default: 1

# Distributed search

# This section contains settings for distributed search connection information.

addpeer_skew_limit = <positive integer>
* Absolute value of the largest time skew, in seconds, that is allowed when configuring a search peer from a search head, independent of time.
* If the difference in time (skew) between the search head and the peer is greater than addpeer_skew_limit, the search peer is not added.
* This is only relevant to manually added peers. This setting has no effect on index cluster search peers.
* Default: 600 (10 minutes)

fetch_remote_search_log = [enabled|disabledSavedSearches|disabled]
* When set to enabled: All remote search logs are downloaded barring the oneshot search.
* When set to disabledSavedSearches: Downloads all remote logs other than saved search logs and oneshot search logs.
* When set to disabled: Irrespective of the search type, all remote search log download functionality is disabled.
* NOTE:
  * The previous values:[true|false] are still supported but not
recommended.
  * The previous value of ?true? maps to the current value of ?enabled?.
  * The previous value of ?false? maps to the current value of
  ?disabled?.
  * Default: disabledSavedSearches

max_chunk_queue_size = <int>
* The maximum size of the chunk queue.
  * default: 10000000

max_combiner_memevents = <int>
* Maximum size of the in-memory buffer for the search results combiner.
  The <int> is the number of events.
  * Default: 50000

max_tolerable_skew = <positive integer>
* Absolute value of the largest timeskew, in seconds, that we will
tolerate
  between the native clock on the search head and the native clock on
the peer
  (independent of time-zone).
* If this time skew is exceeded, a warning is logged. This estimate is
  approximate and tries to account for network delays.
  * Default: 60

max_workers_searchparser = <int>
* The number of worker threads in processing search result when using
  round
  robin policy.
  * default: 5

results_queue_min_size = <integer>
* The minimum size, of search result chunks, that will be kept from
  peers
  for processing on the search head before throttling the rate that data
  is accepted.
* The minimum queue size in chunks is the ?results_queue_min_size?
  value
  and the number of peers providing results, which ever is greater.
  * Default: 10

result_queue_max_size = <integer>
* The maximum size, in MB, that will be kept from peers for processing
  on
  the search head before throttling the rate that data is accepted.
* The ?results_queue_min_size? value takes precedence. The number of
  search
  results chunks specified by ?results_queue_min_size? will always be
  retained in the queue even if the combined size in MB exceeds the
?result_queue_max_size? value.
  * Default: 100
results_queue_read_timeout_sec = <integer>
* The amount of time, in seconds, to wait when the search executing on
  the
  search head has not received new results from any of the peers.
* Cannot be less than the 'receiveTimeout' setting in the
distsearch.conf
  file.
* Default: 900

batch_wait_after_end = <int>
* DEPRECATED: Use the 'results_queue_read_timeout_sec' setting instead.

############################################################################
# Field stats
############################################################################
# This section contains settings for field statistics.

fieldstats_update_freq = <number>
* How often to update the field summary statistics, as a ratio to the
  elapsed
  run time so far.
* Smaller values means update more frequently.
* When set to ?0?: Specifies to update as frequently as possible.
* Default: 0

fieldstats_update_maxperiod = <number>
* The maximum period, in seconds, for updating field summary statistics.
* When set to ?0?: Specifies that there is not maximum period. The
  period
  is dictated by the calculation:
  current_run_time x fieldstats_update_freq
* Fractional seconds are allowed.
* Default: 60

min_freq = <number>
* Minimum frequency of a field that is required for the field to be
  included
  in the /summary endpoint.
* The frequency must be a fraction >=0 and <=1.
* Default: 0.01 (1%)

############################################################################
# History
############################################################################
# This section contains settings for search history.

enable_history = <bool>
* Specifies whether to keep a history of the searches that are run.
* Default: true

max_history_length = <int>
* Maximum number of searches to store in history for each user and
application.
* Default: 1000

# Memory tracker

# This section contains settings for the memory tracker.

enable_memory_tracker = <bool>
* Specifies if the memory tracker is enabled.
* When set to ?false? (disabled): The search is not terminated even if
  the search exceeds the memory limit.
* When set to ?true?: Enables the memory tracker.
* Must be set to ?true? to enable the
  ?search_process_memory_usage_threshold?
  setting or the ?search_process_memory_usage_percentage_threshold?
  setting.
* Default: false

search_process_memory_usage_threshold = <double>
* To use this setting, the ?enable_memory_tracker? setting must be set
to ?true?.
* Specifies the maximum memory, in MB, that the search process can
  consume
  in RAM.
* Search processes that violate the threshold are terminated.
* If the value is set to 0, then search processes are allowed to grow
  unbounded in terms of in memory usage.
* Default: 4000 (4GB)

search_process_memory_usage_percentage_threshold = <float>
* To use this setting, the ?enable_memory_tracker? setting must be set
to ?true?.
* Specifies the percent of the total memory that the search process is
  entitled to consume.
* Search processes that violate the threshold percentage are terminated.
* If the value is set to zero, then splunk search processes are allowed
to
  grow unbounded in terms of percentage memory usage.
* Any setting larger than 100 or less than 0 is discarded and the
  default
  value is used.
* Default: 25%

# Meta search

# This section contains settings for meta search.

allow_inexact_metasearch = <bool>
* Specifies if a metasearch that is inexact be allowed.
* When set to ?true?: An INFO message is added to the inexact
metasearches.
* When set to \texttt{false}: A fatal exception occurs at search parsing time.
* Default: \texttt{false}

\texttt{indexed\_as\_exact\_metasearch = \texttt{bool}}
* Specifies if a metasearch can process \texttt{\langle field\rangle=\langle value\rangle} the same as \texttt{\langle field\rangle::\langle value\rangle}, if \texttt{\langle field\rangle} is an indexed field.
* When set to \texttt{true}: Allows a larger set of metasearches when the \texttt{allow\_inexact\_metasearch} setting is \texttt{false}. However, some of the metasearches might be inconsistent with the results of doing a normal search.
* Default: \texttt{false}

# Misc

# This section contains miscellaneous search settings.

disk\_usage\_update\_period = \texttt{number}
* Specifies how frequently, in seconds, should the search process estimate the artifact disk usage.
* The quota for the amount of disk space that a search job can use is controlled by the 'srchDiskQuota' setting in the authorize.conf file.
* Exceeding this quota causes the search to be auto-finalized immediately, even if there are results that have not yet been returned.
* Fractional seconds are allowed.
* Default: \texttt{10}

dispatch\_dir\_warning\_size = \texttt{int}
* Specifies the number of jobs in the dispatch directory that triggers when to issue a bulletin message. The message warns that performance might be impacted.
* Default: \texttt{5000}

do\_not\_use\_summaries = \texttt{bool}
* Do not use this setting without working in tandem with Splunk support.
* This setting is a very narrow subset of \texttt{summary\_mode=none}.
* When set to \texttt{true}: Disables some functionality that is necessary for report acceleration.
* In particular, when set to \texttt{true}, search processes will no longer query the main splunkd's /admin/summarization endpoint for report acceleration summary IDs.
* In certain narrow use-cases this might improve performance if report acceleration (savedsearches.conf:auto_summarize) is not in use, by lowering the main splunkd's process overhead.
* Default: \texttt{false}
enable_datamodel_meval = <bool>
* Enable concatenation of successively occurring evals into a single
  comma-separated eval during the generation of datamodel searches.
* default: true

force_saved_search_dispatch_as_user = <bool>
* Specifies whether to overwrite the ?dispatchAs? value.
* When set to ?true?: The ?dispatchAs? value is overwritten by ?user? 
  regardless of the [user|owner] value in the savedsearches.conf file.
* When set to ?false?: The value in the savedsearches.conf file is used.
* You might want to set this to ?true? to effectively disable
  ?dispatchAs = owner? for the entire install, if that more closely 
  aligns
  with security goals.
* Default: false

max_id_length = <integer>
* Maximum length of the custom search job ID when spawned by using 
  REST API argument ?id?.

search_keepalive_frequency = <int>
* Specifies how often, in milliseconds, a keepalive is sent while a 
  search is running.
* Default: 30000 (30 seconds)

search_keepalive_max = <int>
* The maximum number of uninterupted keepalives before the connection is 
  closed.
* This counter is reset if the search returns results.
* Default: 100

search_retry = <bool>
* Specifies whether the Splunk software retries parts of a search within 
  a 
  currently-running search processs when there are indexer failures in the 
  indexer clustering environment.
* Indexers can fail during rolling restart or indexer upgrade when indexer 
  clustering is enabled. Indexer reboots can also result in failures.
* This setting applies only to historical search in batch mode, 
  real-time 
  search, and indexed real-time search.
* When set to true, the Splunk software attempts to rerun searches on 
  indexer 
  cluster nodes that go down and come back up again. The search process 
  on the 
  search head maintains state information about the indexers and 
  buckets.
* NOTE: Search retry is on a best-effort basis, and it is possible 
  for Splunk software to return partial results for searches
without warning when you enable this setting.  
* When set to false, the search process will stop returning results from  
a specific  
    indexer when that indexer undergoes a failure.  
* Default: false

stack_size = <int>  
* The stack size, in bytes, of the thread that executes the search.  
* Default: 4194304 (4MB)

summary_mode = [all|only|none]  
* Specifies if precomputed summary data are to be used.  
* When set to ?all?: Use summary data if possible, otherwise use raw  
data.  
* When set to ?only?: Use summary data if possible, otherwise do not use  
    any data.  
* When set to ?none?: Never use precomputed summary data.  
* Default: all

track_indextime_range = <bool>  
* Specifies if the system should track the _indextime range of returned  
search results.  
* Default: true

use_bloomfilter = <bool>  
* Controls whether to use bloom filters to rule out buckets.  
* Default: true

use_metadata_elimination = <bool>  
* Control whether to use metadata to rule out buckets.  
* Default: true

results_serial_format = [csv|srs]  
* The internal format used for storing serialized results on disk.  
* Options:  
    csv: Comma-separated values format  
    srs: Splunk binary format  
* Default: srs  
* NOTE: Do not change unless instructed to do so by Splunk Support.

results_compression_algorithm = [gzip|none]  
* The compression algorithm used for storing serialized results on disk.  
* Options:  
    gzip: gzip  
    none: No compression  
* Default: gzip  
* NOTE: Do not change unless instructed to do so by Splunk Support.

use_dispatchtmp_dir = <bool>  
* DEPRECATED. This setting has been deprecated and has no effect.

auto_cancel_after_pause = <integer>
* Specifies the amount of time, in seconds, that a search must be paused before
  the search is automatically cancelled.
* If set to 0, a paused search is never automatically cancelled.
* Default: 0

always_include_indexedfield_lispy = <bool>
* Controls if we should always search for a field that does not have
  INDEXED=true set in fields.conf using both the indexed and non-indexed forms
* If true, when searching for <field>=<val>, we search the lexicon for both
  <field>::<val> and <val>
* If false, when searching for <field>=<val>, we search the lexicon for only
  <val>
* Set to true if you have fields that are sometimes indexed and sometimes not indexed. For field name that are always indexed, it is much better performance wise to set INDEXED=true in fields.conf for that field instead.
* Default: false

############################################################################
# Parsing
############################################################################
# This section contains settings related to parsing searches.

max_macro_depth = <int>
* Maximum recursion depth for macros. Specifies the maximum levels for macro
  expansion.
* It is considered a search exception if macro expansion does not stop after
  this many levels.
* Value must be greater than or equal to 1.
* Default: 100

max_subsearch_depth = <int>
* Maximum recursion depth for subsearches. Specifies the maximum levels for
  subsearches.
* It is considered a search exception if a subsearch does not stop after
  this many levels.
* Default: 8

min_prefix_len = <integer>
* The minimum length of a prefix before a wildcard (*) to use in the query
  to the index.
* Default: 1

use_directives = <bool>

376
* Specifies whether a search can take directives and interpret them into arguments.
* This is used in conjunction with the search optimizer in order to improve search performance.
* Default: true

# Phased execution settings
# This section contains settings for multi-phased execution

phased_execution = <bool>
DEPRECATED This setting has been deprecated.

phased_execution_mode = [multithreaded|auto|singlethreaded]
* NOTE: Do not change this setting unless instructed to do so by Splunk Support!
* Controls whether searches use the multiple-phase method of search execution,
  which is required for parallel reduce functionality as of Splunk Enterprise 7.1.0.
* When set to 'multithreaded' the Splunk platform uses the multiple-phase search execution method. Allows usage of the 'redistribute' command.
* When set to 'auto', the Splunk platform uses the multiple-phase search execution method when the 'redistribute' command is used in the search string. If the 'redistribute' command is not present in the search string, the single-phase search execution method is used.
* When set to 'singlethreaded' the Splunk platform uses the single-threaded search execution method, which does not allow usage of the 'redistribute' command.
* Default: multithreaded

# Preview
# This section contains settings for previews.

max_preview_period = <integer>
* The maximum time, in seconds, between previews.
* Used with the preview interval that is calculated with the ?preview_duty_cycle? setting.
* When set to 0: Specifies unlimited time between previews.
* Default: 0

min_preview_period = <integer>
* The minimum time, in seconds, required between previews. When the calculated
interval using ?preview_duty_cycle? indicates previews should be run frequently. This setting is used to limit the frequency with which previews run.
* Default: 1

preview_duty_cycle = <number>
* The maximum time to spend generating previews, as a fraction of the total search time.
* Must be > 0.0 and < 1.0
* Default: 0.25

preview_freq = <timespan> or <ratio>
* Minimum amount of time between results preview updates.
* If specified as a number, between > 0 and < 1, the minimum time between previews is computed as a ratio of the amount of time that the search has been running, or as a ratio of the length of the time window for real-time windowed searches.
* Default: a ratio of 0.05

# Quota or queued searches

default_allow_queue = [0|1]
* Unless otherwise specified by using a REST API argument, specifies if an asynchronous job spawning request should be queued on quota violation. If not, an http error of server too busy is returned.
* Default: 1 (true)

dispatch_quota_retry = <integer>
* The maximum number of times to retry to dispatch a search when the quota has been reached.
* Default: 4

dispatch_quota_sleep_ms = <integer>
* The time, in milliseconds, between retrying to dispatch a search when a quota is reached.
* Retries the given number of times, with each successive wait 2x longer than the previous wait time.
* Default: 100

enable_cumulative_quota = <bool>
* Specifies whether to enforce cumulative role based quotas.
* Default: false
queued_job_check_freq = <number>
* Frequency, in seconds, to check queued jobs to determine if the jobs can
  be started.
* Fractional seconds are allowed.
* Default: 1.

# This section contains settings for reading chunk controls.

chunk_multiplier = <integer>
* A multiplier that the ?max_results_perchunk?, ?min_results_perchunk?,
  and
  ?target_time_perchunk? settings are multiplied by for a long running
  search.
* Default: 5

long_search_threshold = <integer>
* The time, in seconds, until a search is considered "long running".
* Default: 2

max_rawsize_perchunk = <integer>
* The maximum raw size, in bytes, of results for each call to search
  (in dispatch).
* When set to ?0?: Specifies that there is no size limit.
* This setting is not affected by the ?chunk_multiplier? setting.
* Default: 100000000 (100MB)

max_results_perchunk = <integer>
* The maximum number of results to emit for each call to the preview
  data
  generator.
* Default: 2500

max_results_perchunk = <integer>
* Maximum results for each call to search (in dispatch).
* Must be less than or equal to the ?maxresultrows? setting.
* Default: 2500

min_results_perchunk = <integer>
* The minimum results for each call to search (in dispatch).
* Must be less than or equal to the ?max_results_perchunk? setting.
* Default: 100

target_time_perchunk = <integer>
* The target duration, in milliseconds, of a particular call to fetch
  search results.
* Default: 2000 (2 seconds)
# Real-time

This section contains settings for real-time searches.

check_splunkd_period = <number>
* Amount of time, in seconds, that determines how frequently the search process (when running a real-time search) checks whether the parent process (splunkd) is running or not.
* Fractional seconds are allowed.
* Default: 60 (1 minute)

realtime_buffer = <int>
* Maximum number of accessible events to keep for real-time searches in Splunk Web.
* Acts as circular buffer after this buffer limit is reached.
* Must be greater than or equal to 1.
* Default: 10000

# Remote storage

This section contains settings for remote storage.

bucket_localize_acquire_lock_timeout_sec = <int>
* The maximum amount of time, in seconds, to wait when attempting to acquire a lock for a localized bucket.
* When set to 0, waits indefinitely.
* This setting is only relevant when using remote storage.
* Default: 60 (1 minute)

bucket_localize_max_timeout_sec = <int>
* The maximum amount of time, in seconds, to spend localizing a bucket stored in remote storage.
* If the bucket contents (what is required for the search) cannot be localized in that timeframe, the bucket will not be searched.
* When set to ?0?: Specifies an unlimited amount of time.
* This setting is only relevant when using remote storage.
* Default: 300 (5 minutes)

bucket_localize_status_check_period_ms = <int>
* The amount of time, in milliseconds, between consecutive status checks to see if the needed bucket contents required by the search have been localized.
* This setting is only relevant when using remote storage.
* The minimum and maximum values are 10 and 60000, respectively. If the specified value falls outside this range, it is effectively set to the
nearest value within the range. For example, if you set the value to 70000, the effective value will be 60000.
* Default: 500 (.5 seconds)

bucket_localize_max_lookahead = <int>
* Specifies the maximum number of buckets the search command localizes for look-ahead purposes, in addition to the required bucket.
* Increasing this value can improve performance, at the cost of additional network/io/disk utilization.
* Valid values are 0-64. Any value larger than 64 will be set to 64. Other invalid values will be discarded and the default will be substituted.
* This setting is only relevant when using remote storage.
* Default: 5

bucket_localize_lookahead_priority_ratio = <int>
* A value of N means that lookahead localizations will occur only 1 out of N search localizations, if any.
* Default: 5

bucket_predictor = [consec_not_needed|everything]
* Specifies which bucket file prediction algorithm to use.
* Do not change this unless you know what you are doing.
* Default: consec_not_needed

# Results storage
# This section contains settings for storing final search results.
max_count = <integer>
* The number of events that can be accessible in any given status bucket (when status_buckets = 0).
* The last accessible event in a call that takes a base and count.
* Note: This value does not reflect the number of events displayed in the UI after the search is evaluated or computed.
* Default: 500000

max_events_per_bucket = <integer>
* For searches with ?status_buckets>0?, this setting limits the number of events retrieved for each timeline bucket.
* Default: 1000 in code.

status_buckets = <integer>
* The approximate maximum number buckets to generate and maintain in the timeline.
* Default: 0, which means do not generate timeline information
truncate_report = [1|0]
* Specifies whether or not to apply the ?max_count? setting to report output.
* Default: 0 (false)

write_multifile_results_out = <bool>
* At the end of the search, if results are in multiple files, write out the multiple files to the results_dir directory, under the search results directory.
* This setting speeds up post-processing search, since the results will already be split into appropriate size files.
* Default: true

############################################################################
# Search process
############################################################################
# This section contains settings for search process configurations.

idle_process_cache_search_count = <int>
* The number of searches that the search process must reach, before purging older data from the cache. The purge is performed even if the ?idle_process_cache_timeout" has not been reached.
* When a search process is allowed to run more than one search, the search process can cache some data between searches.
* When set to a negative value: No purge occurs, no matter how many searches are run.
* Has no effect on Windows if ?search_process_mode? is not ?auto" or if ?max_searches_per_process? is set to 0 or 1.
* Default: 8

idle_process_cache_timeout = <number>
* The amount of time, in seconds, that a search process must be idle before the system purges some older data from these caches.
* When a search process is allowed to run more than one search, the search process can cache some data between searches.
* When set to a negative value: No purge occurs, no matter on how long the search process is idle.
* When set to ?0?: Purging always occurs, regardless of whether the process has been idle or not.
* Has no effect on Windows if ?search_process_mode? is not "auto" or if ?max_searches_per_process? is set to 0 or 1.
* Default: 0.5 (seconds)

idle_process_regex_cache_hiwater = <int>
* A threshold for the number of entries in the regex cache. If the regex
cache
  grows to larger than this number of entries, the systems attempts to
  purge some of the older entries.
* When a search process is allowed to run more than one search, the
  search
  process can cache compiled regex artifacts.
* Normally the "idle_process_cache_search_count? and the
  ?idle_process_cache_timeout? settings will keep the regex cache a
  reasonable size. This setting is to prevent the cache from growing
  extremely large during a single large search.
* When set to a negative value: No purge occurs, not matter how large
  the cache.
* Has no effect on Windows if ?search_process_mode? is not "auto" or
  if ?max_searches_per_process? is set to 0 or 1.
* Default: 2500

idle_process_reaper_period = <number>
* The amount of time, in seconds, between checks to determine if there
  are
  too many idle search processes.
* When a search process is allowed to run more than one search, the
  system
  checks if there are too many idle search processes.
* Has no effect on Windows if ?search_process_mode? is not "auto" or
  if ?max_searches_per_process? is set to 0 or 1.
* Default: 30

launcher_max_idle_checks = <int>
* Specifies the number of idle processes that are inspected before
  giving up
  and starting a new search process.
* When allowing more than one search to run for each process, the system
  attempts to find an appropriate idle process to use.
* When set to a negative value: Every eligible idle process is
  inspected.
* Has no effect on Windows if ?search_process_mode? is not "auto" or
  if ?max_searches_per_process? is set to 0 or 1.
* Default: 5

launcher_threads = <int>
* The number of server thread to run to manage the search processes.
* Valid only when more than one search is allowed to run for each
  process.
* Has no effect on Windows if ?search_process_mode? is not "auto" or
  if ?max_searches_per_process? is set to 0 or 1.
* Default: -1 (a value is selected automatically)

max_old_bundle_idle_time = <number>
* The amount of time, in seconds, that a process bundle must be idle
  before
  the process bundle is considered for reaping.
* Used when reaping idle search processes and the process is not

383
configured
   with the most recent configuration bundle.
* When set to a negative value: The idle processes are not reaped sooner
   than normal if the processes are using an older configuration bundle.
* Has no effect on Windows if ?search_process_mode? is not "auto" or
   if ?max_searches_per_process? is set to 0 or 1.
* Default: 5

max_searches_per_process = <int>
* On UNIX, specifies the maximum number of searches that each search
   process
   can run before exiting.
* After a search completes, the search process can wait for another
   search to
   start and the search process can be reused.
* When set to ?0? or ?1?: The process is never reused.
* When set to a negative value: There is no limit to the number of
   searches
   that a process can run.
* Has no effect on Windows if search_process_mode is not "auto".
* Default: 500

max_time_per_process = <number>
* Specifies the maximum time, in seconds, that a process can spend
   running
   searches.
* When a search process is allowed to run more than one search, limits
   how
   much time a process can accumulate running searches before the process
   must exit.
* When set to a negative value: There is no limit on the amount of time
   a
   search process can spend running.
* Has no effect on Windows if ?search_process_mode? is not "auto" or
   if ?max_searches_per_process? is set to 0 or 1.
* NOTE: A search can run longer than the value set for
   ?max_time_per_process?
   without being terminated. This setting ONLY prevents the process from
   being used to run additional searches after the maximum time is
   reached.
* Default: 300 (5 minutes)

process_max_age = <number>
* Specifies the maximum age, in seconds, for a search process.
* When a search process is allowed to run more than one search, a
   process
   is not reused if the process is older than the value specified.
* When set to a negative value: There is no limit on the the age of the
   search process.
* This setting includes the time that the process spends idle, which is
   different than "max_time_per_process" setting.
* Has no effect on Windows if ?search_process_mode? is not "auto" or
if $\text{max\_searches\_per\_process}$ is set to 0 or 1.

* NOTE: A search can run longer than the time set for $\text{process\_max\_age}$.

  without being terminated. This setting ONLY prevents that process from
  being used to run more searches after the search completes.

* Default: 7200 (120 minutes or 2 hours)

$\text{process\_min\_age\_before\_user\_change} = \text{<number>}$

* The minimum age, in seconds, of an idle process before using a process
  from a different user.

* When a search process is allowed to run more than one search, the
  system
  tries to reuse an idle process that last ran a search by the same
  Splunk
  user.

* If no such idle process exists, the system tries to use an idle
  process
  from a different user. The idle process from a different user must be
  idle for at least the value specified for the
  $\text{process\_min\_age\_before\_user\_change}$ setting.

* When set to $0$: Any idle process by any Splunk user can be reused.

* When set to a negative value: Only a search process by same Splunk
  user
  can be reused.

* Has no effect on Windows if $\text{search\_process\_mode}$ is not "auto" or
  if $\text{max\_searches\_per\_process}$ is set to 0 or 1.

* Default: 4

$\text{search\_process\_mode} = [\text{auto}|\text{traditional}|\text{debug} \text{<debugging\_command>}
  \text{<debugging\_args>}]$

* Controls how search processes are started.

* When set to $\text{traditional}$: Each search process is initialized
  completely
  from scratch.

* When set to $\text{debug}$: When set to a string beginning with "debug",
  searches are routed through the $\text{<debugging\_command>}$, where the user
  can
  "plug in" debugging tools.

* The $\text{<debugging\_command>}$ must reside in one of the following
  locations:

  * $\text{SPLUNK\_HOME/etc/system/bin/}$
  * $\text{SPLUNK\_HOME/etc/apps/$YOUR\_APP/bin/}$
  * $\text{SPLUNK\_HOME/bin/scripts/}$

* The $\text{<debugging\_args>}$ are passed, followed by the search command it
  would normally run, to $\text{<debugging\_command>}$

* For example, given the following setting:

  $\text{search\_process\_mode} = \text{debug}$

  $\text{SPLUNK\_HOME/bin/scripts/search\_debugger.sh} \ 5$

  A command similar to the following is run:

  $\text{SPLUNK\_HOME/bin/scripts/search\_debugger.sh} \ 5 \ \text{splunkd search}
  --id=... --maxbuckets=... --ttl=... [...]

* Default: auto

385
# Search reuse

This section contains settings for search reuse.

allow_reuse = <bool>
* Specifies whether to allow normally executed historical searches to be implicitly re-used for newer requests if the newer request allows it.
* Default: true

reuse_map_maxsize = <int>
* Maximum number of jobs to store in the reuse map.
* Default: 1000

# Splunk Analytics for Hadoop

This section contains settings for use with Splunk Analytics for Hadoop.

reduce_duty_cycle = <number>
* The maximum time to spend performing the reduce, as a fraction of total search time.
* Must be > 0.0 and < 1.0.
* Default: 0.25

reduce_freq = <integer>
* When the specified number of chunks is reached, attempt to reduce the intermediate results.
* When set to 0: Specifies that there is never an attempt to reduce the intermediate result.
* Default: 10

remote_reduce_limit = <unsigned long>
* The number of results processed by a streaming search before a reduce is forced.
* NOTE: this option applies only if the search is run with --runReduce=true  
  (currently only Splunk Analytics for Hadoop does this)
* When set to 0: Specifies that there is no limit.
* Default: 1000000

unified_search = <bool>
* Specifies if unified search is turned on for hunk archiving.
* Default: false

# Status

386
# This section contains settings for search status.

status_cache_size = <int>
* The number of status data for search jobs that splunkd can cache in RAM.
  This cache improves performance of the jobs endpoint.
* Default: 10000

status_period_ms = <int>
* The minimum amount of time, in milliseconds, between successive status/info.csv file updates.
* This setting ensures that search does not spend significant time just updating these files.
  * This is typically important for very large number of search peers.
  * It could also be important for extremely rapid responses from search peers, when the search peers have very little work to do.
* Default: 1000 (1 second)

# Timelines
# This section contains settings for timelines.

remote_event_download_finalize_pool = <int>
* Size of the pool, in threads, responsible for writing out the full remote events.
* Default: 5

remote_event_download_initialize_pool = <int>
* Size of the pool, in threads, responsible for initiating the remote event fetch.
* Default: 5

remote_event_download_local_pool = <int>
* Size of the pool, in threads, responsible for reading full local events.
* Default: 5

remote_timeline = [0|1]
* Specifies if the timeline can be computed remotely to enable better map/reduce scalability.
* Default: 1 (true)

remote_timeline_connection_timeout = <int>
* Connection timeout, in seconds, for fetching events processed by remote peer timeliner.
* Default: 5.

remote_timeline_fetchall = [0|1]
* When set to ?1? (true): Splunk fetches all events accessible through the
timeline from the remote peers before the job is considered done.
* Fetching of all events might delay the finalization of some searches,
  typically those running in verbose mode from the main Search view in
  Splunk Web.
* This potential performance impact can be mitigated by lowering the
  `max_events_per_bucket` settings.
* When set to 0 (false): The search peers might not ship all matching
  events to the search head, particularly if there is a very large
  number of them.
* Skipping the complete fetching of events back to the search head will
  result in prompt search finalization.
* Some events may not be available to browse in the UI.
* This setting does NOT affect the accuracy of search results computed
  by reporting searches.
* Default: 1 (true)

`remote_timeline_max_count = <int>`
* Maximum number of events to be stored per timeline bucket on each
  search peer.
* Default: 10000

`remote_timeline_max_size_mb = <int>`
* Maximum size of disk, in MB, that remote timeline events should take
  on each peer.
* If the limit is reached, a DEBUG message is emitted and should be
  visible in the job inspector or in messages.
* Default: 100

`remote_timeline_min_peers = <int>`
* Minimum number of search peers for enabling remote computation of
  timelines.
* Default: 1

`remote_timeline_parallel_fetch = <bool>`
* Specifies whether to connect to multiple peers at the same time when
  fetching remote events.
* Default: true

`remote_timeline_prefetch = <int>`
* Specifies the maximum number of full eventuate that each peer should
  proactively send at the beginning.
* Default: 100

`remote_timeline_receive_timeout = <int>`
* Receive timeout, in seconds, for fetching events processed by remote
  peer timeliner.
* Default: 10
remote_timeline_send_timeout = <int>
* Send timeout, in seconds, for fetching events processed by remote peer
timeliner.
* Default: 10

remote_timeline_thread = [0|1]
* Specifies whether to use a separate thread to read the full events from
remote peers if ?remote_timeline? is used and
?remote_timeline_fetchall?
  is set to ?true?.
  Has no effect if ?remote_timeline? or ?remote_timeline_fetchall? is
set to
  ?false?.
* Default: 1 (true)

remote_timeline_touchperiod = <number>
* How often, in seconds, while a search is running to touch remote
timeline
  artifacts to keep the artifacts from being deleted by the remote peer.
* When set to ?0?: The remote timelines are never touched.
* Fractional seconds are allowed.
* Default: 300 (5 minutes)

timeline_events_preview = <bool>
* When set to ?true?: Display events in the Search app as the events are
  scanned, including events that are in-memory and not yet committed,
  instead
  of waiting until all of the events are scanned to see the search results.
* You will not be able to expand the event information in the event
  viewer
  until events are committed.
* When set to ?false?: Events are displayed only after the events are
  committed (the events are written to the disk).
* This setting might increase disk usage to temporarily save uncommitted
  events while the search is running. Additionally, search performance
  might
  be impacted.
* Default: false

timeline_freq = <timespan> or <ratio>
* The minimum amount of time, in seconds, between timeline commits.
* If specified as a number < 1 (and > 0), minimum time between commits
  is
  computed as a ratio of the amount of time that the search has been
  running.
* Default: 0
# This section contains time to live (ttl) settings.

cache_ttl = <integer>
* The length of time, in seconds, to persist search cache entries.
* Default: 300 (5 minutes)

default_save_ttl = <integer>
* How long, in seconds, the ttl for a search artifact should be extended in
  response to the save control action.
* When set to 0, the system waits indefinitely.
* Default: 604800 (1 week)

failed_job_ttl = <integer>
* How long, in seconds, the search artifacts should be stored on disk after
  a job has failed. The ttl is computed relative to the modtime of the
  status.csv file of the job, if the file exists, or the modtime of the
  artifact directory for the search job.
* If a job is being actively viewed in the Splunk UI then the modtime of
  the status.csv file is constantly updated such that the reaper does not
  remove the job from underneath.
* Default: 86400 (1 day)

remote_ttl = <integer>
* How long, in seconds, the search artifacts from searches run in behalf of
  a search head should be stored on the indexer after completion.
* Default: 600 (10 minutes)

ttl = <integer>
* How long, in seconds, the search artifacts should be stored on disk after
  the job completes. The ttl is computed relative to the modtime of the
  status.csv file of the job, if the file exists, or the modtime of the
  artifact directory for the search job.
* If a job is being actively viewed in the Splunk UI then the modtime of
  the status.csv file is constantly updated such that the reaper does not
  remove the job from underneath.
* Default: 600 (10 minutes)

check_search_marker_done_interval = <integer>
* The amount of time, in seconds, that elapses between checks of search
  marker files, such as hot bucket markers and backfill complete markers.
* This setting is used to identify when the remote search process on the indexer completes processing all hot bucket and backfill portions of the search.
  * Default: 60

check_search_marker_sleep_interval = <integer>
  * The amount of time, in seconds, that the process will sleep between subsequent search marker file checks.
  * This setting is used to put the process into sleep mode periodically on the indexer, then wake up and check whether hot buckets and backfill portions of the search are complete.
  * Default: 1

srtemp_dir_ttl = <integer>
  * The time to live, in seconds, for the temporary files and directories within the intermediate search results directory tree.
  * These files and directories are located in $SPLUNK_HOME/var/run/splunk/srtemp.
  * Every 'srtemp_dir_ttl' seconds, the reaper removes files and directories within this tree to reclaim disk space.
  * The reaper measures the time to live through the newest file modification time within the directory.
  * When set to 0, the reaper does not remove any files or directories in this tree.
  * Default: 86400 (24 hours)

# Unsupported settings

enable_status_cache = <bool>
  * This is not a user tunable setting. Do not use this setting without working in tandem with Splunk personnel. This setting is not tested at non-default.
  * This controls whether the status cache is used, which caches information about search jobs (and job artifacts) in memory in main splunkd.
  * Normally this caching is enabled and assists performance. However, when using Search Head Pooling, artifacts in the shared storage location will be changed by other search heads, so this caching is disabled.
  * Explicit requests to jobs endpoints, eg /services/search/jobs/<sid> are always satisfied from disk, regardless of this setting.
  * Defaults to true; except in Search Head Pooling environments where it

391
defaults to false.

status_cache_in_memory_ttl = <positive integer>
* This is not a user tunable setting. Do not use this setting without working
  in tandem with Splunk personnel. This setting is not tested at non-default.
* This setting has no effect unless search head pooling is enabled, AND
  enable_status_cache has been set to true.
* If set, controls the number of milliseconds which a status cache entry
  may be used before it expires.
* Defaults to 60000, or 60 seconds.

# Unused settings
# This section contains settings that have been deprecated. These settings
# remain listed in this file for backwards compatibility.

max_bucket_bytes = <integer>
* This setting has been deprecated and has no effect.

rr_min_sleep_ms = <int>
* REMOVED. This setting is no longer used.

rr_max_sleep_ms = <int>
* REMOVED. This setting is no longer used.

rr_sleep_factor = <int>
* REMOVED. This setting is no longer used.

OTHER COMMAND SETTINGS

# This section contains the stanzas for the SPL commands, except for the
# search command, which is in separate section.

[anomalousvalue]

maxresultrows = <integer>
* Configures the maximum number of events that can be present in memory
  at one time.
* Default: searchresults::maxresultsrows (which is by default 50000)

maxvalues = <integer>
* Maximum number of distinct values for a field.
* Default: 100000

maxvaluesize = <integer>
* Maximum size, in bytes, of any single value (truncated to this size if larger).
* Default: 1000

[associate]

maxfields = <integer>
* Maximum number of fields to analyze.
* Default: 10000

maxvalues = <integer>
* Maximum number of values for any field to keep track of.
* Default: 10000

maxvaluesize = <integer>
* Maximum length of a single value to consider.
* Default: 1000

[autoregress]

maxp = <integer>
* Maximum number of events for auto regression.
* Default: 10000

maxrange = <integer>
* Maximum magnitude of range for p values when given a range.
* Default: 1000

[concurrency]

batch_search_max_pipeline = <int>
* Controls the number of search pipelines launched at the indexer during batch search.
* Increasing the number of search pipelines should help improve search performance but there will be an increase in thread and memory usage.
* This value applies only to searches that run on remote indexers.
* Default: 1

max_count = <integer>
* Maximum number of detected concurrencies.
* Default: 10000000

[correlate]

maxfields = <integer>
* Maximum number of fields to correlate.
* Default: 1000

[ctable]

* This stanza controls settings for the contingency command.
* Aliases for the contingency command are: ctable and counttable.

maxvalues = <integer>
* Maximum number of columns/rows to generate (the maximum number of distinct
 values for the row field and column field).
* Default: 1000

[discretize]

* This stanza contains the settings for the bin command.
* Aliases for the bin command are: bucket and discretize.

default_time_bins = <integer>
* When discretizing time for timechart or explicitly via bin, the default bins
 to use if no span or bins is specified.
* Default: 100

maxbins = <integer>
* Maximum number of bins to discretize into.
* If maxbins is not specified or = 0, it defaults to
 searchresults::maxresultrows
* Default: 50000

[findkeywords]

maxevents = <integer>
* Maximum number of events used by the findkeywords command and the
 Patterns tab.
* Default: 50000
[geomfilter]

enable_clipping = <boolean>
* Whether or not polygons are clipped to the viewport provided by the
  render client.
* Default: true

enable_generalization = <boolean>
* Whether or not generalization is applied to polygon boundaries to
  reduce
  point count for rendering.
* Default: true

[geostats]

filterstrategy = <integer>
* Controls the selection strategy on the geoviz map.
* Valid values are 1 and 2.

maxzoomlevel = <integer>
* Controls the number of zoom levels that geostats will cluster events
  on.

zl_0_gridcell_latspan = <float>
* Controls what is the grid spacing in terms of latitude degrees at the
  lowest zoom level, which is zoom-level 0.
* Grid-spacing at other zoom levels are auto created from this value by
  reducing by a factor of 2 at each zoom-level.

zl_0_gridcell_longspan = <float>
* Controls what is the grid spacing in terms of longitude degrees at the
  lowest zoom level, which is zoom-level 0
* Grid-spacing at other zoom levels are auto created from this value by
  reducing by a factor of 2 at each zoom-level.

[inputcsv]

mkdir_max_retries = <integer>
* Maximum number of retries for creating a tmp directory (with random
  name as
  subdir of SPLUNK_HOME/var/run/splunk)
* Default: 100
[iplocation]

db_path = <path>
* The absolute path to the GeoIP database in the MMDB format.
* The ?db_path? setting does not support standard Splunk environment
  variables such as SPLUNK_HOME.
* Default: The database that is included with the Splunk platform.

[join]

subsearch_maxout = <integer>
* Maximum result rows in output from subsearch to join against.
* Default: 50000

subsearch_maxtime = <integer>
* Maximum search time, in seconds, before auto-finalization of
  subsearch.
* Default: 60

subsearch_timeout = <integer>
* Maximum time, in seconds, to wait for subsearch to fully finish.
* Default: 120

[kmeans]

maxdatapoints = <integer>
* Maximum data points to do kmeans clusterings for.
* Default: 100000000 (100 million)

maxkrange = <integer>
* Maximum number of k values to iterate over when specifying a range.
* Default: 100

maxkvalue = <integer>
* Maximum number of clusters to attempt to solve for.
* Default: 1000

[lookup]

batch_index_query = <bool>
* Should non-memory file lookups (files that are too large) use batched
  queries
  to possibly improve performance?
* Default: true
batch_response_limit = <integer>
* When doing batch requests, the maximum number of matches to retrieve
  if more than this limit of matches would otherwise be retrieve, we
  will fall back to non-batch mode matching
* Default: 5000000

max_lookup_messages = <positive integer>
* If more than "max_lookup_messages" log entries are generated, additional
  entries will not be logged in info.csv. All entries will still be
  logged in search.log.

max_matches = <integer>
* Maximum matches for a lookup.
* Valid values range from 1 - 1000.
* Default: 1000

max_memtable_bytes = <integer>
* Maximum size, in bytes, of static lookup file to use an in-memory
  index for.
* Lookup files with size above max_memtable_bytes will be indexed on disk
* A large value results in loading large lookup files in memory leading to bigger
  process memory footprint.
* Caution must be exercised when setting this parameter to arbitrarily high values!
* Default: 10000000 (10MB)

max_reverse_matches = <integer>
* maximum reverse lookup matches (for search expansion)
* Default: 50

[metadata]

bucket_localize_max_lookahead = <int>
* This setting is only relevant when using remote storage.
  This setting specifies the maximum number of buckets the metadata command localizes
  for look-ahead purposes, in addition to the required bucket.
* Increasing this value can improve performance, at the cost of additional
  network/io/disk utilization.
* Valid values are 0-64. Any value larger than 64 will be set to 64.
  Other invalid values will be discarded and the default will be substituted.
* Default: 10

maxcount = <integer>
* The total number of metadata search results returned by the search head;
  after the maxcount is reached, any additional metadata results received from
  the search peers will be ignored (not returned).
* A larger number incurs additional memory usage on the search head.
* Default: 100000

maxresultrows = <integer>
* The maximum number of results in a single chunk fetched by the metadata
  command
* A smaller value will require less memory on the search head in setups
  with
  large number of peers and many metadata results, though, setting this too
  small will decrease the search performance.
* NOTE: Do not change unless instructed to do so by Splunk Support.
* Default: 10000

[mvexpand]

* This stanza allows for fine tuning of mvexpand search command.

max_mem_usage_mb = <non-negative integer>
* Overrides the default value for ?max_mem_usage_mb?.
* Limits the amount of RAM, in megabytes (MB), a batch of events or
  results will
  use in the memory of a search process.
* See definition in the [default] stanza for ?max_mem_usage_mb? for more
details.
* Default: 500

[mvcombine]

* This stanza allows for fine tuning of mvcombine search command.

max_mem_usage_mb = <non-negative integer>
* Overrides the default value for ?max_mem_usage_mb?.
* Limits the amount of RAM, in megabytes (MB), a batch of events or
  results will
  use in the memory of a search process.
* See definition in the [default] stanza for ?max_mem_usage_mb? for more
details.
* Default: 500
[outputlookup]

outputlookup_check_permission = <bool>
* Specifies whether the outputlookup command should verify that users have write permissions to CSV lookup table files.
* outputlookup_check_permission is used in conjunction with the transforms.conf setting check_permission.
* The system only applies outputlookup_check_permission to .csv lookup configurations in transforms.conf that have check_permission=true.
* You can set lookup table file permissions in the .meta file for each lookup file, or through the Lookup Table Files page in Settings. By default, only users who have the admin or power role can write to a shared CSV file.
* Default: false

[rare]

maxresultrows = <integer>
* Maximum number of result rows to create.
* If not specified, defaults to searchresults::maxresultrows
* Default: 50000

maxvalues = <integer>
* Maximum number of distinct field vector values to keep track of.
* Default: 100000

maxvaluesize = <integer>
* Maximum length of a single value to consider.
* Default: 1000

[set]

maxresultrows = <integer>
* The maximum number of results the set command will use from each result set to compute the required set operation.
* Default: 50000

[sort]

maxfiles = <integer>
* Maximum files to open at once. Multiple passes are made if the number
of result chunks exceeds this threshold.
* Default: 64.

[spath]

extract_all = <boolean>
* Controls whether we respect automatic field extraction when spath is invoked manually.
* If true, we extract all fields regardless of settings. If false, we only extract fields used by later search commands.
* Default: true

extraction_cutoff = <integer>
* For extract-all spath extraction mode, only apply extraction to the first <integer> number of bytes.
* Default: 5000

[stats/sistats]

approx_dc_threshold = <integer>
* When using approximate distinct count (i.e. estdc(<field>)) in stats/chart/timechart), do not use approximated results if the actual number of distinct values is less than this number.
* Default: 1000

dc_digest_bits = <integer>
* 2<integer> bytes will be size of digest used for approximating distinct count.
* Must be >= 8 (128B) and <= 16 (64KB)
* Default: 10 (equivalent to 1KB)

default_partitions = <int>
* Number of partitions to split incoming data into for parallel/multithreaded reduce
* Default: 1

list_maxsize = <int>
* Maximum number of list items to emit when using the list() function.
* stats/sistats
* Default: 100

maxmem_check_freq = <integer>
* How frequently, in rows, to check to see if we are exceeding the in
memory data structure size limit as specified by `max_mem_usage_mb`.
* Default: 50000

`maxresultrows = <integer>`
* Maximum number of rows allowed in the process memory.
* When the search process exceeds `max_mem_usage_mb` and `maxresultrows`, data is spilled out to the disk.
* If not specified, defaults to `searchresults::maxresultrows`
* Default: 50000

`max_stream_window = <integer>`
* For the `streamstats` command, the maximum allow window size.
* Default: 10000

`maxvalues = <integer>`
* Maximum number of values for any field to keep track of.
* When set to `0`: Specifies an unlimited number of values.
* Default: 0

`maxvaluesize = <integer>`
* Maximum length of a single value to consider.
* When set to `0`: Specifies an unlimited number of values.
* Default: 0

`max_valuemap_bytes = <integer>`
* For the `sistsats` command, the maximum encoded length of the valuemap, per result written out.
* If limit is exceeded, extra result rows are written out as needed.
* `0 = no limit per row`
* Default: 100000

`natural_sort_output = <bool>`
* Do a natural sort on the output of `stats` if output size is `<= maxresultrows`
* Natural sort means that we sort numbers numerically and non-numbers lexicographically
* Default: true

`partitions_limit = <int>`
* Maximum number of partitions to split into that can be specified via the 'partitions' option.
* When exceeded, the number of partitions is reduced to this limit.
* Default: 100

`perc_method = nearest-rank|interpolated`
* Which method to use for computing percentiles (and medians=50 percentile).
* nearest-rank picks the number with 0-based rank $R = \text{floor}(\text{percentile}/100)\times\text{count}$
* interpolated means given $F = (\text{percentile}/100)\times(\text{count}-1)$,
pick ranks $R_1 = \text{floor}(F)$ and $R_2 = \text{ceiling}(F)$.

Answer = $(R_2 \times (F - R_1)) + (R_1 \times (1 - (F - R_1)))$

* See wikipedia percentile entries on nearest rank and "alternative methods"
* Default: nearest-rank

perc_digest_type = rdigest|tdigest

* Which digest algorithm to use for computing percentiles (and medians=50 percentile).
  * rdigest picks the rdigest_k, rdigest_maxnodes and perc_method properties.
  * tdigest picks the tdigest_k and tdigest_max_buffer_size properties.
* Default: tdigest

sparkline_maxsize = <int>

* Maximum number of elements to emit for a sparkline
* Default: The value of the ?list_maxsize? setting

sparkline_time_steps = <time-step-string>

* Specify a set of time steps in order of decreasing granularity. Use an integer and one of the following time units to indicate each step.
  * s = seconds
  * m = minutes
  * h = hours
  * d = days
  * month
* A time step from this list is selected based on the <sparkline_maxsize> setting.
  * The lowest <sparkline_time_steps> value that does not exceed the maximum number of bins is used.
* Default: 1s,5s,10s,30s,1m,5m,10m,30m,1h,1d,1month

NOTE: The following are rdigest and tdigest settings.

rdigest is a data structure used to compute approximate order statistics (such as median and percentiles) using sublinear space.

rdigest_k = <integer>

* rdigest compression factor
* Lower values mean more compression
* After compression, number of nodes guaranteed to be greater than or equal to
  11 times k.
* Must be greater than or equal to 2.
* Default: 100

\[ \text{rdigest\_maxnodes} = \text{<integer>} \]
* Maximum rdigest nodes before automatic compression is triggered.
* When set to ?1??: Specifies to automatically configure based on k value.
* Default: 1

\[ \text{tdigest\_k} = \text{<integer>} \]
* tdigest compression factor
* Higher values mean less compression, more mem usage, but better accuracy.
* Must be greater than or equal to 1.
* Default: 50

\[ \text{tdigest\_max\_buffer\_size} = \text{<integer>} \]
* Maximum number of elements before automatic reallocation of buffer storage is triggered.
* Smaller values result in less memory usage but is slower.
* Very small values (<100) are not recommended as they will be very slow.
* Larger values help performance up to a point after which it actually hurts performance.
* Recommended range is around 10tdigest\_k to 30tdigest\_k.
* Default: 1000

\[ \text{maxresultrows} = \text{<integer>} \]
* Maximum number of result rows to create.
* If not specified, defaults to searchresults::maxresultrows.
* Default: 50000

\[ \text{maxvalues} = \text{<integer>} \]
* Maximum number of distinct field vector values to keep track of.
* Default: 100000

\[ \text{maxvaluesize} = \text{<integer>} \]
* Maximum length of a single value to consider.
* Default: 1000
[transactions]

maxopentxn = <integer>
* Specifies the maximum number of not yet closed transactions to keep in the
  open pool before starting to evict transactions.
* Default: 5000

maxopenevents = <integer>
* Specifies the maximum number of events (which are) part of open transactions
  before transaction eviction starts happening, using LRU policy.
* Default: 100000

[tscollect]

squashcase = <boolean>
* The default value of the 'squashcase' argument if not specified by the command
* Default: false

keepresults = <boolean>
* The default value of the 'keepresults' argument if not specified by the command
* Default: false

optimize_max_size_mb = <unsigned int>
* The maximum size in megabytes of files to create with optimize
  Specify 0 for no limit (may create very large tsidx files)
* Default: 1024

[tstats]

allow_old_summaries = <boolean>
* The default value of 'allow_old_summaries' arg if not specified by the command
* When running tstats on an accelerated datamodel, allow_old_summaries=false
  ensures we check that the datamodel search in each bucket's summary metadata
  is considered up to date with the current datamodel search. Only summaries
  that are considered up to date will be used to deliver results.
* The allow_old_summaries=true attribute overrides this behavior and will deliver results
  even from bucket summaries that are considered out of date with the
current
datamodel.
* Default: false

apply_search_filter = <boolean>
* Controls whether we apply role-based search filters when users run
tstats on
  normal index data
* Note: we never apply search filters to data collected with tscollect
or
datamodel acceleration
* Default: true

bucket_localize_max_lookahead = <int>
* This setting is only relevant when using remote storage.
* Specifies the maximum number of buckets the tstats command localizes
  for
  look-ahead purposes, in addition to the required bucket.
* Increasing this value can improve performance, at the cost of
  additional
  network/io/disk utilization.
* Valid values are 0-64. Any value larger than 64 will be set to 64.
Other
  invalid values will be discarded and the default will be substituted.
* Default: 10

chunk_size = <unsigned int>
* ADVANCED: The default value of 'chunk_size' arg if not specified by
  the command
* This argument controls how many events are retrieved at a time within
  a
  single TSIDX file when answering queries
* Consider lowering this value if tstats queries are using too much
  memory
  (cannot be set lower than 10000)
* Larger values will tend to cause more memory to be used (per search)
  and
  might have performance benefits.
* Smaller values will tend to reduce performance and might reduce memory
  used
  (per search).
* Altering this value without careful measurement is not advised.
* Default: 10000000

summariesonly = <boolean>
* The default value of 'summariesonly' arg if not specified by the
  command
* When running tstats on an accelerated datamodel, summariesonly=false
  implies
  a mixed mode where we will fall back to search for missing TSIDX data
* summariesonly=true overrides this mixed mode to only generate results
  from
TSIDX data, which may be incomplete
* Default: false

warn_on_missing_summaries = <boolean>
* ADVANCED: Only meant for debugging summariesonly=true searches on accelerated datamodels.
* When true, search will issue a warning for a tstats summariesonly=true search for the following scenarios:
  a) If there is a non-hot bucket that has no corresponding datamodel acceleration summary whatsoever.
  b) If the bucket's summary does not match with the current datamodel acceleration search.
* Default: false

[typeahead]

cache_ttl_sec = <integer>
* How long, in seconds, the typeahead cached results are valid.
* Default 300

fetch_multiplier = <integer>
* A multiplying factor that determines the number of terms to fetch from the index, fetch = fetch_multiplier x count.
* Default: 50

max_concurrent_per_user = <integer>
* The maximum number of concurrent typeahead searches per user. Once this maximum is reached only cached typeahead results might be available.
* Default: 3

maxcount = <integer>
* Maximum number of typeahead results to find.
* Default: 1000

min_prefix_length = <integer>
* The minimum length of the string prefix after which to provide typeahead.
* Default: 1

use_cache = [0|1]
* Specifies whether the typeahead cache will be used if use_cache is not specified in the command line or endpoint.
* Default: true or 1
[typer]

maxlen = <int>
* In eventtyping, pay attention to first <int> characters of any attribute
  (such as _raw), including individual tokens. Can be overridden by
  supplying the typer operator with the argument maxlen (for example,
  "|typer maxlen=300").
* Default: 10000

[xyseries]

* This stanza allows for fine tuning of xyseries search command.

max_mem_usage_mb = <non-negative integer>
* Overrides the default value for ?max_mem_usage_mb?
* See definition in [default] max_mem_usage_mb for more details

GENERAL SETTINGS

# This section contains the stanzas for a variety of general settings.

[authtokens]

expiration_time = <integer>
* Expiration time, in seconds, of auth tokens.
* Default: 3600 (60 minutes)

[auto_summarizer]

allow_event_summarization = <bool>
* Whether auto summarization of searches whose remote part returns events
  rather than results will be allowed.
* Default: false

cache_timeout = <integer>
* The minimum amount of time, in seconds, to cache auto summary details
  and search hash codes.
* The cached entry expires randomly between cache_timeout and 2*cache_timeout value.
  * Default: 600 (10 minutes)

detailed_dashboard = <bool>
* Turn on/off the display of both normalized and regular summaries in the Report Acceleration summary dashboard and details.
  * Default: false

maintenance_period = <integer>
* The period of time, in seconds, that the auto summarization maintenance happens
  * Default: 1800 (30 minutes)

max_run_stats = <int>
* Maximum number of summarization run statistics to keep track and expose via REST.
  * Default: 48

max_verify_buckets = <int>
* When verifying buckets, stop after verifying this many buckets if no failures have been found
  * 0 means never
  * Default: 100

max_verify_bucket_time = <int>
* Maximum time, in seconds, to spend verifying each bucket.
  * Default: 15

max_verify_ratio = <number>
* Maximum fraction of data in each bucket to verify
  * Default: 0.1 (10%)

max_verify_total_time = <int>
* Maximum total time in seconds to spend doing verification, regardless if any buckets have failed or not
  * When set to ?0?: Specifies no limit.
  * Default: 0

normalized_summaries = <bool>
* Turn on/off normalization of report acceleration summaries.
  * Default: true

return_actions_with_normalized_ids = [yes|no|fromcontext]
* Report acceleration summaries are stored under a signature/hash which can be regular or normalized.
* Normalization improves the re-use of pre-built summaries but is not supported before 5.0. This config will determine the default value of how normalization works (regular/normalized)
* When set to ?fromcontext?: Specifies that the end points and summaries would be operating based on context.
* Normalization strategy can also be changed via admin/summarization REST calls with the "use_normalization" parameter which can take the values "yes"/"no"/"fromcontext"
* Default: fromcontext

search_2_hash_cache_timeout = <integer>
* The amount of time, in seconds, to cache search hash codes
* Default: The value of the ?cache_timeout? setting, which by default is 600 (10 minutes)

shc_accurate_access_counts = <bool>
* Only relevant if you are using search head clustering
* Turn on/off to make acceleration summary access counts accurate on the captain.
* by centralizing

verify_delete = <bool>
* Should summaries that fail verification be automatically deleted? 
* Default: false

[export]

add_offset = <bool>
* Add an offset/row number to JSON streaming output
* Default: true

add_timestamp = <bool>
* Add a epoch time timestamp to JSON streaming output that reflects the time the results were generated/retrieved
* Default: false

[extern]

perf_warn_limit = <integer>
* Warn when external scripted command is applied to more than this many events
* When set to ?0?: Specifies for no message (message is always INFO level)
* Default: 10000
[http_input]

max_number_of_tokens = <unsigned int>
* The maximum number of tokens reported by logging input metrics.
* Default: 10000

max_content_length = <integer>
* The maximum length, in bytes, of HTTP request content that is accepted by the HTTP Event Collector server.
* Default: 838860800 (~ 800 MB)

max_number_of_ack_channel = <integer>
* The maximum number of ACK channels accepted by HTTP Event Collector server.
* Default: 1000000 (~ 1 million)

max_number_of_acked_requests_pending_query = <integer>
* The maximum number of ACKed requests pending query on HTTP Event Collector server.
* Default: 10000000 (~ 10 million)

max_number_of_acked_requests_pending_query_per_ack_channel = <integer>
* The maximum number of ACKed requested pending query per ACK channel on HTTP Event Collector server.
* Default: 1000000 (~ 1 million)

metrics_report_interval = <integer>
* The interval, in seconds, of logging input metrics report.
* Default: 60 (1 minute)

[indexpreview]

max_preview_bytes = <integer>
* Maximum number of bytes to read from each file during preview
* Default: 2000000 (2 MB)

max_results_perchunk = <integer>
* Maximum number of results to emit per call to preview data generator
* Default: 2500

soft_preview_queue_size = <integer>
* Loosely-applied maximum on number of preview data objects held in memory
* Default: 100
[inputproc]

file_tracking_db_threshold_mb = <integer>
* This setting controls the trigger point at which the file tracking db
(also
commonly known as the "fishbucket" or btree) rolls over. A new
database is
created in its place. Writes are targeted at new db. Reads are first
targeted at new db, and we fall back to old db for read failures. Any
reads
served from old db successfully will be written back into new db.
* MIGRATION NOTE: if this setting doesn't exist, the initialization code in
splunkd triggers an automatic migration step that reads in the current value
for "maxDataSize" under the "_thefishbucket" stanza in indexes.conf
and
writes this value into etc/system/local/limits.conf.

learned_sourcetypes_limit = <0 or positive integer>
* Limits the number of entries added to the learned app for performance
reasons.
* If nonzero, limits two properties of data added to the learned app by the
file classifier. (Code specific to monitor:: stanzas that
auto-determines
sourcetypes from content.)
* The number of sourcetypes added to the learned app's props.conf file
will
be limited to approximately this number.
* The number of file-content fingerprints added to the learned app's
sourcetypes.conf file will be limited to approximately this number.
* The tracking for uncompressed and compressed files is done separately,
so in
some cases this value may be exceeded.
* This limit is not the recommended solution for auto-identifying
sourcetypes.
The usual best practices are to set sourcetypes in input stanzas, or
alternatively to apply them based on filename pattern in props.conf
[source::<pattern>] stanzas.
* Default: 1000

max_fd = <integer>
* Maximum number of file descriptors that a ingestion pipeline in Splunk
will keep open, to capture any trailing data from files that are
written
to very slowly.
* Note that this limit will be applied per ingestion pipeline. For more
information about multiple ingestion pipelines see
parallelIngestionPipelines
in the server.conf.spec file.
* With N parallel ingestion pipelines the maximum number of file descriptors that can be open across all of the ingestion pipelines will be N * max_fd.
  * Default: 100

monitornohandle_max_heap_mb = <integer>
* Controls the maximum memory used by the Windows-specific modular input MonitorNoHandle in user mode.
  * The memory of this input grows in size when the data being produced by applications writing to monitored files comes in faster than the Splunk system can accept it.
  * When set to 0, the heap size (memory allocated in the modular input) can grow without limit.
  * If this size is limited, and the limit is encountered, the input will drop some data to stay within the limit.
  * Default: 0

tailing_proc_speed = <integer>
  * REMOVED. This setting is no longer used.

monitornohandle_max_driver_mem_mb = <integer>
  * Controls the maximum NonPaged memory used by the Windows-specific kernel driver of modular input MonitorNoHandle.
    * The memory of this input grows in size when the data being produced by applications writing to monitored files comes in faster than the Splunk system can accept it.
    * When set to 0, the NonPaged memory size (memory allocated in the kernel driver of modular input) can grow without limit.
    * If this size is limited, and the limit is encountered, the input will drop some data to stay within the limit.
    * Default: 0

monitornohandle_max_driver_records = <integer>
  * Controls memory growth by limiting the maximum in-memory records stored by the kernel module of Windows-specific modular input MonitorNoHandle.
    * When monitornohandle_max_driver_mem_mb is set to > 0, this config is ignored.
    * monitornohandle_max_driver_mem_mb and monitornohandle_max_driver_records are mutually exclusive.
    * If the limit is encountered, the input will drop some data to stay within the limit.
    * Defaults to 500.
time_before_close = <integer>
* MOVED. This setting is now configured per-input in inputs.conf.
* Specifying this setting in limits.conf is DEPRECATED, but for now will
  override the setting for all monitor inputs.

[journal_compression]

threads = <integer>
* Specifies the maximum number of indexer threads which will be work on
  compressing hot bucket journal data.
* This setting does not typically need to be modified.
* Default: The number of CPU threads of the host machine

[kv]

avg_extractor_time = <integer>
* Maximum amount of CPU time, in milliseconds, that the average (over
  search
  results) execution time of a key-value pair extractor will be allowed
  to take
  before warning. Once the average becomes larger than this amount of
  time a
  warning will be issued
* Default: 500 (.5 seconds)

limit = <integer>
* The maximum number of fields that an automatic key-value field
  extraction
  (auto kv) can generate at search time.
* If search-time field extractions are disabled (KV_MODE=none in
  props.conf)
  then this setting determines the number of index-time fields that will
  be
  returned.
* The summary fields 'host', 'index', 'source', 'sourcetype',
  'eventtype',
  'linecount', 'splunk_server', and 'splunk_server_group' do not count
  against
  this limit and will always be returned.
* Increase this setting if, for example, you have indexed data with a
  large
  number of columns and want to ensure that searches display all fields
  from
  the data.
* Default: 100

maxchars = <integer>
maxcols = <integer>
* When non-zero, the point at which kv should stop creating new fields.
* Default: 512

max_extractor_time = <integer>
* Maximum amount of CPU time, in milliseconds, that a key-value pair extractor
  will be allowed to take before warning. If the extractor exceeds this execution time on any event a warning will be issued
* Default: 1000 (1 second)

[kvstore]

max_accelerations_per_collection = <unsigned int>
* The maximum number of accelerations that can be assigned to a single collection
* Valid values range from 0 to 50
* Default: 10

max_documents_per_batch_save = <unsigned int>
* The maximum number of documents that can be saved in a single batch
* Default: 1000

max_fields_per_acceleration = <unsigned int>
* The maximum number of fields that can be part of a compound acceleration
  (i.e. an acceleration with multiple keys)
* Valid values range from 0 to 50
* Default: 10

max_queries_per_batch = <unsigned int>
* The maximum number of queries that can be run in a single batch
* Default: 1000

max_rows_in_memory_per_dump = <unsigned int>
* The maximum number of rows in memory before flushing it to the CSV projection
  of KVStore collection.
* Default: 200

max_rows_per_query = <unsigned int>
* The maximum number of rows that will be returned for a single query to a collection.
* If the query returns more rows than the specified value, then returned result set will contain the number of rows specified in this value.
* Default: 50000
max_size_per_batch_result_mb = <unsigned int>
* The maximum size, in megabytes (MB), of the result set from a set of batched queries
* Default: 100

max_size_per_batch_save_mb = <unsigned int>
* The maximum size, in megabytes (MB), of a batch save query.
* Default: 50

max_size_per_result_mb = <unsigned int>
* The maximum size, in megabytes (MB), of the result that will be returned for a single query to a collection.
* Default: 50

max_threads_per_outputlookup = <unsigned int>
* The maximum number of threads to use during outputlookup commands on KVStore
* If the value is 0 the thread count will be determined by CPU count
* Default: 1

[input_channels]

max_inactive = <integer>
* Internal setting, do not change unless instructed to do so by Splunk Support.

lowater_inactive = <integer>
* Internal setting, do not change unless instructed to do so by Splunk Support.

inactive_eligibility_age_seconds = <integer>
* Internal setting, do not change unless instructed to do so by Splunk Support.

[ldap]

allow_multiple_matching_users = <bool>
* This controls whether we allow login when we find multiple entries with the same value for the username attribute
* When multiple entries are found, we choose the first user DN lexicographically
* Setting this to false is more secure as it does not allow any ambiguous login, but users with duplicate entries will not be able to login.
* Default: true

max_users_to_precache = <unsigned integer>
* The maximum number of users we will attempt to pre-cache from LDAP after reloading auth
  * Set this to 0 to turn off pre-caching

**[metrics]**

interval = <integer>
* Number of seconds between logging splunkd metrics to metrics.log.
  * Minimum of 10.
  * Default: 30

maxseries = <integer>
* The number of series to include in the per_x_thruput reports in metrics.log.
  * Default: 10

**[metrics:tcpin_connections]**

aggregate_metrics = [true|false]
* For each splunktcp connection from forwarder, splunk logs metrics information
  every metrics interval.
  * When there are large number of forwarders connected to indexer, the amount of
    information logged can take lot of space in metrics.log. When set to true, it
    will aggregate information across each connection and report only once per
    metrics interval.
  * Default: false

suppress_derived_info = [true|false]
* For each forwarder connection, _tcp_Bps, _tcp_KBps, _tcp_avg_thruput, _tcp_Kprocessed is logged in metrics.log.
  * This can be derived from kb. When set to true, the above derived info will
    not be emitted.
  * Default: false

**[pdf]**

max_rows_per_table = <unsigned int>
* The maximum number of rows that will be rendered for a table within integrated PDF rendering
  * Default: 1000
render_endpoint_timeout = <unsigned int>
* The number of seconds after which the pdfgen render endpoint will
  timeout if
  it has not yet finished rendering the PDF output
* Default: 3600 (60 minutes)

[realtime]

# Default options for indexer support of real-time searches
# These can all be overridden for a single search via REST API arguments

alerting_period_ms = <int>
* This limits the frequency that we will trigger alerts during a
  realtime search.
* A value of 0 means unlimited and we will trigger an alert for every
  batch of
  events we read in dense realtime searches with expensive alerts this
  can
  overwhelm the alerting system.
* Precedence: Searchhead
* Default: 0

blocking = [0|1]
* Specifies whether the indexer should block if a queue is full.
* Default: false

default_backfill = <bool>
* Specifies if windowed real-time searches should backfill events
* Default: true

enforce_time_order = <bool>
* Specifies if real-time searches should ensure that events are sorted
  in
  ascending time order (the UI will automatically reverse the order that
  it
  display events for real-time searches so in effect the latest events
  will be
  first)
* Default: true

indexfilter = [0|1]
* Specifies whether the indexer should prefilter events for efficiency.
* Default: 1 (true)

indexed_realtime_update_interval = <int>
* When you run an indexed realtime search, the list of searchable
  buckets
  needs to be updated. If the Splunk software is installed on a cluster,
  the list of allowed primary buckets is refreshed. If not installed on
  a cluster, the list of buckets, including any new hot buckets are
refreshed. This setting controls the interval for the refresh. The setting must be less than the "indexed_realtime_disk_sync_delay" setting. If your realtime buckets transition from new to warm in less time than the value specified for the "indexed_realtime_update_interval" setting, data will be skipped by the realtime search in a clustered environment.
* Precedence: Indexers
* Default: 30

indexed_realtime_cluster_update_interval = <int>
* This setting is deprecated. Use the "indexed_realtime_update_interval" setting instead.
* While running an indexed realtime search, if we are on a cluster we need to update the list of allowed primary buckets. This controls the interval that we do this. And it must be less than the indexed_realtime_disk_sync_delay. If your buckets transition from Brand New to warm in less than this time indexed realtime will lose data in a clustered environment.
* Precedence: Indexers
* Default: 30

indexed_realtime_default_span = <int>
* An indexed realtime search is made up of many component historical searches that by default will span this many seconds. If a component search is not completed in this many seconds the next historical search will span the extra seconds. To reduce the overhead of running an indexed realtime search you can change this span to delay longer before starting the next component historical search.
* Precedence: Indexers
* Default: 1

indexed_realtime_disk_sync_delay = <int>
* This settings controls the number of seconds to wait for disk flushes to finish when using indexed/continuous/pseudo realtime search so that we see all of the data.
* After indexing there is a non-deterministic period where the files on disk when opened by other programs might not reflect the latest flush to
disk,
  particularly when a system is under heavy load.
* Precedence: SearchHead overrides Indexers
* Default: 60

indexed_realtime_maximum_span = <int>
* While running an indexed realtime search, if the component searches regularly
  take longer than indexed_realtime_default_span seconds, then indexed realtime
  search can fall more than indexed_realtime_disk_sync_delay seconds behind
  realtime. Use this setting to set a limit after which we will drop data to
  return back to catch back up to the specified delay from realtime, and only
  search the default span of seconds.
* Precedence: API overrides SearchHead overrides Indexers
* Default: 0 (unlimited)

indexed_realtime_use_by_default = <bool>
* Should we use the indexedRealtime mode by default
* Precedence: SearchHead
* Default: false

local_connect_timeout = <int>
* Connection timeout, in seconds, for an indexer's search process when
  connecting to that indexer's splunkd.
* Default: 5

local_receive_timeout = <int>
* Receive timeout, in seconds, for an indexer's search process when
  connecting to that indexer's splunkd.
* Default: 5

local_send_timeout = <int>
* Send timeout, in seconds, for an indexer's search process when
  connecting
  to that indexer's splunkd.
* Default: 5

max_blocking_secs = <int>
* Maximum time, in seconds, to block if the queue is full (meaningless
  if blocking = false)
* 0 means no limit
* Default: 60

queue_size = <int>
* Size of queue for each real-time search (must be >0).
* Default: 10000
[restapi]

maxresultrows = <integer>
* Maximum result rows to be returned by /events or /results getters from REST API.
  * Default: 50000

jobscontentmaxcount = <integer>
* Maximum length of a property in the contents dictionary of an entry from /jobs getter from REST API
  * Value of 0 disables truncation
  * Default: 0

time_format_reject = <regular expression>
* HTTP parameters for time_format and output_time_format which match this regex will be rejected (blacklisted).
  * The regex will be satisfied by a substring match anywhere in the parameter.
  * Intended as defense-in-depth against XSS style attacks against browser users by crafting specially encoded URLs for them to access splunkd.
  * If unset, all parameter strings will be accepted.
  * To disable this check entirely, set the value to empty.
    * Example of disabling: time_format_reject =
  * Default: [<>!] , which means that the less-than '<', greater-than '>', and exclamation point '!' are not allowed.

[reversedns]

rdnsMaxDutyCycle = <integer>
* Generate diagnostic WARN in splunkd.log if reverse dns lookups are taking more than this percent of time
  * Range 0-100
  * Default: 10

[sample]

maxsamples = <integer>
* Default: 10000

maxtotalsamples = <integer>
* Default: 100000
[scheduler]

action_execution_threads = <integer>
* Number of threads to use to execute alert actions, change this number if your
  alert actions take a long time to execute.
* This number is capped at 10.
* Default: 2

actions_queue_size = <integer>
* The number of alert notifications to queue before the scheduler starts
  blocking, set to 0 for infinite size.
* Default: 100

actions_queue_timeout = <integer>
* The maximum amount of time, in seconds, to block when the action queue
  size is
  full.
* Default: 30

alerts_expire_period = <integer>
* The amount of time, in seconds, between expired alert removal
* This period controls how frequently the alerts list is scanned, the only
  benefit from reducing this is better resolution in the number of
  alerts fired
  at the savedsearch level.
* Change not recommended.
* Default: 120

alerts_max_count = <integer>
* Maximum number of unexpired alerts information to keep for the alerts
  manager, when this number is reached Splunk will start discarding the
  oldest
  alerts.
* Default: 50000

alerts_max_history = <integer>[s|m|h|d]
* Maximum time to search in the past for previously triggered alerts.
* splunkd uses this property to populate the Activity -> Triggered
  Alerts
  page at startup.
* Values greater than the default may cause slowdown.
* Relevant units are: s, sec, second, secs, seconds, m, min, minute,
  mins,
  minutes, h, hr, hour, hrs, hours, d, day, days.
* Default: 7d

alerts_scoping = host|splunk_server|all
* Determines the scoping to use on the search to populate the triggered
alerts
page. Choosing splunk_server will result in the search query
using splunk_server=local, host will result in the search query using
host=<search-head-host-name>, and all will have no scoping added to
the
search query.
* Default: splunk_server

auto_summary_perc = <integer>
* The maximum number of concurrent searches to be allocated for auto
summarization, as a percentage of the concurrent searches that the
scheduler
can run.
* Auto summary searches include:
  * Searches which generate the data for the Report Acceleration
feature.
  * Searches which generate the data for Data Model acceleration.
  * Note: user scheduled searches take precedence over auto summary
searches.
  * Default: 50

auto_summary_perc.<n> = <integer>
auto_summary_perc.<n>.when = <cron string>
* The same as auto_summary_perc but the value is applied only when the
cron
  string matches the current time. This allows auto_summary_perc to
have
different values at different times of day, week, month, etc.
* There may be any number of non-negative <n> that progress from least
  specific
to most specific with increasing <n>.
* If either these settings aren't provided at all or no "when" matches
the
  current time, the value falls back to the non-<n> value of
auto_summary_perc.

concurrency_message_throttle_time = <int>[s|m|h|d]
* Amount of time controlling throttling between messages warning about
scheduler
  concurrency limits.
* Relevant units are: s, sec, second, secs, seconds, m, min, minute,
  mins,
  minutes, h, hr, hour, hrs, hours, d, day, days.
* Default: 10m

introspection_lookback = <duration-specifier>
* The amount of time to "look back" when reporting introspection
statistics.
* For example: what is the number of dispatched searches in the last 60
  minutes?
* Use [<int>]<unit> to specify a duration; a missing <int> defaults to
1. Relevant units are: m, min, minute, mins, minutes, h, hr, hour, hrs, hours, d, day, days, w, week, weeks. For example: "5m" = 5 minutes, "1h" = 1 hour.
   * Default: 1h

max_action_results = <integer>
* The maximum number of results to load when triggering an alert action.
  * Default: 50000

max_continuous_search_lookback = <duration-specifier>
* The maximum amount of time to run missed continuous scheduled searches for
  once Splunk comes back up in the event it was down.
  * Use [<int>]<unit> to specify a duration; a missing <int> defaults to 1.
  * Relevant units are: m, min, minute, mins, minutes, h, hr, hour, hrs, hours, d, day, days, w, week, weeks, mon, month, months.
  * For example: "5m" = 5 minutes, "1h" = 1 hour.
  * A value of 0 means no lookback.
  * Default: 24h

max_lock_files = <int>
* The number of most recent lock files to keep around.
  * This setting only applies in search head pooling.

max_lock_file_ttl = <int>
* Time, in seconds, that must pass before reaping a stale lock file.
  * Only applies in search head pooling.

max_per_result_alerts = <int>
* Maximum number of alerts to trigger for each saved search instance (or real-time results preview for RT alerts)
  * Only applies in non-digest mode alerting. Use 0 to disable this limit
  * Default: 500

max_per_result_alerts_time = <integer>
* Maximum number of time, in seconds, to spend triggering alerts for each saved search instance (or real-time results preview for RT alerts)
  * Only applies in non-digest mode alerting. Use 0 to disable this limit.
  * Default: 300 (5 minutes)

max_searches_perc = <integer>
* The maximum number of searches the scheduler can run, as a percentage of the maximum number of concurrent searches, see [search]
  * Default: 50
max_searches_perc.<n> = <integer>
max_searches_perc.<n>.when = <cron string>
* The same as max_searches_perc but the value is applied only when the cron
  string matches the current time. This allows max_searches_perc to have
  different values at different times of day, week, month, etc.
* There may be any number of non-negative <n> that progress from least specific
  to most specific with increasing <n>.
* The scheduler looks in reverse-<n> order looking for the first match.
* If either these settings aren't provided at all or no "when" matches the
current time, the value falls back to the non-<n> value of max_searches_perc.

persistance_period = <integer>
* The period, in seconds, between scheduler state persistance to disk. The
  scheduler currently persists the suppression and fired-unexpired alerts to
  disk.
* This is relevant only in search head pooling mode.
* Default: 30

priority_runtime_factor = <double>
* The amount to scale the priority runtime adjustment by.
* Every search's priority is made higher (worse) by its typical running time.
  Since many searches run in fractions of a second and the priority is
  integral, adjusting by a raw runtime wouldn't change the result;
  therefore,
  it's scaled by this value.
* Default: 10

priority_skipped_factor = <double>
* The amount to scale the skipped adjustment by.
* A potential issue with the priority_runtime_factor is that now
  longer-running searches may get starved. To balance this out, make a search's
  priority lower (better) the more times it's been skipped. Eventually, this
  adjustment will outweigh any worse priority due to a long runtime. This value
  controls how quickly this happens.
* Default: 1

dispatch_retry_delay = <unsigned int>
* The amount of time, in seconds, to delay retrying a scheduled search that
failed to dispatch (usually due to hitting concurrency limits).
* Maximum value: 30
* Default: 0

saved_searches_disabled = <bool>
* Whether saved search jobs are disabled by the scheduler.
* Default: false

scheduled_view_timeout = <int>[s|m|h|d]
* The maximum amount of time that a scheduled view (pdf delivery) would be allowed to render
* Relevant units are: s, sec, second, secs, seconds, m, min, minute, mins, minutes, h, hr, hour, hrs, hours, d, day, days.
* Default: 60m

shc_role_quota_enforcement = <bool>
* When this attribute is enabled, the search head cluster captain enforces user-role quotas for scheduled searches globally (cluster-wide).
* A given role can have (n * number_of_members) searches running cluster-wide, where n is the quota for that role as defined by srchJobsQuota and rtSrchJobsQuota on the captain and number_of_members include the members capable of running scheduled searches.
* Scheduled searches will therefore not have an enforcement of user role quota on a per-member basis.
* Role-based disk quota checks (srchDiskQuota in authorize.conf) can be enforced only on a per-member basis. These checks are skipped when shc_role_quota_enforcement is enabled.
* Quota information is conveyed from the members to the captain. Network delays can cause the quota calculation on the captain to vary from the actual values in the members and may cause search limit warnings. This should clear up as the information is synced.
* Default: false

shc_syswide_quota_enforcement = <bool>
* When this is enabled, Maximum number of concurrent searches is enforced globally (cluster-wide) by the captain for scheduled searches. Concurrent searches include both scheduled searches and ad hoc searches.
* This is (n * number_of_members) where n is the max concurrent searches per node (see max_searches_per_cpu for a description of how this is computed) and number_of_members include members capable of running scheduled
searches.
* Scheduled searches will therefore not have an enforcement of
  instance-wide
  concurrent search quota on a per-member basis.
* Note that this does not control the enforcement of the scheduler
  quota.
  For a search head cluster, that is defined as
  \[(\text{max_searches_perc} \times \text{number_of_members})\]
  and is always enforced globally on the captain.
* Quota information is conveyed from the members to the captain. Network
  delays
  can cause the quota calculation on the captain to vary from the actual
  values
  in the members and may cause search limit warnings. This should clear
  up as
  the information is synced.
* Default: false

\text{shc\_local\_quota\_check = <bool>}
* DEPRECATED. Local (per-member) quota check is enforced by default.
* To disable per-member quota checking, enable one of the cluster-wide
  quota
  checks (\text{shc\_role\_quota\_enforcement} or \text{shc\_syswide\_quota\_enforcement}).
* For example, setting \text{shc\_role\_quota\_enforcement=true} turns off local
  role
  quota enforcement for all nodes in the cluster and is enforced
  cluster-wide
  by the captain.

\text{shp\_dispatch\_to\_slave = <bool>}
* By default the scheduler should distribute jobs throughout the pool.
* Default: true

\text{search\_history\_load\_timeout = <duration-specifier>}
* The maximum amount of time to defer running continuous scheduled
  searches
  while waiting for the KV Store to come up in order to load historical
  data.
  This is used to prevent gaps in continuous scheduled searches when
  splunkd
  was down.
  * Use [<int>]<unit> to specify a duration; a missing <int> defaults to
    1.
  * Relevant units are: s, sec, second, secs, seconds, m, min, minute,
    mins,
    minutes.
  * For example: "60s" = 60 seconds, "5m" = 5 minutes.
  * Default: 2m

\text{search\_history\_max\_runtimes = <unsigned int>}
* The number of runtimes kept for each search.
* Used to calculate historical typical runtime during search
prioritization.
* Default: 10

**[search_metrics]**

debug_metrics = <bool>
* This indicates whether we should output more detailed search metrics for debugging.
* This will do things like break out where the time was spent by peer, and may add additional deeper levels of metrics.
* This is NOT related to "metrics.log" but to the "Execution Costs" and "Performance" fields in the Search inspector, or the count_map in the info.csv file.
* Default: false

**[show_source]**

distributed = <bool>
* Controls whether we will do a distributed search for show source to get events from all servers and indexes.
* Turning this off results in better performance for show source, but events will only come from the initial server and index.
* NOTE: event signing and verification is not supported in distributed mode.
* Default: true

distributed_search_limit = <unsigned int>
* Sets a limit on the maximum events we will request when doing the search for distributed show source.
* As this is used for a larger search than the initial non-distributed show source, it is larger than max_count.
* Splunk will rarely return anywhere near this amount of results, as we will prune the excess results.
* The point is to ensure the distributed search captures the target event in an environment with many events.
* Default: 30000

max_count = <integer>
* Maximum number of events accessible by show_source.
* The show source command will fail when more than this many events are
in the same second as the requested event.
* Default: 10000

`max_timeafter = <timespan>`
* Maximum time after requested event to show.
* Default: '1day' (86400 seconds)

`max_timebefore = <timespan>`
* Maximum time before requested event to show.
* Default: '1day' (86400 seconds)

### [rex]

`match_limit = <integer>`
* Limits the amount of resources that are spent by PCRE when running patterns that will not match.
* Use this to set an upper bound on how many times PCRE calls an internal function, match(). If set too low, PCRE might fail to correctly match a pattern.
* Default: 100000

`depth_limit = <integer>`
* Limits the amount of resources that are spent by PCRE when running patterns that will not match.
* Use this to limit the depth of nested backtracking in an internal PCRE function, match(). If set too low, PCRE might fail to correctly match a pattern.
* Default: 1000

### [slc]

`maxclusters = <integer>`
* Maximum number of clusters to create.
* Default: 10000.

### [slow_peer_disconnect]

# This stanza contains settings for the heuristic that will detect and disconnect slow peers towards the end of a search that has returned a large volume of data.

`batch_search_activation_fraction = <double>`
* The fraction of peers that must have completed before we start disconnecting.
* This is only applicable to batch search because the slow peers will
not hold back the fast peers.
* Default: 0.9

bound_on_disconnect_threshold_as_fraction_of_mean = <double>
* The maximum value of the threshold data rate we will use to determine
  if a peer is slow. The actual threshold will be computed dynamically
  at search time but will never exceed
  (100*maximum_threshold_as_fraction_of_mean)% on either side of the
  mean.
* Default: 0.2

disabled = <boolean>
* Specifies if this feature is enabled.
* Default: true

grace_period_before_disconnect = <double>
* If the heuristic consistently claims that the peer is slow for at
  least
  <grace_period_before_disconnect>*life_time_of_collector seconds then
  only
  will we disconnect the peer
* Default: 0.1

packets_per_data_point = <unsigned int>
* Rate statistics will be sampled once every packets_per_data_point
  packets.
* Default: 500

sensitivity = <double>
* Sensitivity of the heuristic to newer values. For larger values of
  sensitivity the heuristic will give more weight to newer statistic.
* Default: 0.3

threshold_connection_life_time = <unsigned int>
* All peers will be given an initial grace period of at least these many
  seconds before we consider them in the heuristic.
* Default: 60

threshold_data_volume = <unsigned int>
* The volume of uncompressed data that must have accumulated, in
  kilobytes (KB), from a peer before we consider them in the heuristic.
* Default: 1024

[summarize]

bucket_refresh_interval = <int>
* When poll_buckets_until_maxtime is enabled in a non-clustered
  environment, this is the minimum amount of time (in seconds)
  between bucket refreshes.
* Default: 30
bucket_refresh_interval_cluster = <int>
* When poll_buckets_until_maxtime is enabled in a clustered
  environment, this is the minimum amount of time (in seconds)
  between bucket refreshes.
* Default: 120

hot_bucket_min_new_events = <integer>
* The minimum number of new events that need to be added to the hot
  bucket
  (since last summarization) before a new summarization can take place.
  To disable hot bucket summarization set this value to a * large
  positive number.
* Default: 100000

indextime_lag = <unsigned int>
* The amount of lag time, in seconds, to give indexing to ensure that
  it has synced any received events to disk. Effectively, the data that
  has been received in the past ?indextime_lag? will NOT be summarized.
* Do not change this value unless directed by Splunk support.
* Default: 90

max_hot_bucket_summarization_idle_time = <unsigned int>
* Maximum amount of time, in seconds, a hot bucket can be idle. When the
  time exceeds the maximum, all of the events are summarized even if
  there are not enough events (determined by the hot_bucket_min_new_events
  attribute).
* Default: 900 (15 minutes)

max_replicated_hot_bucket_idle_time = <unsigned int>
* Maximum amount of time, in seconds, a replicated hot bucket can be idle
  after which we won't apply indextime_lag.
* This applies to only idle replicated hot buckets. As soon as new
  events start flowing in we will revert to the default behavior of applying
  indextime_lag
* Default: 150

max_summary_ratio = <float>
* A number in the [0-1] range that indicates the maximum ratio of
  summary data / bucket size at which point the summarization of that
  bucket, for the particular search, will be disabled. Use 0 to disable.
* Default: 0

max_summary_size = <int>
* Size of summary, in bytes, at which point we'll start applying the
  max_summary_ratio. Use 0 to disable.
* Default: 0
max_time = <int>
* The maximum amount of time, seconds, that a summary search process is allowed to run.
* Use 0 to disable.
* Default: 0

poll_buckets_until_maxtime = <bool>
* Only modify this setting when you are directed to do so by Support.
* Use the datamodels.conf setting acceleration.poll_buckets_until_maxtime for individual data models that are sensitive to summarization latency delays.
* Default: false

sleep_seconds = <integer>
* The amount of time, in seconds, to sleep between polling of summarization complete status.
* Default: 5

stale_lock_seconds = <integer>
* The amount of time, in seconds, to have elapse since the mod time of a .lock file before summarization considers that lock file stale and removes it.
* Default: 600

[system_checks]

insufficient_search_capabilities = enabled | disabled
* Enables/disables automatic daily logging of scheduled searches by users who have insufficient capabilities to run them as configured.
* Such searches are those that:
  + Have schedule_priority set to a value other than "default" but the owner does not have the edit_search_schedule_priority capability.
  + Have schedule_window set to a value other than "auto" but the owner does not have the edit_search_schedule_window capability.
* This check and any resulting logging occur on system startup and every 24 hours thereafter.
* Default: enabled

installed_files_integrity = enabled | log_only | disabled
* Enables/disables automatic verification on every startup that all the files that were installed with the running Splunk version are still the files that should be present.
* Effectively this finds cases where files were removed or changed that
should not be removed or changed, whether by accident or intent.
* The source of truth for the files that should be present is the manifest
  file in the $SPLUNK_HOME directory that comes with the release, so
  if
  this file is removed or altered, the check cannot work correctly.
  * Reading of all the files provided with the install has some I/O cost,
    though it is paid out over many seconds and should not be severe.
  * When "enabled", detected problems will cause a message to be posted to
    the bulletin board (system UI status message).
  * When "enabled" or "log_only", detected problems will cause details to be
    written out to splunkd.log
* When "disabled", no check will be attempted or reported.
* Default: enabled

orphan_searches = enabled|disabled
* Enables/disables automatic UI message notifications to admins for
  scheduled saved searches with invalid owners.
  * Scheduled saved searches with invalid owners are considered
    "orphaned".
    They cannot be run because Splunk cannot determine the roles to use
    for
    the search context.
  * Typically, this situation occurs when a user creates scheduled searches
    then departs the organization or company, causing their account to be
    deactivated.
* Currently this check and any resulting notifications occur on system
  startup and every 24 hours thereafter.
* Default: enabled

[thruput]

maxKBps = <integer>
* The maximum speed, in kilobytes per second, that incoming data is
  processed through the thruput processor in the ingestion pipeline.
* To control the CPU load while indexing, use this setting to throttle
  the number of events this indexer processes to the rate (in
  kilobytes per second) that you specify.
* NOTE:
  * There is no guarantee that the thruput processor
    will always process less than the number of kilobytes per
    second that you specify with this setting. The status of
    earlier processing queues in the pipeline can cause
    temporary bursts of network activity that exceed what
    is configured in the setting.
  * The setting does not limit the amount of data that is
written to the network from the tcpoutput processor, such as what happens when a universal forwarder sends data to an indexer.

* The throughput processor applies the 'maxKBps' setting for each ingestion pipeline. If you configure multiple ingestion pipelines, the processor multiplies the 'maxKBps' value by the number of ingestion pipelines that you have configured.

* For more information about multiple ingestion pipelines, see the 'parallelIngestionPipelines' setting in the server.conf.spec file.
* Default (Splunk Enterprise): 0 (unlimited)
* Default (Splunk Universal Forwarder): 256

[viewstates]

enable_reaper = <boolean>
* Controls whether the viewstate reaper runs
* Default: true

reaper_freq = <integer>
* Controls how often, in seconds, the viewstate reaper runs.
* Default: 86400 (24 hours)

reaper_soft_warn_level = <integer>
* Controls what the reaper considers an acceptable number of viewstates.
* Default: 1000

ttl = <integer>
* Controls the age, in seconds, at which a viewstate is considered eligible for reaping
* Default: 86400 (24 hours)

[scheduled_views]

# Scheduled views are hidden [saved searches / reports] that trigger PDF generation
# for a dashboard. When a user enables scheduled PDF delivery in the dashboard UI,
# scheduled views are created.
#
# The naming pattern for scheduled views is
#  _ScheduledView__<view_name>,
# where <view_name> is the name of the corresponding dashboard.
#
# The scheduled views reaper, if enabled, runs periodically to look for
# scheduled views that have been orphaned. A scheduled view becomes
orphaned
# when its corresponding dashboard has been deleted. The scheduled views
reaper
# deletes these orphaned scheduled views. The reaper only deletes
scheduled
# views if the scheduled views have not been disabled and their
permissions
# have not been modified.

enable_reaper = <boolean>
* Controls whether the scheduled views reaper runs, as well as whether
* scheduled views are deleted when the dashboard they reference is
deleted.
* Default: true

reaper_freq = <integer>
* Controls how often, in seconds, the scheduled views reaper runs.
* Default: 86400 (24 hours)

**OPTIMIZATION**

# This section contains global and specific optimization settings

[search_optimization]

enabled = <bool>
* Enables search optimizations
* Default: true

[search_optimization::search_expansion]

enabled = <bool>
* Enables optimizer-based search expansion.
* This enables the optimizer to work on pre-expanded searches.
* Default: true

# NOTE: Do not edit the below configurations unless directed by support

[search_optimization::replace_append_with_union]

enabled = <bool>
* Enables replace append with union command optimization
* Default: true

[search_optimization::merge_union]

enabled = <bool>
* Merge consecutive unions
* Default: true

[search_optimization::predicate_merge]

enabled = <bool>
* Enables predicate merge optimization
* Default: true

inputlookup_merge = <bool>
* Enables predicate merge optimization to merge predicates into inputlookup
* predicate_merge must be enabled for this optimization to be performed
* Default: true

merge_to_base_search = <bool>
* Enable the predicate merge optimization to merge the predicates into the first search in the pipeline.
* Default: true

fields_black_list = <fields_list>
* A comma-separated list of fields that will not be merged into the first search in the pipeline.
* If a field contains sub-tokens as values, then the field should be added to fields_black_list
* Default: no default

[search_optimization::predicate_push]

enabled = <bool>
* Enables predicate push optimization
* Default: true

[search_optimization::predicate_split]

enabled = <bool>
* Enables predicate split optimization
* Default: true
[search_optimization::projection_elimination]

enabled = <bool>
* Enables projection elimination optimization
* Default: true

cmds_black_list = <Commands List>
* A comma-separated list of commands that are not affected by projection elimination optimization.
* Default: no default

[search_optimization::required_field_values]

enabled = <bool>
* Enables required field value optimization
* Default: true

fields = <comma-separated-string>
* Provide a comma-separated-list of field names to optimize.
* Currently the only valid field names are eventtype and tag.
* Optimization of event type and tag field values applies to transforming searches.
  This optimization ensures that only the event types and tags necessary to process a search are loaded by the search processor.
* Only change this setting if you need to troubleshoot an issue.
* Default: eventtype, tag

[search_optimization::search_flip_normalization]

enabled = <bool>
* Enables predicate flip normalization.
* This type of normalization takes 'where' command statements in which the value is placed before the field name and reverses them so that the field name comes first.
* Predicate flip normalization only works for numeric values and string values where the value is surrounded by quotes.
* Predicate flip normalization also prepares searches to take advantage of predicate merge optimization.
* Disable search_flip_normalization if you determine that it is causing slow search performance.
* Default: true

[search_optimization::reverse_calculated_fields]

enabled = <bool>
* Enables reversing of calculated fields optimization.
  * Default: true

**[search_optimization::search_sort_normalization]**

`enabled = <bool>`
* Enables predicate sort normalization.
  * This type of normalization applies lexicographical sorting logic to 'search' command expressions and 'where' command statements, so they are consistently ordered in the same way.
  * Disable search_sort_normalization if you determine that it is causing slow search performance.
  * Default: true

**[search_optimization::eval_merge]**

`enabled = <bool>`
* Enables a search language optimization that combines two consecutive "eval" statements into one and can potentially improve search performance.
  * There should be no side-effects to enabling this setting and need not be changed unless you are troubleshooting an issue with search results.
  * Default: true

**[search_optimization::replace_table_with_fields]**

`enabled = <bool>`
* Enables a search language optimization that replaces the table command with the fields command in reporting or stream reporting searches.
  * There should be no side-effects to enabling this setting and need not be changed unless you are troubleshooting an issue with search results.
  * Default: true

**[directives]**

`required_tags = enabled|disabled`
* Enables the use of the required tags directive, which allows the search processor to load only the required tags from the conf system.
  * Disable this setting only to troubleshoot issues with search results.
  * Default: true
required_eventtypes = enabled|disabled
* Enables the use of the required eventtypes directive, which allows the
  search
  processor to load only the required event types from the conf system.
* Disable this setting only to troubleshoot issues with search results.
* Default: true

read_summary = enabled|disabled
* Enables the use of the read summary directive, which allows the search
  processor to leverage existing data model acceleration summary data
  when it
  performs event searches.
* Disable this setting only to troubleshoot issues with search results.
* Default: true

[parallelreduce]

maxReducersPerPhase = <positive integer>
* The maximum number of valid indexers that can be used as intermediate
  reducers in the reducing phase of a parallel reduce operation. Only
  healthy
  search peers are valid indexers.
* If you specify a number greater than 200 or an invalid value, parallel
  reduction does not take place. All reduction processing moves to the
  search
  head.
* Default: 4

maxRunningPrdSearches = <unsigned int>
* DEPRECATED. Use the 'maxPrdSearchesPerCpu' setting instead.

maxPrdSearchesPerCpu = <unsigned int>
* The maximum number of parallel reduce searches that can run, per CPU
  core,
  on an indexer that has been configured as an intermediate reducer.
* If you specify 0, there is no limit. The indexer runs as many parallel
  reduce searches as the indexer hardware permits.
* Default: 1

reducers = <string>
* Use this setting to configure one or more valid indexers as dedicated
  intermediate reducers for parallel reduce search operations. Only
  healthy
  search peers are valid indexers.
* For <string>, specify the indexer host and port using the following
  format -
  host:port. Separate each host:port pair with a comma to specify a list
  of
  intermediate reducers.
* If the 'reducers' list includes one or more valid indexers, all of
those
indexers (and only these indexers) are used as intermediate reducers
when you
run a parallel reduce search. If the number of valid indexers in the
'reducers' list exceeds 'maxReducersPerPhase', the Splunk software
randomly
selects the set of indexers that are used as intermediate reducers.
* If all of the indexers in the 'reducers' list are invalid, the search runs
  without parallel reduction. All reduce operations for the search are
  processed on the search head.
* If 'reducers' is empty or not configured, all valid indexers are
  potential intermediate reducer candidates. The Splunk software randomly selects
valid
  indexers as intermediate reducers with limits determined by the
  'winningRate'
  and 'maxReducersPerPhase' settings.
* Default: ""

winningRate = <positive integer>
* The percentage of valid indexers that can be selected from the search
  peers
  as intermediate reducers for a parallel reduce search operation.
* This setting is only respected when the 'reducers' setting is empty or not
  configured.
* If 100 is specified, the search head attempts to use all of the
  indexers.
* If 1 is specified, the search head attempts to use 1% of the indexers.
* The minimum number of indexers used as intermediate reducers is 1.
* The maximum number of indexers used as intermediate reducers is the value of
  'maxReducersPerPhase'.
* Default: 50

limits.conf.example

#    Version 7.2.0
# CAUTION: Do not alter the settings in limits.conf unless you know what
# you are doing.
# Improperly configured limits may result in splunkd crashes and/or
# memory overuse.

[searchresults]
maxresultrows = 50000
# maximum number of times to try in the atomic write operation (1 = no
retries)
tocsv_maxretry = 5
# retry period is 1/2 second (500 milliseconds)
tocsv_retryperiod_ms = 500

[subsearch]
# maximum number of results to return from a subsearch
maxout = 100
# maximum number of seconds to run a subsearch before finalizing
maxtime = 10
# time to cache a given subsearch's results
ttl = 300

[anomalousvalue]
maxresultrows = 50000
# maximum number of distinct values for a field
maxvalues = 100000
# maximum size in bytes of any single value (truncated to this size if larger)
maxvaluesize = 1000

[associate]
maxfields = 10000
maxvalues = 10000
maxvaluesize = 1000

# for the contingency, ctable, and counttable commands
[ctable]
maxvalues = 1000

[correlate]
maxfields = 1000

# for bin/bucket/discretize
[discretize]
maxbins = 50000
# if maxbins not specified or = 0, defaults to searchresults::maxresultrows

[inputcsv]
# maximum number of retries for creating a tmp directory (with random name in
# SPLUNK_HOME/var/run/splunk)
mkdir_max_retries = 100

[kmeans]
maxdatapoints = 100000000

[kv]
# when non-zero, the point at which kv should stop creating new columns
maxcols = 512
maxresultrows = 50000
# maximum distinct value vectors to keep track of
maxvalues = 100000
maxvaluesize = 1000

# maximum result rows to be returned by /events or /results getters from
# REST
# API
maxresultrows = 50000

# how long searches should be stored on disk once completed
ttl = 86400

# the approximate maximum number of timeline buckets to maintain
status_buckets = 300

# the last accessible event in a call that takes a base and bounds
max_count = 10000

# the minimum length of a prefix before a * to ask the index about
min_prefix_len = 1

# the length of time to persist search cache entries (in seconds)
cache_ttl = 300

# By default, we will not retry searches in the event of indexer
# failures with indexer clustering enabled.
# Hence, the default value for search_retry here is false.
search_retry = false

# Timeout value for checking search marker files like hotbucketmarker or
# backfill
# marker.
check_search_marker_done_interval = 60

# Time interval of sleeping between subsequent search marker files
# checks.
check_search_marker_sleep_interval = 1

# If number of cpu's in your machine is 14 then total system wide number
# of
# concurrent searches this machine can handle is 20.
# which is base_max_searches + max_searches_per_cpu x num_cpus = 6 + 14
# x 1 = 20
base_max_searches = 6
max_searches_per_cpu = 1

[restapi]
# Percent of total concurrent searches that will be used by scheduler is
total concurrency x max_searches_perc = 20 x 60% = 12 scheduled searches
# User default value (needed only if different from system/default value) when
# no max_searches_perc.<n>.when (if any) below matches.
max_searches_perc = 60

# Increase the value between midnight-5AM.
max_searches_perc.0 = 75
max_searches_perc.0.when = * 0-5 * * *

# More specifically, increase it even more on weekends.
max_searches_perc.1 = 85
max_searches_perc.1.when = * 0-5 * * 0,6

# Maximum number of concurrent searches is enforced cluster-wide by the
# captain for scheduled searches. For a 3 node SHC total concurrent
# searches = 3 x 20 = 60. The total searches (adhoc + scheduled) = 60,
# then
# no more scheduled searches can start until some slots are free.
shc_syswide_quota_enforcement = true

[slc]
# maximum number of clusters to create
maxclusters = 10000

[findkeywords]
#events to use in findkeywords command (and patterns UI)
maxevents = 50000

[stats]
maxresultrows = 50000
maxvalues = 10000
maxvaluesize = 1000

[top]
maxresultrows = 50000
# maximum distinct value vectors to keep track of
maxvalues = 100000
maxvaluesize = 1000

[search_optimization]
enabled = true
[search_optimization::predicate_split]
enabled = true
[search_optimization::predicate_push]
enabled = true
[search_optimization::predicate_merge]
enabled = true
inputlookup_merge = true
merge_to_base_search = true

[search_optimization::projection_elimination]
enabled = true
cmds_black_list = eval, rename

[search_optimization::search_flip_normalization]
enabled = true

[search_optimization::reverse_calculated_fields]
enabled = true

[search_optimization::search_sort_normalization]
enabled = true

[search_optimization::replace_table_with_fields]
enabled = true

logging.conf

The following are the spec and example files for logging.conf.

logging.conf.spec

# The format and semantics of this file are described in this article at Python.org:
#
# [Configuration file format](https://docs.python.org/2/library/logging.config.html#configuration-file-format)
#
# This file must contain sections called [loggers], [handlers] and [formatters] which identify by name the entities of each type which are defined in the file. For each such entity, there is a separate section which identifies how that entity is configured.

level = <DEBUG|INFO|WARNING|ERROR|CRITICAL|NOTSET>
* For the root logger only, NOTSET means that all messages will be logged.

handlers = <comma-separated list>
* A comma-separated list of handler names, which must appear in the [handlers] section.
* These names must appear in the [handlers] section and have corresponding sections in the configuration file.

```
qualname = <string>
* The hierarchical channel name of the logger (the name used by the application to get the logger).

propagate = <0|1>
* Set to "1" to indicate that messages must propagate to handlers higher up the logger hierarchy from this logger.
* Set to "0" to indicate that messages are not propagated to handlers up the hierarchy.
```

class = <string>
* Indicates the handler?s class, as determined by eval() in the logging package?s namespace.

```
args = <comma-separated list>
* The list of arguments to the constructor for the handler class, when eval()uated in the context of the logging package?s namespace.
```

```
formatter = <string>
* The key name of the formatter for this handler.
* If blank, a default formatter (logging._defaultFormatter) is used.
* If a name is specified, it must appear in the [formatters] section and have a corresponding section in the configuration file.
```

**logging.conf.example**

No example

**macros.conf**

The following are the spec and example files for macros.conf.

**macros.conf.spec**

```
#   Version 7.2.0
#
```

444
# This file contains possible attribute/value pairs for search language macros.

# To learn more about configuration files (including precedence) please see the
# documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

[<STANZA_NAME>]

* Each stanza represents a search macro that can be referenced in any search.
* The stanza name is the name of the macro if the macro takes no arguments.
  Otherwise, the stanza name is the macro name appended with
  "(<numargs>)", where <numargs> is the number of arguments that this macro takes.
* Macros can be overloaded. In other words, they can have the same name but a
different number of arguments. If you have [foobar], [foobar(1)],
[foobar(2)], etc., they are not the same macro.
* Macros can be used in the search language by enclosing the macro name and any
  argument list within tick marks, for example:`foobar(arg1,arg2)` or`footer`.
* Splunk does not expand macros when they are inside of quoted values, for
  example: "foo`bar`baz".

args = <string>,<string>,...
* A comma-delimited string of argument names.
* Argument names can only contain alphanumeric characters, underscores`
  '_' , and hyphens '-'.
* If the stanza name indicates that this macro takes no arguments, this
  attribute will be ignored.
* This list cannot contain any repeated elements.

definition = <string>
* The string that the macro will expand to, with the argument substitutions
  made. (The exception is when iseval = true, see below.)
* Arguments to be substituted must be wrapped by dollar signs ($), for example:
  "the last part of this string will be replaced by the value of
  argument foo $foo$".
* Splunk replaces the $<$arg>$ pattern globally in the string, even inside of
  quotes.
validation = <string>
* A validation string that is an 'eval' expression. This expression must
  evaluate to a boolean or a string.
* Use this to verify that the macro's argument values are acceptable.
* If the validation expression is boolean, validation succeeds when it
  returns true. If it returns false or is NULL, validation fails, and Splunk
  returns the error message defined by the attribute, errormsg.
* If the validation expression is not boolean, Splunk expects it to
  return a string or NULL. If it returns NULL, validation is considered a
  success.
  Otherwise, the string returned is the error string.

errmsg = <string>
* The error message to be displayed if validation is a boolean
  expression and
  it does not evaluate to true.

iseval = <true/false>
* If true, the definition attribute is expected to be an eval expression
  that
  returns a string that represents the expansion of this macro.
  * Defaults to false.

description = <string>
* OPTIONAL. Simple english description of what the macro does.

macros.conf.example

#   Version 7.2.0
#
# Example macros.conf
#

# macro foobar that takes no arguments can be invoked via `foobar`
[foobar]
# the definition of a macro can invoke another macro. nesting can be
# indefinite
# and cycles will be detected and result in an error
definition = `foobar(foo=defaultfoo)`

# macro foobar that takes one argument, invoked via `foobar(someval)`
[foobar(1)]
args = foo
# note this is definition will include the leading and trailing quotes, i.e.
# something `foobar(someval)`
# would expand to
# something "foo = someval"
definition = "foo = $foo$"

# macro that takes two arguments
# note that macro arguments can be named so this particular macro could be
# invoked equivalently as `foobar(1,2)` `foobar(foo=1,bar=2)` or
# `foobar(bar=2,foo=1)`
[foobar(2)]
args = foo, bar
definition = "foo = $foo$, bar = $bar$"

# macro that takes one argument that does validation
[foovalid(1)]
args = foo
definition = "foovalid = $foo$"

# the validation eval function takes any even number of arguments (>=2)
where
# the first argument is a boolean expression, the 2nd a string, the
# third
# boolean, 4th a string, etc etc etc
validation = validate(foo>15,"foo must be greater than 15",foo<=100,"foo must be <= 100")

# macro showing simple boolean validation, where if foo > bar is not true,
# errormsg is displayed
[foovalid(2)]
args = foo, bar
definition = "foo = $foo$ and bar = $bar$"
validation = foo > bar
errormsg = foo must be greater than bar

# example of an eval-based definition. For example in this case
# `fooeval(10,20)` would get replaced by 10 + 20
[fooeval(2)]
args = foo, bar
definition = if (bar > 0, "$foo$ + $bar$", "$foo$ - $bar$")
iseval = true

mad.conf

The following are the spec and example files for mad.conf.
# This file contains possible settings you can use to configure metric anomaly detection.
# Use anomaly detection to identify trends and outliers in KPI search results that might
# indicate an issue with your system.
# There is a mad.conf in
#$SPLUNK_HOME/etc/apps/SA-ITSI-MetricAD/default. To set custom
# configurations, place a mad.conf in
#$SPLUNK_HOME/etc/apps/SA-ITSI-MetricAD/local.
#
# To learn more about configuration files (including precedence), see
# the
documentation located at
#
http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
#
# To learn more about metric anomaly detection, see
#
http://docs.splunk.com/Documentation/ITSI/latest/Configure/Enableanomalydetection
#
# In most situations, the default values specified in mad.conf should work as-is.
# Modifying this file can result in negative changes to anomaly detection accuracy.
# Do NOT remove any stanzas or settings in the configuration file.
#
# For <duration> format, this configuration file accepts the following units:
#   * ms => milliseconds
#   * s, sec, secs, second, seconds => second
#   * m, min, mins, minute, minutes => minute
#   * h, hr, hrs, hour, hours => hour
#   * d, day, days => day

[service]

unbounded_buffer_size = <duration>
* The size of the data buffer used in batch mode.
* For example, "4d" stores a maximum of 4 days of data.
* Default: 400d

kvstore_connect_interval = <duration>
* How often to retry connecting to the KV store when the connection is lost.
* Default: 30s
rest_ssl_permissive_trustmanager = <boolean>
* Whether to enable PermissiveX509TrustManager with HTTPS connection to Splunk REST API.
* Do not modify this setting unless Splunk is not running in HTTPS mode.
* Default: true

rest_ssl_permissive_hostnameverifier = <boolean>
* Whether hostname verification is strict or permissive.
* If set to "true", hostname verification is permissive.
* If set to "false", hostname verification is strict.
* This setting can be disabled when the Splunk certificate is not self-signed.
* Default: true

trending_bounded_buffer_size = <duration>
* The size of the data buffer for the trending algorithm in real-time mode.
* This setting MUST be larger than the value of the 'training_period' setting in the [trending] stanza.
* Default: 15d

cohesive_bounded_rt_buffer_size = <duration>
* The size of the real-time data buffer for the cohesive algorithm in real-time mode.
* Default: 12h

cohesive_bounded_backfill_buffer_size = <duration>
* The size of the backfill data buffer for the cohesive algorithm in real-time mode.
* Default: 25h

[trending]

* Use this stanza to configure the 'mad' command for the trending algorithm.

periods.days = <positive integer>
* How many days to look back for normal patterns in the data.
* Must be a value greater than zero.
* Default: 6

periods.weeks = <integer>
* How many weeks to look back for normal patterns in the data.
* Must be a value greater than or equal to zero.
* Default: 2

window_size = <positive integer>
* How many data points to use to construct an analysis window.
* Must be a value greater than 1.
step_size = <positive integer>
* The offset size of two consecutive analysis window.
* Must be a value greater than 0.
* Default: 1

training_period = <duration>
* The amount of time used to train the algorithm.
* Must be a value greater than 1.
* Default: 7d

max_NA_ratio = <float>
* The maximum possible ratio of NaN (undefined) data points.
* Must be a decimal between 0.0 and 1.0.
* Default: 0.5

na_rm = <boolean>
* Whether or not to remove NaN (undefined) data points.
* If set to "true", NaN data points are removed.
* Default: true

Nkeep = <duration>
* How much data to keep in memory for analysis.
* Default: 50h

Naccum = <float>
* The accumulation score for anomaly alerting.
* Must be a value greater than zero.
* Default: 35.0

[trending:limits]

* Use this stanza to configure the 'naccum' command for trending algorithm.

Naccum_max = <float>
* The maximum accumulation score to use for detecting anomalies.
* This value MUST be larger than the 'Naccum' setting in the [trending] stanza.
* Default: 50.0

Naccum_min = <float>
* The minimum accumulation score to use for detecting anomalies.
* This value MUST be smaller than the 'Naccum' in the [trending] stanza.
* Default: 30.0

sensitivity_max = <integer>
* The number of sensitivity levels.
* Must be a value greater than 1.
* Default: 10

**[cohesive]**

* Use this stanza to configure the 'mad' command for the cohesive algorithm.

`window_size = <positive integer>`
* How many data points to use to construct an analysis window.
* Must be a value greater than 1.
* Default: 60

`step_size = <positive integer>`
* The offset size of two consecutive analysis windows.
* Must be a value greater than 0.
* Default: 1

`training_period = <duration>`
* The amount of time used to train the algorithm.
* Must be a value greater than 1.
* Default: 7d

`max_NA_ratio = <float>`
* The maximum possible ratio of NaN (undefined) data points.
* Must be a decimal between 0.0 and 1.0.
* Default: 0.5

`na_rm = <boolean>`
* Whether or not to remove NaN (undefined) data points.
* If set to "true", NaN data points are removed.
* Default: true

`Nkeep = <duration>`
* How much data to keep in memory for analysis.
* Default: 10h

`Naccum = <float>`
* The accumulation score for anomaly alerting.
* Must be a number greater than zero.
* Default: 35.0

`norm_Ntrend = <integer>`
* The window of moving median for normalization of incoming data.
* Default: 10

`norm_maxNAratio = <float>`
* The maximum ratio of NaN data points allowed in the dataset for normalization of incoming data.
* Must be a decimal between 0.0 and 1.0.
* Default: 0.5
norm_trendOnly = <boolean>
  * Whether to use only the trend of the data for normalization.
  * Default: false

norm_MAratio = 0.8
  * The moving average ratio of the normalization window.
  * Must be a decimal between 0.0 and 1.0.
  * Default: 0.8

norm_NArm = <boolean>
  * Whether to remove NaN (undefined) data points for normalization.
  * Default: false

norm_Nwindow = <integer>
  * The size, in data points, of the normalization buffer.
  * Default: 10080

norm_Nshift = <integer>
  * The interval at which the normalization constants are recalculated.
  * After receiving this many data points, the constants are recalculated.
  * Default: 1440

norm_Ninit = <integer>
  * The number of data points needed to calculate the normalization
    constants.
  * Default: 30

metrics_maximum = <integer>
  * The maximum number of metrics that can be analyzed for the cohesive
    algorithm.
  * Default: 30

[cohesive:limits]

  * Use this stanza to configure the 'naccum' command for the cohesive
    algorithm.

Naccum_max = <float>
  * The maximum accumulation score that can be used for detecting
    anomalies.
  * This value MUST be larger than the 'Naccum' setting in the [cohesive]
    stanza.
  * Default: 50.0

Naccum_min = <float>
  * The minimum accumulation score that can be used for detecting
    anomalies.
  * This value MUST be smaller than the 'Naccum' setting in the [cohesive]
    stanza.
sensitivity_max = <integer>
* The number of sensitivity levels.
* Must be a value greater than 1.
* Default: 10

[logging]

* Use this stanza to configure logging.

metric_registry = <boolean>
* Enable logging metrics of the 'mad' command.
* CAUTION: Enabling this setting will have a significant performance impact.
* Default: false

>alerting

* Use this stanza to configure external HTTP endpoint connections for posting alerts.

rest_ssl_permissive_trustmanager = <boolean>
* Whether to enable PermissiveX509TrustManager with HTTPS connection to the Splunk REST API.
* Default: true

rest_ssl_permissive_hostnameverifier = <boolean>
* Whether to be strict or permissive in hostname verification.
* If set to "true", hostname verification is permissive.
* If set to "false", hostname verification is strict.
* Default: true

max_http_connection = 100
* How many simultaneous HTTP connections are allowed.
* Default: 100

mad.conf.example

No example
notable_event_actions.conf

The following are the spec and example files for `notable_event_actions.conf`.

`notable_event_actions.conf.spec`

```plaintext
# This file contains attributes and values for taking actions on episodes
# in Episode Review.
#
# There is a notable_event_actions.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/default/.
# To set custom configurations, place a notable_event_actions.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/local/. You must restart Splunk to enable
# configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the top
#   of the file.
# * Each conf file should have at most one default stanza. If there are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in
#   the file wins.
# * If an attribute is defined at both the global level and in a
#   specific stanza, the value in the specific stanza takes precedence.

disabled = <boolean>
* Disable a notable event action by setting to 1.
* Optional.
* Default: 0

is_group_compatible = <boolean>
* Make an action available for episodes by setting to 1.
```
* Default: 1

**is_bulk_compatible = <boolean>**
* Make an action available for bulk events by setting to 1.
* Default: 1

**[<action_name>]**

* Each stanza represents an episode action. The action name is the type of action you want to configure.
* Options are email, script, itsi_sample_event_action_ping, itsi_event_action_link_ticket, snow_incident, and remedy_incident.

**execute_in_sync = <boolean>**
* If 1, ITSI executes the action synchronously.
* The UI notifies you when the action is truly complete, rather than requiring you to check back later to confirm.
* It is recommended that you set this value to 1 for an external ticket created by a Splunk custom search command or modular alert.
* Default: 0

**execute_once_per_group = <boolean>**
* If 1, ITSI executes the action exactly once in the case of a bulk action.
* In special cases (like if this alert action has 'type' set to "external_ticket"), the result of a refresh is associated with all the events in the group.
* Default: 0

**type = <string>**
* The type of action to take on the episode.
* Use this setting if you are creating a ServiceNow or Remedy ticket from an episode.
* The only supported value for this setting is "external_ticket", which creates a ticket in the external ticketing system you choose.
* If you set the value to "external_ticket", ITSI runs a refresh action right after execution.
* The attribute-value pairs below are applicable when 'type' is "external_ticket".

**ticket_system_name = <string>**
* The name of the external ticketing system in which to create the ticket.

**relative_refresh_uri = <string>**
* A relative URI for the search head where ITSI is installed.
* https://localhost:8089/ or something similar is prepended to the URI.
* ITSI constructs this link so you can navigate directly to the
external ticket.
* ITSI issues a GET call on this URI and outputs JSON data.
* 'refresh_response_json_path' indicates the path to walk through the received JSON output.
* Do not change this from the default value or refresh will not work.

correlation_key = <string>
* Optional. The query parameter to be appended to 'relative_refresh_uri'.
* The parameter is also saved in the KV store collection that contains all created tickets.
* Do not change this from the default value or refresh will not work.
* Default: correlation_id

correlation_value = <string>
* The key in the raw notable event whose value to append to the refresh URI.
* If a 'correlation_key' exists, ITSI appends this value to the refresh URI instead.
* Do not change this from the default value or refresh will not work.
* Default: $result.event_id$

correlation_value_for_group = <string>
* The key in the episode whose value to append to the refresh URI.
* By default, ITSI uses the value corresponding to 'itsi_group_id'.
* Do not change this from the default value or refresh will not work.
* Default: $result.itsi_group_id$

refresh_response_json_path = <string>
* Because the JSON output of 'relative_refresh_uri' can be nested and complex, this setting indicates the path to walk through the received output.
* Do not change this from the default value or refresh will not work.
* Default: entry.{0}.content

refresh_response_ticket_id_key = <string>
* After traversing the JSON path specified in 'refresh_response_json_path' and fetching a JSON blob, the key corresponding to the external ticket ID.
* Do not change this from the default value or refresh will not work.

refresh_response_ticket_url_key = <string>
* After traversing the JSON path specified in 'refresh_response_json_path' and fetching a JSON blob, the key corresponding to the external ticket URL.
* Do not change this from the default value or refresh will not work.
# This is an example notable_event_actions.conf. Use this file to configure
# episode actions.
#
# To use one or more of these configurations, copy the configuration block
# into notable_event_actions.conf in
#$SPLUNK_HOME/etc/apps/SA-ITOA/local.
# You must restart Splunk to enable configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles
#
# This example alert includes running a script, sending an email, and
# creating an incident in ServiceNow.

[email]
disabled = 0

[script]
disabled = 0

[snow_incident]
disabled = 0
type = external_ticket
execute_in_sync = 1
execute_once_per_group = 1
ticket_system_name = ServiceNow
relative_refresh_uri = /servicesNS/nobody/-/service_now_incident/snow_incident
correlation_key = correlation_id
* Refresh URI now becomes
  /servicesNS/nobody/-/service_now_incident/snow_incident?correlation_id=$result.event_id$

  * Assuming we are dealing with an event whose event_id is 'myevent1234',
    the URI now becomes:
    /servicesNS/nobody/-/service_now_incident/snow_incident?correlation_id=myevent1234
  * Final URI with output_mode:
    /servicesNS/nobody/-/service_now_incident/snow_incident?correlation_id=myevent1234&output_mode=json

  * If there is no 'correlation_key' specified, the final URI looks like:
    /servicesNS/nobody/-/service_now_incident/snow_incident/myevent1234?output_mode=json
correlation_value_for_group = $result.itsi_group_id$
* When operating on an episode, we will use the value corresponding to 'itsi_group_id'
  as the correlation_id. Similar to correlation_value mentioned above.

refresh_response_json_path = entry(0).content
* Assuming the JSON response looks like the following:

  
  
  
  {  

  "entry": [  

  {  

  "content": {  

  "number": "INC0047495",

  }  

  }  

  ]  

  }  

... the path value is indicative of how to extract the ticket_id and ticket_url.

refresh_response_ticket_id_key = number
* After extracting the JSON blob we are interested in, which looks like the following:

  
  
  {  

  "number": "INC0047495",
  "url": "https://abc.service-now.com/incident.do?sysparm_query=correlation_id=myevent1234"

  }  

... 'number' is the value we are interested in.

refresh_response_ticket_url_key = url
* After extracting the JSON blob we are interested in, which looks like the following:

  
  {  

  "number": "INC0047495",
  "url": "https://abc.service-now.com/incident.do?sysparm_query=correlation_id=myevent1234"

  }  

... 'url' is the value we are interested in.
notable_event_commonality.conf

The following are the spec and example files for notable_event_commonality.conf.

notable_event_commonality.conf.spec

# This file contains possible attribute/value pairs for blacklisting
# notable event fields from the Common Fields section of episodes.
#
# There is a notable_event_commonality.conf in
$SPLUNK_HOME/etc/apps/SA-ITOA/default/.
# To set custom configurations, place a notable_event_commonality.conf
in
# $SPLUNK_HOME/etc/apps/SA-ITOA/local. You must restart Splunk software
to enable
# configurations.
#
# To learn more about configuration files (including precedence) please
see
# the documentation located at
#
http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

[common_event_fields]

black_list_fields = <comma-separated list>
* A list of field names in a notable event that will not appear in the
Common Fields section of an episode.
* By default, ITS1 blacklists fields that are not core to the raw event
itself, or ones that are mainly used internally.
* Add fields here that you don't necessarily care about, but that you
know
  will probably appear in most of your events.

notable_event_commonality.conf.example

No example
**notable_event_correlation.conf**

The following are the spec and example files for notable_event_correlation.conf.

**notable_event_correlation.conf.spec**

```plaintext
# This file contains attributes and values that ITSI Smart Mode uses to correlate
# notable events.
#
# There is a notable_event_correlation.conf in
#$SPLUNK_HOME/etc/apps/SA-ITOA/default/.
# To set custom configurations, place a notable_event_correlation.conf
# in
#$SPLUNK_HOME/etc/apps/SA-ITOA/local. You must restart Splunk software
to enable
# configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

**GLOBAL SETTINGS**

```plaintext
# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the top
#   of the file.
# * Each .conf file should have at most one default stanza. If there are
#   multiple default stanzas, attributes are combined. In the case of
#   multiple definitions of the same attribute, the last definition in
#   the file wins.
# * If an attribute is defined at both the global level and in a specific
#   stanza, the value in the specific stanza takes precedence.
```
[smart_mode_correlation_engine]

* The settings under this stanza determine how ITSI Smart Mode analyzes notable event fields to determine whether they contain textual or categorical content.
* Smart Mode uses machine learning to compare event field values and group events that are related to each other.
* CAUTION: This configuration file does not support adding any additional stanzas.
  Do not add, remove, or change any of the settings or stanzas in this file unless specifically instructed to do so by a Splunk support specialist.

BLACKLIST FIELDS

black_list_fields = <comma-separated list>
* A list of field names in a notable event whose values to discard from consideration for Smart Mode event correlation.

TEXTUAL FIELDS

text_field_names = <comma-separated list>
* A list of field names in a notable event that usually represent textual content.
* A text field is a data structure that holds alphanumeric data, such as name and address.
* Defaults: comment, description, summary, review, message

ignore_fields_that_contain = <comma-separated list>
* A list of field names to implicitly ignore because they are not useful for event correlation.
* ITSI ignores field names that contain any of the words in this list.
* For example, with the default "time", ITSI ignores fields that represent time, like alert_triggertime, alerttriggertime, lasttimeup, etc.
* Default: time

threshold_event_coverage_perc = <int>
* A threshold value for considering a notable event field as a text field.
* If the count (total number of occurrences) of a field divided by
the total number of events processed in the time frame is less than the percentage specified in 'threshold_event_coverage_perc', then the notable event field is a text field.
* Default: 10

**CATEGORICAL FIELDS**

threshold_distinct_value_perc = <int>
* A threshold value for considering a notable event field as a categorical field.
* If the distinct_count (count of distinct values) of a field divided by the count (total number of occurrences) of the field is less than the percentage specified, then the notable event field is a categorical field.
* Categorical fields have a distinct value, such as a status field, as opposed to textual data, descriptions, numerical values, and comments.
* If this setting determines that a field is NOT a categorical field, ITSI uses the two settings below ('min_distinct_value_perc' and 'max_count_perc') in a second calculation to check whether the field is a categorical field.
* Default: 35

min_distinct_value_perc = <int>
* Helps confirm whether a notable event field is a categorical field.
* Sets the minimum distinctive value percentage that a notable event field must be to be considered a categorical field.
* If the cumulative sum of min_distinct_value_perc of distinct_count (count of unique values) of a field is at least max_count_perc of the count (total number of occurrences) of the field, then the notable event field is considered a categorical field.
* For example, consider the following field:value pairs:
  {field:value1 count:34}, {field:value2 count:31}, {field:value3 count:5},
  {field:value4 count:5}, ?, {field:value18 count:1} {field:value19 count:1},
  {field:value20 count:1}
  There are 20 different values listed for this field, so distinct_count = 20.
  ITSI sums the counts of all the values, so count = 80.
  80% of count = 64
  10% of distinct_count = 2, so you add the counts of the first two values above (34 + 31).
(field: value1 count: 34) + (field: value2 count: 31) = 34 + 31 = 65 > 64
Because 65 is at least 64, ?field? is a categorical field.
* Default: 10

max_count_perc = <int>
* Helps confirm whether a notable event field is a categorical field.
* Sets the maximum count percentage that a notable event field must
  be to be considered a categorical field.
* If the cumulative sum of ?min_distinct_value_perc? of distinct_count
  (count
  of unique values) of a field is at least ?max_count_perc? of the count
  (total number of occurrences) of the field, then the notable event
  field is
  considered a categorical field.
* See the example for the 'min_distinct_value_perc' setting to
  understand
  how this setting works.
* Default: 80

notable_event_correlation.conf.example

No example

props.conf

The following are the spec and example files for props.conf.

props.conf.spec

#   Version 7.2.0
#
# This file contains possible setting/value pairs for configuring Splunk
# software's processing properties via props.conf.
#
# Props.conf is commonly used for:
#
# * Configuring line breaking for multi-line events.
# * Setting up character set encoding.
# * Allowing processing of binary files.
# * Configuring timestamp recognition.
# * Configuring event segmentation.
# * Overriding automated host and source type matching. You can use
#   props.conf to:
#   * Configure advanced (regex-based) host and source type overrides.
* Override source type matching for data from a particular source.
* Set up rule-based source type recognition.
* Rename source types.
* Anonymizing certain types of sensitive incoming data, such as credit card or social security numbers, using sed scripts.
* Routing specific events to a particular index, when you have multiple indexes.
* Creating new index-time field extractions, including header-based field extractions.
* NOTE: We do not recommend adding to the set of fields that are extracted at index time unless it is absolutely necessary because there are negative performance implications.
* Defining new search-time field extractions. You can define basic search-time field extractions entirely through props.conf, but a transforms.conf component is required if you need to create search-time field extractions that involve one or more of the following:
  * Reuse of the same field-extracting regular expression across multiple sources, source types, or hosts.
  * Application of more than one regex to the same source, source type, or host.
  * Delimiter-based field extractions (they involve field-value pairs that are separated by commas, colons, semicolons, bars, or something similar).
  * Extraction of multiple values for the same field (multivalued field extraction).
  * Extraction of fields with names that begin with numbers or underscores.
* Setting up lookup tables that look up fields from external sources.
* Creating field aliases.

NOTE: Several of the above actions involve a corresponding transforms.conf configuration.

You can find more information on these topics by searching the Splunk documentation (http://docs.splunk.com/Documentation/Splunk).

There is a props.conf in $SPLUNK_HOME/etc/system/default/. To set custom configurations, place a props.conf in $SPLUNK_HOME/etc/system/local/. For help, see props.conf.example.

You can enable configurations changes made to props.conf by typing the following search string in Splunk Web:
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
# For more information about using props.conf in conjunction with
# distributed Splunk deployments, see the Distributed Deployment Manual.

## GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the
top
#   of the file.
# * Each conf file should have at most one default stanza. If there are
# multiple default stanzas, settings are combined. In the case of
# multiple definitions of the same setting, the last definition in
# the
#   file wins.
# * If a setting is defined at both the global level and in a specific
# stanza, the value in the specific stanza takes precedence.

### [<spec>]
* This stanza enables properties for a given <spec>.
* A props.conf file can contain multiple stanzas for any number of
different <spec>.
* Follow this stanza name with any number of the following setting/value pairs, as appropriate for what you want to do.
* If you do not set an setting for a given <spec>, the default is used.

<spec> can be:
1. `<sourcetype>`, the source type of an event.
2. `host::<host>`, where `<host>` is the host, or host-matching pattern, for an
   event.
3. `source::<source>`, where `<source>` is the source, or source-matching
   pattern, for an event.
4. `rule::<rulename>`, where `<rulename>` is a unique name of a source type
   classification rule.
5. `delayedrule::<rulename>`, where `<rulename>` is a unique name of a delayed
   source type classification rule.

These are only considered as a last resort
before generating a new source type based on the source seen.

**[<spec>] stanza precedence:**

For settings that are specified in multiple categories of matching [<spec>] stanzas, [host::<host>] settings override [<source type>] settings. Additionally, [source::<source>] settings override both [host::<host>] and [<source type>] settings.

**Considerations for Windows file paths:**

When you specify Windows-based file paths as part of a [source::<source>] stanza, you must escape any backslashes contained within the specified file path.

Example: [source::c:\\path_to\\file.txt]

**[<spec>] stanza patterns:**

When setting a [<spec>] stanza, you can use the following regex-type syntax:
... recurses through directories until the match is met or equivalently, matches any number of characters.
* matches anything but the path separator 0 or more times.
The path separator is '/' on unix, or '\' on windows.
   Intended to match a partial or complete directory or filename.
| is equivalent to 'or'
( ) are used to limit scope of |.
\\ = matches a literal backslash '\'.

Example: [source:.....(?<!tar.)(gz|bz2)]

This matches any file ending with '.gz' or '.bz2', provided this is not preceded by 'tar.', so tar.bz2 and tar.gz would not be matched.

**[source::<source>] and [host::<host>] stanza match language:**

Match expressions must match the entire name, not just a substring. If you are familiar with regular expressions, match expressions are based on a full implementation of PCRE with the translation of ...,* and . Thus . matches a period, * matches non-directory separators, and ... matches any number of any characters.
For more information search the Splunk documentation for "specify input paths with wildcards".

**[<spec>] stanza pattern collisions:**

Suppose the source of a given input matches multiple [source::<source>] patterns. If the [<spec>] stanzas for these patterns each supply distinct settings, Splunk software applies all of these settings.

However, suppose two [<spec>] stanzas supply the same setting. In this case, Splunk software chooses the value to apply based on the ASCII order of the patterns in question.

For example, take this source:

source::az

and the following colliding patterns:

[source::...a...] sourcetype = a
[source::...z...] sourcetype = z

In this case, the settings provided by the pattern [source::...a...] take precedence over those provided by [source::...z...], and sourcetype ends up with "a" as its value.

To override this default ASCII ordering, use the priority key:

[source::...a...] sourcetype = a priority = 5
[source::...z...] sourcetype = z priority = 10

Assigning a higher priority to the second stanza causes sourcetype to have the value "z".

**Case-sensitivity for [<spec>] stanza matching:**

By default, [source::<source>] and [<sourcetype>] stanzas match in a case-sensitive manner, while [host::<host>] stanzas match in a case-insensitive manner. This is a convenient default, given that DNS
names are case-insensitive.

To force a [host::<host>] stanza to match in a case-sensitive manner use the "(?-i)" option in its pattern.

For example:

```plaintext
[host::foo]
FIELDALIAS-a = a AS one

[host::(?-i)bar]
FIELDALIAS-b = b AS two
```

The first stanza will actually apply to events with host values of "FOO" or "Foo". The second stanza, on the other hand, will not apply to events with host values of "BAR" or "Bar".

**Building the final [<spec>] stanza:**

The final [<spec>] stanza is built by layering together (1) literal-matching stanzas (stanzas which match the string literally) and (2) any regex-matching stanzas, according to the value of the priority field.

If not specified, the default value of the priority key is:
* 0 for pattern-matching stanzas.
* 100 for literal-matching stanzas.

NOTE: Setting the priority key to a value greater than 100 causes the pattern-matched [<spec>] stanzas to override the values of the literal-matching [<spec>] stanzas.

The priority key can also be used to resolve collisions between [<sourcetype>] patterns and [host::<host>] patterns. However, be aware that the priority key does *not* affect precedence across <spec> types. For example, [<spec>] stanzas with [source::<source>] patterns take priority over stanzas with [host::<host>] and [<sourcetype>] patterns, regardless of their respective priority key values.

#******************************************************************************
# The possible setting/value pairs for props.conf, and their default values, are:
#******************************************************************************
priority = <number>
* Overrides the default ASCII ordering of matching stanza names

# International characters and character encoding.

CHARSET = <string>
* When set, Splunk software assumes the input from the given [<spec>] is in
  the specified encoding.
* Can only be used as the basis of [sourcetype] or [source::<spec>], not [host::<spec>].
* A list of valid encodings can be retrieved using the command "iconv -l" on
  most *nix systems.
* If an invalid encoding is specified, a warning is logged during initial
  configuration and further input from that [<spec>] is discarded.
* If the source encoding is valid, but some characters from the [<spec>] are
  not valid in the specified encoding, then the characters are escaped
  as hex (for example, "\xF3").
* When set to "AUTO", Splunk software attempts to automatically
determine the
  character encoding and convert text from that encoding to UTF-8.
* For a complete list of the character sets Splunk software automatically
detects, see the online documentation.
* This setting applies at input time, when data is first read by Splunk
  software, such as on a forwarder that has configured inputs acquiring
  the data.
* Defaults to ASCII.

Line breaking

# Use the following settings to define the length of a line.

TRUNCATE = <non-negative integer>
* Change the default maximum line length (in bytes).
* Although this is in bytes, line length is rounded down when this would
  otherwise land mid-character for multi-byte characters.
* Set to 0 if you never want truncation (very long lines are, however, often
  a sign of garbage data).
* Defaults to 10000 bytes.
**LINE_BREAKER = <regular expression>**

* Specifies a regex that determines how the raw text stream is broken into initial events, before line merging takes place. (See the SHOULD_LINEMERGE setting, below)
  * Defaults to ([\r\n]+), meaning data is broken into an event for each line, delimited by any number of carriage return or newline characters.
  * The regex must contain a capturing group -- a pair of parentheses which defines an identified subcomponent of the match.
  * Wherever the regex matches, Splunk software considers the start of the first capturing group to be the end of the previous event, and considers the end of the first capturing group to be the start of the next event.
  * The contents of the first capturing group are discarded, and will not be present in any event. You are telling Splunk software that this text comes between lines.

* NOTE: You get a significant boost to processing speed when you use LINE_BREAKER to delimit multi-line events (as opposed to using SHOULD_LINEMERGE to reassemble individual lines into multi-line events).
  * When using LINE_BREAKER to delimit events, SHOULD_LINEMERGE should be set to false, to ensure no further combination of delimited events occurs.
  * Using LINE_BREAKER to delimit events is discussed in more detail in the documentation. Search the documentation for "configure event line breaking" for details.

** Special considerations for LINE_BREAKER with branched expressions **

When using LINE_BREAKER with completely independent patterns separated by pipes, some special issues come into play.

EG. LINE_BREAKER = pattern1|pattern2|pattern3

Note, this is not about all forms of alternation, eg there is nothing particular special about

example: LINE_BREAKER = ([\r\n])+(one|two|three)

where the top level remains a single expression.

A caution: Relying on these rules is NOT encouraged. Simpler is better, in
both regular expressions and the complexity of the behavior they rely on. If possible, it is strongly recommended that you reconstruct your regex to have a leftmost capturing group that always matches.

It may be useful to use non-capturing groups if you need to express a group before the text to discard.

EG. LINE_BREAKER = (?i:one|two)([\r\n]+)
* This will match the text one, or two, followed by any amount of newlines or carriage returns. The one-or-two group is non-capturing via the ?i prefix and will be skipped by LINE_BREAKER.

* A branched expression can match without the first capturing group matching, so the line breaker behavior becomes more complex. Rules:
  1: If the first capturing group is part of a match, it is considered the linebreak, as normal.
  2: If the first capturing group is not part of a match, the leftmost capturing group which is part of a match will be considered the linebreak.
  3: If no capturing group is part of the match, the linebreaker will assume that the linebreak is a zero-length break immediately preceding the match.

Example 1: LINE_BREAKER = end(\n)begin|end2(\n)begin2|begin3

* A line ending with 'end' followed a line beginning with 'begin' would match the first branch, and the first capturing group would have a match according to rule 1. That particular newline would become a break between lines.
* A line ending with 'end2' followed by a line beginning with 'begin2' would match the second branch and the second capturing group would have a match. That second capturing group would become the linebreak according to rule 2, and the associated newline would become a break between lines.
* The text 'begin3' anywhere in the file at all would match the third branch, and there would be no capturing group with a match. A linebreak would be assumed immediately prior to the text 'begin3' so a linebreak would be inserted prior to this text in accordance with rule 3. This means that a linebreak will occur before the text 'begin3' at any point in the text, whether a linebreak character exists or not.
Example 2: Example 1 would probably be better written as follows. This is not equivalent for all possible files, but for most real files would be equivalent.

```
LINE_BREAKER = end2?\nbegin(2|3)?
```

```
LINE_BREAKER_LOOKBEHIND = <integer>
* When there is leftover data from a previous raw chunk, LINE_BREAKER_LOOKBEHIND indicates the number of bytes before the end of the raw chunk (with the next chunk concatenated) that Splunk applies the LINE_BREAKER regex. You may want to increase this value from its default if you are dealing with especially large or multi-line events.
  * Defaults to 100 (bytes).
```

```
# Use the following settings to specify how multi-line events are handled.

SHOULD_LINEMERGE = [true|false]
* When set to true, Splunk software combines several lines of data into a single multi-line event, based on the following configuration settings.
  * Defaults to true.

# When SHOULD_LINEMERGE is set to true, use the following settings to define how Splunk software builds multi-line events.

BREAK_ONLY_BEFORE_DATE = [true|false]
* When set to true, Splunk software creates a new event only if it encounters a new line with a date.
  * Note, when using DATETIME_CONFIG = CURRENT or NONE, this setting is not meaningful, as timestamps are not identified.
  * Defaults to true.

BREAK_ONLY_BEFORE = <regular expression>
* When set, Splunk software creates a new event only if it encounters a new line that matches the regular expression.
  * Defaults to empty.

MUST_BREAK_AFTER = <regular expression>
* When set and the regular expression matches the current line, Splunk software creates a new event for the next input line.
  * Splunk software may still break before the current line if another rule
matches.
* Defaults to empty.

MUST_NOT_BREAK_AFTER = <regular expression>
* When set and the current line matches the regular expression, Splunk software
  does not break on any subsequent lines until the MUST_BREAK_AFTER expression
  matches.
* Defaults to empty.

MUST_NOT_BREAK_BEFORE = <regular expression>
* When set and the current line matches the regular expression, Splunk software does not break the last event before the current line.
* Defaults to empty.

MAX_EVENTS = <integer>
* Specifies the maximum number of input lines to add to any event.
* Splunk software breaks after the specified number of lines are read.
* Defaults to 256 (lines).

# Use the following settings to handle better load balancing from UF.
# Please note the EVENT_BREAKER properties are applicable for Splunk Universal
# Forwarder instances only.

EVENT_BREAKER_ENABLE = [true|false]
* When set to true, Splunk software will split incoming data with a
  light-weight chunked line breaking processor so that data is distributed
  fairly evenly amongst multiple indexers. Use this setting on the UF to
  indicate that data should be split on event boundaries across indexers especially for large files.
* Defaults to false

# Use the following to define event boundaries for multi-line events
# For single-line events, the default settings should suffice

EVENT_BREAKER = <regular expression>
* When set, Splunk software will use the setting to define an event boundary at
  the end of the first matching group instance.

**Timestamp extraction configuration**

DATETIME_CONFIG = <filename relative to $SPLUNK_HOME>
* Specifies which file configures the timestamp extractor, which
identifies timestamps from the event text.
* This configuration may also be set to "NONE" to prevent the timestamp
  extractor from running or "CURRENT" to assign the current system time
to each event.
* "CURRENT" will set the time of the event to the time that the event
  was merged from lines, or worded differently, the time it passed through the
  aggregator processor.
* "NONE" will leave the event time set to whatever time was selected by
  the input layer
  * For data sent by splunk forwarders over the splunk protocol, the input
    layer will be the time that was selected on the forwarder by its input
    behavior (as below).
  * For file-based inputs (monitor, batch) the time chosen will be the
    modification timestamp on the file being read.
  * For other inputs, the time chosen will be the current system time when
    the event is read from the pipe/socket/etc.
* Both "CURRENT" and "NONE" explicitly disable the per-text timestamp
  identification, so the default event boundary detection (BREAK_ONLY_BEFORE_DATE = true) is likely to not work as desired.
When using these settings, use SHOULD_LINEMERGE and/or the BREAK_ONLY_* , MUST_BREAK_* settings to control event merging.
* Defaults to /etc/datetime.xml (for example, $SPLUNK_HOME/etc/datetime.xml).

TIME_PREFIX = <regular expression>
* If set, Splunk software scans the event text for a match for this regex
  in event text before attempting to extract a timestamp.
* The timestamping algorithm only looks for a timestamp in the text
  following the end of the first regex match.
* For example, if TIME_PREFIX is set to "abc123", only text following the
  first occurrence of the text abc123 will be used for timestamp extraction.
* If the TIME_PREFIX cannot be found in the event text, timestamp extraction
  will not occur.
  * Defaults to empty.

MAX_TIMESTAMP_LOOKAHEAD = <integer>
* Specifies how far (in characters) into an event Splunk software should look
  for a timestamp.
* This constraint to timestamp extraction is applied from the point of the
  TIME_PREFIX-set location.
* For example, if TIME_PREFIX positions a location 11 characters into
  the event, and MAX_TIMESTAMP_LOOKAHEAD is set to 10, timestamp extraction
  will be constrained to characters 11 through 20.
* If set to 0, or -1, the length constraint for timestamp recognition is
  effectively disabled. This can have negative performance implications
  which scale with the length of input lines (or with event size when
  LINE_BREAKER is redefined for event splitting).
* Defaults to 128 (characters).

TIME_FORMAT = <strptime-style format>
* Specifies a strptime format string to extract the date.
* strptime is an industry standard for designating time formats.
* For more information on strptime, see "Configure timestamp
  recognition" in the online documentation.
* TIME_FORMAT starts reading after the TIME_PREFIX. If both are
  specified, the TIME_PREFIX regex must match up to and including the character
  before the TIME_FORMAT date.
* For good results, the <strptime-style format> should describe the day of
  the year and the time of day.
* Defaults to empty.

TZ = <timezone identifier>
* The algorithm for determining the time zone for a particular event is
  as follows:
  * If the event has a timezone in its raw text (for example, UTC, -08:00),
    use that.
  * If TZ is set to a valid timezone string, use that.
  * If the event was forwarded, and the forwarder-indexer connection is
    using the 6.0+ forwarding protocol, use the timezone provided by the
    forwarder.
  * Otherwise, use the timezone of the system that is running splunkd.
  * Defaults to empty.

TZ_ALIAS = <key=value>[,<key=value>]...
* Provides Splunk software admin-level control over how timezone strings
  extracted from events are interpreted.
  * For example, EST can mean Eastern (US) Standard time, or Eastern
    (Australian) Standard time. There are many other three letter
    timezone acronyms with many expansions.
* There is no requirement to use TZ_ALIAS if the traditional Splunk software
  default mappings for these values have been as expected. For example, EST
  maps to the Eastern US by default.
* Has no effect on TZ value; this only affects timezone strings from event
text, either from any configured TIME_FORMAT, or from pattern-based
guess fallback.
* The setting is a list of key=value pairs, separated by commas.
  * The key is matched against the text of the timezone specifier of the
    event, and the value is the timezone specifier to use when mapping
    the
timestamp to UTC/GMT.
  * The value is another TZ specifier which expresses the desired
    offset.
  * Example: TZ_ALIAS = EST=GMT+10:00 (See props.conf.example for more/full
    examples)
* Defaults to unset.

MAX_DAYS_AGO = <integer>
* Specifies the maximum number of days in the past, from the current
date as provided by input layer(For e.g. forwarder current time, or modtime
for files), that an extracted date can be valid. Splunk software still indexes
events with dates older than MAX_DAYS_AGO with the timestamp of the last
acceptable event. If no such acceptable event exists, new events with timestamps older
than MAX_DAYS_AGO will use the current timestamp.
* For example, if MAX_DAYS_AGO = 10, Splunk software applies the
timestamp of the last acceptable event to events with extracted timestamps older
than 10 days in the past. If no acceptable event exists, Splunk software
applies the current timestamp.
* Defaults to 2000 (days), maximum 10951.
* IMPORTANT: If your data is older than 2000 days, increase this setting.

MAX_DAYS_HENCE = <integer>
* Specifies the maximum number of days in the future, from the current
date as provided by input layer(For e.g. forwarder current time, or modtime
for files), that an extracted date can be valid. Splunk software still indexes
events with dates more than MAX_DAYS_HENCE in the future with the
timestamp
    of the last acceptable event. If no such acceptable event exists, new
    events
    with timestamps after MAX_DAYS_HENCE will use the current timestamp.
    * For example, if MAX_DAYS_HENCE = 3, Splunk software applies the
timestamp of
    the last acceptable event to events with extracted timestamps more
    than 3
days in the future. If no acceptable event exists, Splunk software applies
    the current timestamp.
    * The default value includes dates from one day in the future.
    * If your servers have the wrong date set or are in a timezone that is
    one
day ahead, increase this value to at least 3.
    * Defaults to 2 (days), maximum 10950.
    * IMPORTANT: False positives are less likely with a tighter window, change
    with caution.

MAX_DIFF_SECS_AGO = <integer>
* This setting prevents Splunk software from rejecting events with
timestamps
    that are out of order.
* Do not use this setting to filter events because Splunk software uses
    complicated heuristics for time parsing.
* Splunk software warns you if an event timestamp is more than <integer>
    seconds BEFORE the previous timestamp and does not have the same time
    format as the majority of timestamps from the source.
* After Splunk software throws the warning, it only rejects an event if it
    cannot apply a timestamp to the event (for example, if Splunk software
    cannot recognize the time of the event.)
* IMPORTANT: If your timestamps are wildly out of order, consider
    increasing
    this value.
* Note: if the events contain time but not date (date determined another
    way,
    such as from a filename) this check will only consider the hour. (No
    one
    second granularity for this purpose.)
* Defaults to 3600 (one hour), maximum 2147483646.

MAX_DIFF_SECS_HENCE = <integer>
* This setting prevents Splunk software from rejecting events with
timestamps
    that are out of order.
* Do not use this setting to filter events because Splunk software uses
    complicated heuristics for time parsing.
* Splunk software warns you if an event timestamp is more than <integer>
    seconds AFTER the previous timestamp and does not have the same time
    format
as the majority of timestamps from the source.
* After Splunk software throws the warning, it only rejects an event if it
cannot apply a timestamp to the event (for example, if Splunk software
cannot recognize the time of the event.)
* IMPORTANT: If your timestamps are wildly out of order, or you have
logs that
are written less than once a week, consider increasing this value.
* Defaults to 604800 (one week), maximum 2147483646.

ADD_EXTRA_TIME_FIELDS = [true|false]
* This setting controls whether or not the following keys will be
automatically
generated and indexed with events:
  date_hour, date_mday, date_minute, date_month, date_second,
  date_wday,
  date_year, date_zone, timestartpos, timeendpos, timestamp.
* These fields are never required, and may be turned off as desired.
* Defaults to true and is enabled for most data sources.

**Structured Data Header Extraction and configuration**

* This setting applies at input time, when data is first read by Splunk
  software, such as on a forwarder that has configured inputs acquiring
  the
data.

# Special characters for Structured Data Header Extraction:
# Some unprintable characters can be described with escape sequences.
The
# settings that can use these characters specifically mention that
# capability in their descriptions below.
# \f : form feed byte: 0x0c
# \s : space byte: 0x20
# \t : horizontal tab byte: 0x09
# \v : vertical tab byte: 0x0b

INDEXED_EXTRACTIONS = <CSV|TSV|PSV|W3C|JSON|HEC>
* Tells Splunk software the type of file and the extraction and/or
  parsing
  method Splunk software should use on the file.
  CSV - Comma separated value format
  TSV - Tab-separated value format
  PSV - pipe "|" separated value format
  W3C - W3C Extended Extended Log File Format
  JSON - JavaScript Object Notation format
  HEC - Interpret file as a stream of JSON events in the same format
       as the HTTP Event Collector input.
* These settings default the values of the remaining settings to the appropriate values for these known formats.
* Keep in mind that the HTTP Event Collector format allows the event to override many details on a per-event basis, such as the destination index. It should be only used to read data which is known to be well-formatted and safe, such as data output by locally written tools.
* Defaults to unset.

METRICS_PROTOCOL = <STATSD|COLLECTD_HTTP>
* Tells Splunk software which protocol the incoming metric data is using:
  STATSD - Supports statsd protocol, in the following format:
  <metric name>:<value>|<metric type>
  Use STATS-DIM-TRANSFORMS setting to manually extract dimensions for the above format. Splunk software auto-extracts dimensions when the data has "#" as dimension delimiter as shown below:
  <metric name>:<value>|<metric type>|#<dim1>:<val1>,<dim2>:<val2>...
  COLLECTD_HTTP - This is data from the write_http collectd plugin being parsed as streaming JSON docs with the _value living in "values" array and the dimension names in "dsnames" and the metric type (for example, counter vs gauge) is derived from "dstypes".
* Defaults to unset, for event (non-metric) data.

STATSD-DIM-TRANSFORMS = <statsd_dim_stanza_name1>,<statsd_dim_stanza_name2>..
* Used only when METRICS_PROTOCOL is set as statsd
* A comma separated list of transforms stanza names which are used to extract dimensions from statsd metric data.
* Optional for sourcetype which has only one transforms stanza for extracting dimensions and the stanza name is the same as that of sourcetype's name.

METRIC-SCHEMA-TRANSFORMS =
<metric-schema:stanza_name>[,<metric-schema:stanza_name>]...
* NOTE: This setting is valid only for index-time field extractions.
  You can set up the TRANSFORMS field extraction configuration to create index-time field extractions. The Splunk platform always applies METRIC-SCHEMA-TRANSFORMS after index-time field extraction takes place.
* Optional.
* A comma-separated list of metric-schema stanza names from transforms.conf that the Splunk platform uses to create multiple metrics from index-time field extractions of a single log event.
PREAMBLE_REGEX = <regex>
* Some files contain preamble lines. This setting specifies a regular
expression which allows Splunk software to ignore these preamble
lines,
   based on the pattern specified.

FIELD_HEADER_REGEX = <regex>
* A regular expression that specifies a pattern for prefixed headers.
   Note
   that the actual header starts after the pattern and it is not included
   in
   the header field.
* This setting supports the use of the special characters described
above.

HEADER_FIELD_LINE_NUMBER = <integer>
* Tells Splunk software the line number of the line within the file that
contains the header fields. If set to 0, Splunk software attempts to
locate the header fields within the file automatically.
* The default value is set to 0.

FIELD_DELIMITER = <character>
* Tells Splunk software which character delimits or separates fields in
the
   specified file or source.
* This setting supports the use of the special characters described
above.

HEADER_FIELD_DELIMITER = <character>
* Tells Splunk software which character delimits or separates header
fields in
   the specified file or source.
* This setting supports the use of the special characters described
above.

FIELD_QUOTE = <character>
* Tells Splunk software the character to use for quotes in the specified
file
   or source.
* This setting supports the use of the special characters described
above.

HEADER_FIELD_QUOTE = <character>
* Specifies the character to use for quotes in the header of the
specified file or source.
* This setting supports the use of the special characters described
above.

TIMESTAMP_FIELDS = [ <string>,..., <string>]
* Some CSV and structured files have their timestamp encompass multiple
fields in the event separated by delimiters. This setting tells Splunk software to specify all such fields which constitute the timestamp in a comma-separated fashion.
* If not specified, Splunk software tries to automatically extract the timestamp of the event.

FIELD_NAMES = [ <string>,..., <string>]
* Some CSV and structured files might have missing headers. This setting tells Splunk software to specify the header field names directly.

MISSING_VALUE_REGEX = <regex>
* Tells Splunk software the placeholder to use in events where no value is present.

JSON_TRIM_BRACES_IN_ARRAY_NAMES = <bool>
* Tell the json parser not to add the curly braces to array names.
* Note that enabling this will make json index-time extracted array field names inconsistent with spath search processor's naming convention.
* For a json document containing the following array object, with trimming enabled a indextime field 'mount_point' will be generated instead of the spath consistent field 'mount_point{}' "mount_point": ["/disk48","/disk22"]
* Defaults to false.

**Field extraction configuration**

NOTE: If this is your first time configuring field extractions in props.conf, review the following information first.

There are three different "field extraction types" that you can use to configure field extractions: TRANSFORMS, REPORT, and EXTRACT. They differ in two significant ways: 1) whether they create indexed fields (fields extracted at index time) or extracted fields (fields extracted at search time), and 2), whether they include a reference to an additional component called a "field transform," which you define separately in transforms.conf.

**Field extraction configuration: index time versus search time**

Use the TRANSFORMS field extraction type to create index-time field extractions. Use the REPORT or EXTRACT field extraction types to create
NOTE: Index-time field extractions have performance implications. Creating additions to the default set of indexed fields is ONLY recommended in specific circumstances. Whenever possible, extract fields only at search time.

There are times when you may find that you need to change or add to your set of indexed fields. For example, you may have situations where certain search-time field extractions are noticeably impacting search performance. This can happen when the value of a search-time extracted field exists outside of the field more often than not. For example, if you commonly search a large event set with the expression company_id=1 but the value 1 occurs in many events that do *not* have company_id=1, you may want to add company_id to the list of fields extracted by Splunk software at index time. This is because at search time, Splunk software will want to check each instance of the value 1 to see if it matches company_id, and that kind of thing slows down performance when you have Splunk searching a large set of data.

Conversely, if you commonly search a large event set with expressions like company_id!=1 or NOT company_id=1, and the field company_id nearly *always* takes on the value 1, you may want to add company_id to the list of fields extracted by Splunk software at index time.

For more information about index-time field extraction, search the documentation for "index-time extraction." For more information about search-time field extraction, search the online documentation for "search-time extraction."

**Field extraction configuration: field transforms vs. "inline" (props.conf only) configs**

The TRANSFORMS and REPORT field extraction types reference an additional component called a field transform, which you define separately in transforms.conf. Field transforms contain a field-extracting regular expression and other settings that govern the way that the transform extracts fields. Field transforms are always created in conjunction with field extraction stanzas in props.conf; they do not stand alone.

The EXTRACT field extraction type is considered to be "inline," which
means that it does not reference a field transform. It contains the regular expression that Splunk software uses to extract fields at search time. You can use EXTRACT to define a field extraction entirely within props.conf, no transforms.conf component is required.

**Search-time field extractions: Why use REPORT if EXTRACT will do?**

It's a good question. And much of the time, EXTRACT is all you need for search-time field extraction. But when you build search-time field extractions, there are specific cases that require the use of REPORT and the field transform that it references. Use REPORT if you want to:

* Reuse the same field-extracting regular expression across multiple sources, source types, or hosts. If you find yourself using the same regex to extract fields across several different sources, source types, and hosts, set it up as a transform, and then reference it in REPORT extractions in those stanzas. If you need to update the regex you only have to do it in one place. Handy!
* Apply more than one field-extracting regular expression to the same source, source type, or host. This can be necessary in cases where the field or fields that you want to extract from a particular source, source type, or host appear in two or more very different event patterns.
* Set up delimiter-based field extractions. Useful if your event data presents field-value pairs (or just field values) separated by delimiters such as commas, spaces, bars, and so on.
* Configure extractions for multivalued fields. You can have Splunk software append additional values to a field as it finds them in the event data.
* Extract fields with names beginning with numbers or underscores. Ordinarily, the key cleaning functionality removes leading numeric characters and underscores from field names. If you need to keep them, configure your field transform to turn key cleaning off.
* Manage formatting of extracted fields, in cases where you are extracting multiple fields, or are extracting both the field name and field value.

**Precedence rules for TRANSFORMS, REPORT, and EXTRACT field extraction types**

* For each field extraction, Splunk software takes the configuration from the highest precedence configuration stanza (see precedence rules at the beginning of this file).
* If a particular field extraction is specified for a source and a source type, the field extraction for source wins out.
* Similarly, if a particular field extraction is specified in ../local/ for a <spec>, it overrides that field extraction in ../default/.

TRANSFORMS-<class> = <transform_stanza_name>, <transform_stanza_name2>,...
* Used for creating indexed fields (index-time field extractions).
* <class> is a unique literal string that identifies the namespace of the field you're extracting.
  **Note:** <class> values do not have to follow field name syntax restrictions. You can use characters other than a-z, A-Z, and 0-9, and spaces are allowed. <class> values are not subject to key cleaning.
* <transform_stanza_name> is the name of your stanza from transforms.conf.
* Use a comma-separated list to apply multiple transform stanzas to a single TRANSFORMS extraction. Splunk software applies them in the list order. For example, this sequence ensures that the [yellow] transform stanza gets applied first, then [blue], and then [red]:
  [source::color_logs]
  TRANSFORMS-colorchange = yellow, blue, red

REPORT-<class> = <transform_stanza_name>, <transform_stanza_name2>,...
* Used for creating extracted fields (search-time field extractions) that reference one or more transforms.conf stanzas.
* <class> is a unique literal string that identifies the namespace of the field you're extracting.
  **Note:** <class> values do not have to follow field name syntax restrictions. You can use characters other than a-z, A-Z, and 0-9, and spaces are allowed. <class> values are not subject to key cleaning.
* <transform_stanza_name> is the name of your stanza from transforms.conf.
* Use a comma-separated list to apply multiple transform stanzas to a single REPORT extraction. Splunk software applies them in the list order. For example, this sequence insures that the [yellow] transform stanza gets applied first, then [blue], and then [red]:
  [source::color_logs]
  REPORT-colorchange = yellow, blue, red

EXTRACT-<class> = [<regex>|<regex> in <src_field>]
* Used to create extracted fields (search-time field extractions) that
do not reference transforms.conf stanzas.
* Performs a regex-based field extraction from the value of the source
field.
* `<class>` is a unique literal string that identifies the namespace of
the
  field you're extracting.
  **Note:** `<class>` values do not have to follow field name syntax
  restrictions. You can use characters other than a-z, A-Z, and 0-9, and
  spaces are allowed. `<class>` values are not subject to key cleaning.
* The `<regex>` is required to have named capturing groups. When the
  `<regex>`
  matches, the named capturing groups and their values are added to the
  event.
* dotall (\?) and multi-line (\m) modifiers are added in front of the
  regex.
  So internally, the regex becomes (\ms)<regex>.
* Use '<regex> in <src_field>' to match the regex against the values of
  a
    specific field. Otherwise it just matches against _raw (all raw event
data).
* NOTE: `<src_field>` has the following restrictions:
  * It can only contain alphanumerical characters and underscore
    (a-z, A-Z, 0-9, and _).
  * It must already exist as a field that has either been extracted at
    index time or has been derived from an EXTRACT-<class> configuration
    whose `<class>` ASCII value is *higher* than the configuration in
    which
      you are attempting to extract the field. For example, if you
      have an EXTRACT-ZZZ configuration that extracts `<src_field>`, then
      you can only use 'in `<src_field>' in an EXTRACT configuration with
      a `<class>` of 'aaa' or lower, as 'aaa' is lower in ASCII value
      than 'ZZZ'.
  * It cannot be a field that has been derived from a transform field
    extraction (REPORT-<class>), an automatic key-value field extraction
    (in which you configure the KV_MODE setting to be something other
    than 'none'), a field alias, a calculated field, or a lookup,
    as these operations occur after inline field extractions (EXTRACT-
    <class>) in the search time operations sequence.
* If your regex needs to end with 'in <string>' where <string> is *not*
  a
    field name, change the regex to end with '[i]n <string>' to ensure
    that Splunk software doesn't try to match <string> to a field name.

**kv_mode** = [none|auto|auto_escaped|multi|json|xml]
* Used for search-time field extractions only.
* Specifies the field/value extraction mode for the data.
* Set KV_MODE to one of the following:
  * none: if you want no field/value extraction to take place.
  * auto: extracts field/value pairs separated by equal signs.
* auto_escaped: extracts fields/value pairs separated by equal signs and honors " and \ as escaped sequences within quoted values, e.g field="value with \"nested\" quotes"

* multi: invokes the multikv search command to expand a tabular event into multiple events.

* xml: automatically extracts fields from XML data.

* json: automatically extracts fields from JSON data.

* Setting to 'none' can ensure that one or more user-created regexes are not overridden by automatic field/value extraction for a particular host, source, or source type, and also increases search performance.

* Defaults to auto.

* The 'xml' and 'json' modes will not extract any fields when used on data that isn't of the correct format (JSON or XML).

MATCH_LIMIT = <integer>
* Only set in props.conf for EXTRACT type field extractions.
* For REPORT and TRANSFORMS field extractions, set this in transforms.conf.
* Optional. Limits the amount of resources that will be spent by PCRE when running patterns that will not match.
* Use this to set an upper bound on how many times PCRE calls an internal function, match(). If set too low, PCRE may fail to correctly match a pattern.
* Defaults to 100000

DEPTH_LIMIT = <integer>
* Only set in props.conf for EXTRACT type field extractions.
* For REPORT and TRANSFORMS field extractions, set this in transforms.conf.
* Optional. Limits the amount of resources that are spent by PCRE when running patterns that will not match.
* Use this to limit the depth of nested backtracking in an internal PCRE function, match(). If set too low, PCRE might fail to correctly match a pattern.
* Default: 1000

AUTO_KV_JSON = [true|false]
* Used for search-time field extractions only.
* Specifies whether to try json extraction automatically.
* Defaults to true.

KV_TRIM_SPACES = true|false
* Modifies the behavior of KV_MODE when set to auto, and auto_escaped.
* Traditionally, automatically identified fields have leading and trailing whitespace removed from their values.
* Example event: 2014-04-04 10:10:45 myfield="apples "
would result in a field called 'myfield' with a value of 'apples'.
* If this value is set to false, then external whitespace then this
  outer space is retained.
* Example: 2014-04-04 10:10:45 myfield=" apples "
  would result in a field called 'myfield' with a value of ' apples '.
* The trimming logic applies only to space characters, not tabs, or
  other whitespace.
* NOTE: Splunk Web currently has limitations with displaying and
  interactively clicking on fields that have leading or trailing
  whitespace. Field values with leading or trailing spaces may not look
  distinct in the event viewer, and clicking on a field value will
  typically insert the term into the search string without its embedded spaces.
* These warts are not specific to this feature. Any such embedded spaces
  will behave this way.
* The Splunk search language and included commands will respect the
  spaces.
* Defaults to true.

CHECK_FOR_HEADER = [true|false]
* Used for index-time field extractions only.
* Set to true to enable header-based field extraction for a file.
* If the file has a list of columns and each event contains a field value
  (without field name), Splunk software picks a suitable header line to use
  for extracting field names.
* Can only be used on the basis of [〈sourcetype〉] or [source::〈spec〉],
  not [host::〈spec〉].
* Disabled when LEARN_SOURCETYPE = false.
* Will cause the indexed source type to have an appended numeral; for
  example, sourcetype-2, sourcetype-3, and so on.
* The field names are stored in etc/apps/learned/local/props.conf.
  * Because of this, this feature will not work in most environments
    where the data is forwarded.
* This setting applies at input time, when data is first read by Splunk
  software, such as on a forwarder that has configured inputs acquiring
  the data.
* Defaults to false.

SEDCMD-〈class〉 = 〈sed script〉
* Only used at index time.
* Commonly used to anonymize incoming data at index time, such as credit
  card or social security numbers. For more information, search the online
  documentation for "anonymize data."
* Used to specify a sed script which Splunk software applies to the _raw
  field.
* A sed script is a space-separated list of sed commands. Currently the following subset of sed commands is supported:
  * replace (s) and character substitution (y).
* Syntax:
  * replace - s/regex/replacement/flags
    * regex is a perl regular expression (optionally containing capturing groups).
  * replacement is a string to replace the regex match. Use \n for back references, where "n" is a single digit.
  * flags can be either: g to replace all matches, or a number to replace a specified match.
  * substitute - y/string1/string2/
    * substitutes the string1[i] with string2[i]

FIELDALIAS=<class> = (<orig_field_name> AS <new_field_name>)+
* Use this to apply aliases to a field. The original field is not removed.
  This just means that the original field can be searched on using any of its aliases.
* You can create multiple aliases for the same field.
* <orig_field_name> is the original name of the field.
* <new_field_name> is the alias to assign to the field.
* You can include multiple field alias renames in the same stanza.
* Field aliasing is performed at search time, after field extraction, but before calculated fields (EVAL-* statements) and lookups.
  This means that:
    * Any field extracted at search time can be aliased.
    * You can specify a lookup based on a field alias.
    * You cannot alias a calculated field.

EVAL=<fieldname> = <eval statement>
* Use this to automatically run the <eval statement> and assign the value of the output to <fieldname>. This creates a "calculated field."
* When multiple EVAL-* statements are specified, they behave as if they are run in parallel, rather than in any particular sequence.
  For example say you have two statements: EVAL-x = y*2 and EVAL-y=100. In this case, "x" will be assigned the original value of "y * 2," not the value of "y" after it is set to 100.
* Splunk software processes calculated fields after field extraction and field aliasing but before lookups. This means that:
  * You can use a field alias in the eval statement for a calculated field.
  * You cannot use a field added through a lookup in an eval statement for a calculated field.
LOOKUP-<class> = $TRANSFORM (<match_field> (AS <match_field_in_event>)?) + (OUTPUT|OUTPUTNEW (<output_field> (AS <output_field_in_event>)?) )+
* At search time, identifies a specific lookup table and describes how that lookup table should be applied to events.
* <match_field> specifies a field in the lookup table to match on.
  * By default Splunk software looks for a field with that same name in the event to match with (if <match_field_in_event> is not provided)
  * You must provide at least one match field. Multiple match fields are allowed.
* <output_field> specifies a field in the lookup entry to copy into each matching event, where it will be in the field <output_field_in_event>.
  * If you do not specify an <output_field_in_event> value, Splunk software uses <output_field>.
  * A list of output fields is not required.
  * If they are not provided, all fields in the lookup table except for the match fields (and the timestamp field if it is specified) will be output for each matching event.
  * If the output field list starts with the keyword "OUTPUTNEW" instead of "OUTPUT", then each output field is only written out if it did not previous exist. Otherwise, the output fields are always overridden. Any event that has all of the <match_field> values but no matching entry in the lookup table clears all of the output fields. NOTE that OUTPUTNEW behavior has changed since 4.1.x (where *none* of the output fields were written to if *any* of the output fields previously existed).
* Splunk software processes lookups after it processes field extractions, field aliases, and calculated fields (EVAL-* statements). This means that you can use extracted fields, aliased fields, and calculated fields to specify lookups. But you can't use fields discovered by lookups in the configurations of extracted fields, aliased fields, or calculated fields.
* The LOOKUP- prefix is actually case-insensitive. Acceptable variants include:
  LOOKUP_<class> = [...]  
  LOOKUP<class> = [...]  
  lookup_<class> = [...]
lookup<class> = [...] 

Binary file configuration

NO_BINARY_CHECK = [true|false]  
* When set to true, Splunk software processes binary files.  
* Can only be used on the basis of [<sourcetype>], or  
  [source::<source>],  
  not [host::<host>].  
* Defaults to false (binary files are ignored).  
* This setting applies at input time, when data is first read by Splunk  
  software, such as on a forwarder that has configured inputs acquiring  
  the  
  data.

detect_trailing_nulls = [auto|true|false]  
* When enabled, Splunk software tries to avoid reading in null bytes at  
  the end of a file.  
* When false, Splunk software assumes that all the bytes in the file should  
  be read and indexed.  
* Set this value to false for UTF-16 and other encodings (CHARSET) values  
  that can have null bytes as part of the character text.  
* Subtleties of 'true' vs 'auto':  
  * 'true' is the splunk-on-windows historical behavior of trimming all null  
    bytes.  
  * 'auto' is currently a synonym for true but will be extended to be sensitive to the charset selected (ie quantized for multi-byte  
    encodings, and disabled for unsafe variable-width encodings)  
* This feature was introduced to work around programs which foolishly preallocate their log files with nulls and fill in data later.  The  
  well-known case is Internet Information Server.  
* This setting applies at input time, when data is first read by Splunk  
  software, such as on a forwarder that has configured inputs acquiring  
  the  
  data.  
* Defaults to false on *nix, true on windows.

Segmentation configuration

SEGMENTATION = <segmenter>
* Specifies the segmenter from segmenters.conf to use at index time for
  the
  host, source, or sourcetype specified by `<spec>` in the stanza heading.
* Defaults to indexing.

```
SEGMENTATION-<segment selection> = <segmenter>
```
* Specifies that Splunk Web should use the specific segmenter (from
  segmenters.conf) for the given `segment selection` choice.
* Default `<segment selection>` choices are: all, inner, outer, raw. For
  more
  information see the Admin Manual.
* Do not change the set of default `<segment selection>` choices, unless
  you
  have some overriding reason for doing so. In order for a changed set
  of
  `<segment selection>` choices to appear in Splunk Web, you will need to
  edit
  the Splunk Web UI.

**File checksum configuration**

```
CHECK_METHOD = [endpoint_md5|entire_md5|modtime]
```
* Set `CHECK_METHOD = endpoint_md5` to have Splunk software checksum of
  the
  first and last 256 bytes of a file. When it finds matches, Splunk
  software
  lists the file as already indexed and indexes only new data, or
  ignores it if
  there is no new data.
* Set `CHECK_METHOD = entire_md5` to use the checksum of the entire file.
* Set `CHECK_METHOD = modtime` to check only the modification time of the
  file.
* Settings other than endpoint_md5 cause Splunk software to index the
  entire
  file for each detected change.
* Important: this option is only valid for `[source::<source>]` stanzas.
* This setting applies at input time, when data is first read by Splunk
  software, such as on a forwarder that has configured inputs acquiring
  the
  data.
* Defaults to endpoint_md5.

```
initCrcLength = <integer>
```
* See documentation in inputs.conf.spec.
Small file settings

PREFIX_SOURCETYPE = [true|false]
* NOTE: this setting is only relevant to the "[too_small]" sourcetype.
* Determines the source types that are given to files smaller than 100
  lines, and are therefore not classifiable.
* PREFIX_SOURCETYPE = false sets the source type to "too_small."
* PREFIX_SOURCETYPE = true sets the source type to
  "<sourcename>-too_small",
  where "<sourcename>" is a cleaned up version of the filename.
* The advantage of PREFIX_SOURCETYPE = true is that not all small
  files
  are classified as the same source type, and wildcard searching is
  often
  effective.
* For example, a Splunk search of "sourcetype=access*" will retrieve
  "access" files as well as "access-too_small" files.
* This setting applies at input time, when data is first read by Splunk
  software, such as on a forwarder that has configured inputs acquiring
  the
data.
* Defaults to true.

Sourcetype configuration

sourcetype = <string>
* Can only be set for a [source::...] stanza.
* Anything from that <source> is assigned the specified source type.
* Is used by file-based inputs, at input time (when accessing logfiles)
such
  as on a forwarder, or indexer monitoring local files.
* sourcetype assignment settings on a system receiving forwarded Splunk
  data
  will not be applied to forwarded data.
* For log files read locally, data from log files matching <source> is
  assigned the specified source type.
* Defaults to empty.

# The following setting/value pairs can only be set for a stanza that
# begins with [<sourcetype>]:

rename = <string>
* Renames [<sourcetype>] as <string> at search time
* With renaming, you can search for the [<sourcetype>] with
  sourcetype=<string>
* To search for the original source type without renaming it, use the field _sourcetype.
* Data from a renamed sourcetype will only use the search-time configuration for the target sourcetype. Field extractions (REPORTS/EXTRACT) for this stanza sourcetype will be ignored.
* Defaults to empty.

invalid_cause = <string>
* Can only be set for a \[<sourcetype>\] stanza.
* If invalid_cause is set, the Tailing code (which handles uncompressed logfiles) will not read the data, but hand it off to other components or throw an error.
* Set <string> to "archive" to send the file to the archive processor (specified in unarchive_cmd).
* When set to "winevt", this causes the file to be handed off to the Event Log input processor.
* Set to any other string to throw an error in the splunkd.log if you are running Splunklogger in debug mode.
* This setting applies at input time, when data is first read by Splunk software, such as on a forwarder that has configured inputs acquiring the data.
* Defaults to empty.

is_valid = [true|false]
* Automatically set by invalid_cause.
* This setting applies at input time, when data is first read by Splunk software, such as on a forwarder that has configured inputs acquiring the data.
* DO NOT SET THIS.
* Defaults to true.

force_local_processing = [true|false]
* Forces a universal forwarder to process all data tagged with this sourcetype locally before forwarding it to the indexers.
* Data with this sourcetype will be processed via the linebreaker, aggregator and the regrexreplacement processors in addition to the existing utf8 processor.
* Note that switching this property on will potentially increase the cpu and memory consumption of the forwarder.
* Applicable only on a universal forwarder.
* Defaults to false.

unarchive_cmd = <string>
* Only called if invalid_cause is set to "archive".
* This field is only valid on [source::<source>] stanzas.
* <string> specifies the shell command to run to extract an archived
source.
* Must be a shell command that takes input on stdin and produces output on
  stdout.
* Use _auto for Splunk software's automatic handling of archive files
  (tar,
   tar.gz, tgz, tbz, tbz2, zip)
* This setting applies at input time, when data is first read by Splunk
  software, such as on a forwarder that has configured inputs acquiring the
  data.
* Defaults to empty.

unarchive_sourcetype = <string>
* Sets the source type of the contents of the matching archive file. Use
  this field instead of the sourcetype field to set the source type of
  archive files that have the following extensions: gz, bz, bz2, Z.
* If this field is empty (for a matching archive file props lookup)
  Splunk software strips off the archive file's extension (.gz, bz etc) and
  lookup
  another stanza to attempt to determine the sourcetype.
* This setting applies at input time, when data is first read by Splunk
  software, such as on a forwarder that has configured inputs acquiring the
  data.
* Defaults to empty.

LEARN_SOURCETYPE = [true|false]
* Determines whether learning of known or unknown sourcetypes is
  enabled.
  * For known sourcetypes, refer to LEARN_MODEL.
  * For unknown sourcetypes, refer to the rule:: and delayedrule::
    configuration (see below).
  * Setting this field to false disables CHECK_FOR_HEADER as well (see
    above).
  * This setting applies at input time, when data is first read by Splunk
    software, such as on a forwarder that has configured inputs acquiring the
    data.
  * Defaults to true.

LEARN_MODEL = [true|false]
* For known source types, the file classifier adds a model file to the
  learned directory.
  * To disable this behavior for diverse source types (such as sourcecode,
    where there is no good example to make a sourcetype) set LEARN_MODEL =
      false.
  * This setting applies at input time, when data is first read by Splunk
    software, such as on a forwarder that has configured inputs acquiring the
    data.
* Defaults to true.

maxDist = <integer>
* Determines how different a source type model may be from the current file.
* The larger the maxDist value, the more forgiving Splunk software will be
  with differences.
* For example, if the value is very small (for example, 10), then files
  of the specified sourcetype should not vary much.
* A larger value indicates that files of the given source type can vary
  quite a bit.
* If you're finding that a source type model is matching too broadly, reduce
  its maxDist value by about 100 and try again. If you're finding that a source type model is being too restrictive, increase its maxDist value by
  about 100 and try again.
* This setting applies at input time, when data is first read by Splunk software, such as on a forwarder that has configured inputs acquiring the
data.
* Defaults to 300.

# rule:: and delayedrule:: configuration

MORE_THAN<optional_unique_value>_<number> = <regular expression>
(empty)
LESS_THAN<optional_unique_value>_<number> = <regular expression>
(empty)

* This setting applies at input time, when data is first read by Splunk software, such as on a forwarder that has configured inputs acquiring the
data.

An example:

```
[rule::bar_some]
sourcetype = source_with_lots_of_bars
# if more than 80% of lines have "----", but fewer than 70% have
"####"
  # declare this a "source_with_lots_of_bars"
MORE_THAN_80 = ----
LESS_THAN_70 = ####
```

A rule can have many MORE_THAN and LESS_THAN patterns, and all are required
for the rule to match.
Annotation Processor configured

ANNOTATE_PUNCT = [true|false]
* Determines whether to index a special token starting with "punct::"
  * The "punct::" key contains punctuation in the text of the event.
    * It can be useful for finding similar events
  * If it is not useful for your dataset, or if it ends up taking too much space in your index it is safe to disable it
* Defaults to true.

Header Processor configuration

HEADER_MODE = <empty> | always | firstline | none
* Determines whether to use the inline ***SPLUNK*** directive to rewrite index-time fields.
  * If "always", any line with ***SPLUNK*** can be used to rewrite index-time fields.
  * If "firstline", only the first line can be used to rewrite index-time fields.
  * If "none", the string ***SPLUNK*** is treated as normal data.
  * If <empty>, scripted inputs take the value "always" and file inputs take the value "none".
* This setting applies at input time, when data is first read by Splunk software, such as on a forwarder that has configured inputs acquiring the data.
* Defaults to <empty>.

Internal settings

# NOT YOURS. DO NOT SET.

_actions = <string>
* Internal field used for user-interface control of objects.
  * Defaults to "new,edit,delete".

pulldown_type = <bool>
* Internal field used for user-interface control of source types.
  * Defaults to empty.

given_type = <string>
* Internal field used by the CHECK_FOR_HEADER feature to remember the
original sourcetype.
* This setting applies at input time, when data is first read by Splunk software, such as on a forwarder that has configured inputs acquiring the data.
* Default to unset.

Sourcetype Category and Descriptions

description = <string>
* Field used to describe the sourcetype. Does not affect indexing behavior.
* Defaults to unset.

category = <string>
* Field used to classify sourcetypes for organization in the front end. Case sensitive. Does not affect indexing behavior.
* Defaults to unset.

props.conf.example

# Version 7.2.0
#
# The following are example props.conf configurations. Configure properties for your data.
# To use one or more of these configurations, copy the configuration block into props.conf in $SPLUNK_HOME/etc/system/local/. You must restart Splunk to enable configurations.
# To learn more about configuration files (including precedence) please see the documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

########
# Line merging settings
########
# The following example line-merges source data into multi-line events for
# apache_error sourcetype.

[apache_error]
SHOULD_LINEMERGE = True

########
# Settings for tuning
########

# The following example limits the amount of characters indexed per event from
# host::small_events.

[host::small_events]
TRUNCATE = 256

# The following example turns off DATETIME_CONFIG (which can speed up indexing)
# from any path that ends in /mylogs/*.log.
# In addition, the default splunk behavior of finding event boundaries via per-event timestamps can't work with NONE, so we disable
# SHOULD_LINEMERGE, essentially declaring that all events in this file are
# single-line.

[source:.../mylogs/*.log]
DATETIME_CONFIG = NONE
SHOULD_LINEMERGE = false

########
# Timestamp extraction configuration
########

# The following example sets Eastern Time Zone if host matches nyc*.

[host::nyc*]
TZ = US/Eastern

# The following example uses a custom datetime.xml that has been created and
# placed in a custom app directory. This sets all events coming in from hosts
# starting with dharma to use this custom file.
[host::dharma*]
DATETIME_CONFIG = <etc/apps/custom_time/datetime.xml>

########
## Timezone alias configuration
########
# The following example uses a custom alias to disambiguate the Australian
# meanings of EST/EDT
TZ_ALIAS = EST=GMT+10:00,EDT=GMT+11:00

# The following example gives a sample case wherein, one timezone field is
# being replaced by/interpreted as another.
TZ_ALIAS = EST=AEST,EDT=AEDT

########
# Transform configuration
########
# The following example creates a search field for host::foo if tied to a
# stanza in transforms.conf.
[host::foo]
TRANSFORMS-foo=foobar

# The following stanza extracts an ip address from _raw
[my_sourcetype]
EXTRACT-extract_ip = (?<ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})

# The following example shows how to configure lookup tables
[my_lookuptype]
LOOKUP-foo = mylookuptable userid AS myuserid OUTPUT username AS myusername

# The following shows how to specify field aliases
FIELDALIAS-foo = user AS myuser id AS myid

########
# Sourcetype configuration
########

# The following example sets a sourcetype for the file web_access.log
for a
# unix path.
[source:.../web_access.log]
sourcetype = splunk_web_access

# The following example sets a sourcetype for the Windows file iis6.log.
# Note:
# Backslashes within Windows file paths must be escaped.
[source::...\iis\iis6.log]
sourcetype = iis_access

# The following example extracts syslog events.
[syslog]
invalid_cause = archive
unarchive_cmd = gzip -cd -

# The following example learns a custom sourcetype and limits the range
# between
# different examples with a smaller than default maxDist.
[custom_sourcetype]
LEARN_MODEL = true
maxDist = 30

# rule:: and delayedrule:: configuration
# The following examples create sourcetype rules for custom sourcetypes
# with
# regex.

[rule::bar_some]
sourcetype = source_with_lots_of_bars
MORE_THAN_80 = ----

[delayedrule::baz_some]
sourcetype = my_sourcetype
LESS_THAN_70 = ####

#########
# File configuration
#########

# Binary file configuration
# The following example eats binary files from the sourcetype
# "imported_records".
[imported_records]
NO_BINARY_CHECK = true

500
# File checksum configuration
# The following example checks the entirety of every file in the
# web_access directory rather than skipping files that appear to be the same.

[source:.../web_access/*]
CHECK_METHOD = entire_md5

########
# Metric configuration
########

# A metric sourcetype of type statsd with 'regex_stanza1',
# 'regex_stanza2' to
# extract dimensions
[metric_sourcetype_name]
METRICS_PROTOCOL = statsd
STATSD-DIM-TRANSFORMS = regex_stanza1, regex_stanza2

# Convert a single log event into multiple metrics using
METRIC-SCHEMA-TRANSFORMS
# and index time extraction feature.
[logtometrics]
METRIC-SCHEMA-TRANSFORMS = metric-schema:logtometrics
TRANSFORMS-group = extract_group
TRANSFORMS-name = extract_name
TRANSFORMS-max_size_kb = extract_max_size_kb
TRANSFORMS-current_size_kb = extract_current_size_kb
TRANSFORMS-current_size = extract_current_size
TRANSFORMS-largest_size = extract_largest_size
TRANSFORMS-smallest_size = extract_smallest_size
category = metrics
should_linemerge = false

restmap.conf

The following are the spec and example files for restmap.conf.

restmap.conf.spec

# Version 7.2.0
#
# This file contains possible attribute and value pairs for creating new
# Representational State Transfer (REST) endpoints.
# There is a restmap.conf in $SPLUNK_HOME/etc/system/default/. To set custom configurations, place a restmap.conf in $SPLUNK_HOME/etc/system/local/. For help, see restmap.conf.example. You must restart Splunk to enable configurations.

# To learn more about configuration files (including precedence) please see the documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

# NOTE: You must register every REST endpoint via this file to make it available.

############################
# Global stanza

[global]
* This stanza sets global configurations for all REST endpoints.
* Follow this stanza name with any number of the following attribute/value pairs.

allowGetAuth=[true|false]
* Allow user/password to be passed as a GET parameter to endpoint services/auth/login.
* Setting this to true, while convenient, may result in user/password getting logged as cleartext in Splunk's logs *and* any proxy servers in between.
* Defaults to false.

allowRestReplay=[true|false]
* POST/PUT/DELETE requests can be replayed on other nodes in the deployment.
* This enables centralized management.
* Turn on or off this feature. You can also control replay at each endpoint level. This feature is currently INTERNAL and should not be turned on without consulting splunk support.
* Defaults to false

defaultRestReplayStanza=<string>
* Points to global rest replay configuration stanza.
* Related to allowRestReplay
* Defaults to "restreplayshc"

pythonHandlerPath=<path>
* Path to 'main' python script handler.
* Used by the script handler to determine where the actual 'main' script is located.
* Typically, you should not need to change this.
* Defaults to $SPLUNK_HOME/bin/rest_handler.py.

# Applicable to all REST stanzas
# Stanza definitions below may supply additional information for these.
#

[<rest endpoint name>:<endpoint description string>]
match=<path>
* Specify the URI that calls the handler.
* For example if match=/foo, then https://$SERVER:$PORT/services/foo calls this handler.
* NOTE: You must start your path with a /.

requireAuthentication=[true|false]
* This optional attribute determines if this endpoint requires authentication.
* Defaults to 'true'.

authKeyStanza=<stanza>
* This optional attribute determines the location of the pass4SymmKey in the server.conf to be used for endpoint authentication.
* Defaults to 'general' stanza.
* Only applicable if the requireAuthentication is set true.

restReplay=[true|false]
* This optional attribute enables rest replay on this endpoint group
* Related to allowRestReplay
* This feature is currently INTERNAL and should not be turned on without consulting splunk support.
* Defaults to false

restReplayStanza=<string>
* This points to stanza which can override the [global]/defaultRestReplayStanza value on a per endpoint/regex basis
* Defaults to empty

capability=<capabilityName>
capability.<post|delete|get|put>=<capabilityName>
* Depending on the HTTP method, check capabilities on the authenticated session user.
* If you use 'capability.post|delete|get|put,' then the associated method is
checked against the authenticated user's role.
* If you just use 'capability,' then all calls get checked against this
capability (regardless of the HTTP method).
* Capabilities can also be expressed as a boolean expression. Supported
operators
  include: or, and, ()

acceptFrom=<network_acl> ...
* Lists a set of networks or addresses to allow this endpoint to be
accessed
  from.
* This shouldn't be confused with the setting of the same name in the
  [httpServer] stanza of server.conf which controls whether a host can
make HTTP requests at all
* Each rule can be in the following forms:
  1. A single IPv4 or IPv6 address (examples: "10.1.2.3", "fe80::4a3")
  2. A CIDR block of addresses (examples: "10/8", "fe80:1234/32")
  3. A DNS name, possibly with a '*' used as a wildcard (examples:
      "myhost.example.com", "*.splunk.com")
  4. A single '*' which matches anything
* Entries can also be prefixed with '!' to cause the rule to reject the
connection. Rules are applied in order, and the first one to match is
used. For example, "!10.1/16, *" will allow connections from
everywhere
  except the 10.1.*.* network.
* Defaults to "*" (accept from anywhere)

includeInAccessLog=[true|false]
* If this is set to false, requests to this endpoint will not appear
  in splunkd_access.log
* Defaults to 'true'.

# Per-endpoint stanza
# Specify a handler and other handler-specific settings.
# The handler is responsible for implementing arbitrary namespace
underneath
# each REST endpoint.

[script:<uniqueName>]
* NOTE: The uniqueName must be different for each handler.
* Call the specified handler when executing this endpoint.
* The following attribute/value pairs support the script handler.

scripttype=python
* Tell the system what type of script to execute when using this
endpoint.
* Defaults to python.
* If set to "persist" it will run the script via a persistent-process
  that
    uses the protocol from persistconn/appserver.py.
**handler=<SCRIPT>.<CLASSNAME>**
* The name and class name of the file to execute.
* The file *must* live in an application's bin subdirectory.
* For example, `$SPLUNK_HOME/etc/apps/<APPNAME>/bin/TestHandler.py` has a class called `MyHandler` (which, in the case of python must be derived from a base class called `'splunk.rest.BaseRestHandler'`). The tag/value pair for this is:
  
  "handler=TestHandler.MyHandler".

**xsl=<path to XSL transform file>**
* Optional.
* Perform an optional XSL transform on data returned from the handler.
* Only use this if the data is XML.
* Does not apply to scripttype=persist.

**script=<path to a script executable>**
* For scripttype=python this is optional. It allows you to run a script which is *not* derived from `'splunk.rest.BaseRestHandler'`. This is rarely used. Do not use this unless you know what you are doing.
* For scripttype=persist this is the path with is sent to the driver to execute. In that case, environment variables are substituted.

**script.arg.<N>=<string>**
* Only has effect for scripttype=persist.
* List of arguments which are passed to the driver to start the script.
* The script can make use of this information however it wants.
* Environment variables are substituted.

**script.param=<string>**
* Optional.
* Only has effect for scripttype=persist.
* Free-form argument that is passed to the driver when it starts the script.
* The script can make use of this information however it wants.
* Environment variables are substituted.

**output_modes=<csv list>**
* Specifies which output formats can be requested from this endpoint.
* Valid values are: json, xml.
* Defaults to xml.

**passSystemAuth=<bool>**
* Specifies whether or not to pass in a system-level authentication token on each request.
* Defaults to false.

**driver=<path>**
* For scripttype=persist, specifies the command to start a persistent server for this process.
* Endpoints that share the same driver configuration can share processes.
* Environment variables are substituted.
* Defaults to using the persistconn/appserver.py server.

\[\text{driver.arg.<n>} = \langle\text{string}\rangle\]
* For scripttype=persist, specifies the command to start a persistent server for this process.
* Environment variables are substituted.
* Only takes effect when "driver" is specifically set.

\[\text{driver.env.<name>=<value>}\]
* For scripttype=persist, specifies an environment variable to set when running the driver process.

\[\text{passConf=<bool>}\]
* If set, the script is sent the contents of this configuration stanza as part of the request.
* Only has effect for scripttype=persist.
* Defaults to true.

\[\text{passPayload=[true | false | base64]}\]
* If set to true, sends the driver the raw, unparsed body of the POST/PUT as a "payload" string.
* If set to "base64", the same body is instead base64-encoded and sent as a "payload_base64" string.
* Only has effect for scripttype=persist.
* Defaults to false.

\[\text{passSession=<bool>}\]
* If set to true, sends the driver information about the user's session. This includes the user's name, an active authtoken, and other details.
* Only has effect for scripttype=persist.
* Defaults to true.

\[\text{passHttpHeaders=<bool>}\]
* If set to true, sends the driver the HTTP headers of the request.
* Only has effect for scripttype=persist.
* Defaults to false.

\[\text{passHttpCookies=<bool>}\]
* If set to true, sends the driver the HTTP cookies of the request.
* Only has effect for scripttype=persist.
* Defaults to false.


```
506
```

```
# 'admin'
# The built-in handler for the Extensible Administration Interface.
# Exposes the listed EAI handlers at the given URL.
```
[admin:<uniqueName>]

match=<partial URL>
* URL which, when accessed, will display the handlers listed below.

members=<csv list>
* List of handlers to expose at this URL.
* See https://localhost:8089/services/admin for a list of all possible
  handlers.

# 'admin_external'
# Register Python handlers for the Extensible Administration Interface.
# Handler will be exposed via its "uniqueName".
#
[admin_external:<uniqueName>]

handlertype=<script type>
* Currently only the value 'python' is valid.

handlerfile=<unique filename>
* Script to execute.
* For bin/myAwesomeAppHandler.py, specify only myAwesomeAppHandler.py.

handlerpersistentmode=[true|false]
* Set to true to run the script in persistent mode and keep the process
  running
  between requests.

handleractions=<comma separated list>
* List of EAI actions supported by this handler.
* Valid values are: create, edit, list, delete, _reload.

# Validation stanzas
# Add stanzas using the following definition to add arg validation to
# the appropriate EAI handlers.
[validation:<handler-name>]

<field> = <validation-rule>
* <field> is the name of the field whose value would be validated when
  an
  object is being saved.
* <validation-rule> is an eval expression using the validate() function
  to
  evaluate arg correctness and return an error message. If you use a
  boolean
  returning function, a generic message is displayed.
* <handler-name> is the name of the REST endpoint which this stanza applies to
handler-name is what is used to access the handler via
/servicesNS/<user>/<app/admin/<handler-name>.
  * For example:
    action.email.sendresult = validate(isbool('action.email.sendresults'), "'action.email.sendresults' must be a boolean value").
* NOTE: use ' or $ to enclose field names that contain non alphanumeric characters.

#############################
# 'eai'
# Settings to alter the behavior of EAI handlers in various ways.
# These should not need to be edited by users.
#
[eai:<EAI handler name>]

showInDirSvc = [true|false]
  * Whether configurations managed by this handler should be enumerated via the
directory service, used by SplunkWeb's "All Configurations" management page.
  Defaults to false.

desc = <human readable string>
  * Allows for renaming the configuration type of these objects when enumerated
  via the directory service.

#############################
# Miscellaneous
# The un-described parameters in these stanzas all operate according to the
# descriptions listed under "script:", above.
# These should not need to be edited by users - they are here only to quiet
# down the configuration checker.
#
[input:...]
dynamic = [true|false]
  * If set to true, listen on the socket for data.
  * If false, data is contained within the request body.
  * Defaults to false.

[peerupload:...]
path = <directory path>
  * Path to search through to find configuration bundles from search peers.
untar = [true|false]
* Whether or not a file should be untarred once the transfer is complete.

[restreplayshc]
methods = <comma separated strings>
* REST methods which will be replayed. POST, PUT, DELETE, HEAD, GET are the available options

nodelists = <comma separated string>
* strategies for replay. Allowed values are shc, nodes, filternodes
* shc - replay to all other nodes in Search Head Cluster
* nodes - provide raw comma separated URIs in nodes variable
* filternodes - filter out specific nodes. Always applied after other strategies

nodes = <comma separated management uris>
* list of specific nodes that you want the REST call to be replayed to

filternodes = <comma separated management uris>
* list of specific nodes that you do not want the REST call to be replayed to

[proxy:appsbrowser]
destination = <splunkbaseAPIURL>
* protocol, subdomain, domain, port, and path of the splunkbase api used to browse apps
* Defaults to https://splunkbase.splunk.com/api

**restmap.conf.example**

```bash
# Version 7.2.0
#
# This file contains example REST endpoint configurations.
#
# To use one or more of these configurations, copy the configuration block into
# restmap.conf in $SPLUNK_HOME/etc/system/local/. You must restart Splunk to
# enable configurations.
#
# To learn more about configuration files (including precedence) please see the
# documentation located at
#
http://docs.splunk.com/Documentation/Splunk/7.2.0/Admin/Aboutconfigurationfiles
```
# The following are default REST configurations. To create your own endpoints,
# modify the values by following the spec outlined in restmap.conf.spec.

#
# //////////////////////////////////////////////////////////////////////
# global settings
#
# //////////////////////////////////////////////////////////////////////

[global]

# indicates if auths are allowed via GET params
allowGetAuth=false

#The default handler (assuming that we have PYTHONPATH set)
pythonHandlerPath=$SPLUNK_HOME/bin/rest_handler.py

#
# //////////////////////////////////////////////////////////////////////
# internal C++ handlers
# NOTE: These are internal Splunk-created endpoints. 3rd party developers can
# only use script or search can be used as handlers.
# (Please see restmap.conf.spec for help with configurations.)
#
# //////////////////////////////////////////////////////////////////////

[SBA:sba]
match=/properties
capability=get_property_map

[asyncsearch:asyncsearch]
match=/search
capability=search

[indexing-preview:indexing-preview]
match=/indexing/preview
capability=(edit_monitor or edit_sourcetypes) and (edit_user and edit_tcp)
savedsearches.conf

The following are the spec and example files for savedsearches.conf.

savedsearches.conf.spec

# This file contains possible attribute/value pairs for saved search entries in
# savedsearches.conf. You can configure saved searches by creating your own
# savedsearches.conf.
#
# There is a default savedsearches.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/default. To
# set custom configurations, place a savedsearches.conf in
# $SPLUNK_HOME/etc/apps/SA-ITOA/local/. For examples, see
# savedsearches.conf.example. You must restart Splunk to enable configurations.
#
# To learn more about configuration files (including precedence) please see the
# documentation located at
#
http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

Ping

action.itsi_sample_event_action_ping.param.host = <string>
* Field in the event indicating the host to ping.
* Required. If no value is provided, no host will be pinged.
* Set to $result.host$ or $result.server$ etc...

Event generator settings

action.itsi_event_generator = <boolean>
* Whether the alert is enabled.

action.itsi_event_generator.param.title = <string>
* The title of the notable event in Episode Review.
* Optional. If title is not provided then the search name becomes the title.

action.itsi_event_generator.param.description = <string>
* A description of the notable event.
* Optional. If a description is not provided then the search description becomes the event description.

action.itsi_event_generator.param.owner = <string>
* The initial owner of the notable event.
* Optional. If an owner is not provided then default_owner is assigned.

action.itsi_event_generator.param.status = <string>
* The triage status of the event in Episode Review.
* Values must match an integer specified in the default version of itsi_notable_event_status.conf (or the local version if you created one).
* Optional. If a status is not provided then default_status is assigned.

action.itsi_event_generator.param.severity = <string>
* The level of importance of the event.
* Values must match an integer specified in the default version of itsi_notable_event_severity.conf (or the local version if you created one).
* Optional. If a severity is not provided then default_severity is assigned.

action.itsi_event_generator.param.drilldown_search_title = <string>
* You can drill down to a specific Splunk search from an event or episode.
* The name of the drilldown search link.
* Optional.

action.itsi_event_generator.param.drilldown_search_search = <string>
* The drilldown search string.
* Optional.

action.itsi_event_generator.param.drilldown_search_latest_offset = <seconds>
* Defines how far ahead from the time of the event, in seconds, to look for related events.
* This offset is added to the event time.
* Optional.

action.itsi_event_generator.param.drilldown_search_earliest_offset = <string>
* Defines how far back from the time of the event, in seconds, to start looking for related events.
* This offset is subtracted from the event time.
* Optional.

action.itsi_event_generator.param.drilldown_title = <string>
* You can drill down to a specific website from an event or episode.
* The name of the drilldown website link.
* Optional.
action.itsi_event_generator.param.drilldown_uri = <string>
* The URI of the website you drill down to.
* Optional.

action.itsi_event_generator.param.event_identifier_fields = <comma-separated list>
* A list of fields used to identify if a notable event is unique.
* Optional.
* This setting is useful for identifying if a given notable event is already present.
* ITSI usually builds a hash using this set of fields.

action.itsi_event_generator.param.service_ids = <comma-separated list>
* A list of service IDs representing one or more ITSI services to which this correlation search applies.
* Optional.

action.itsi_event_generator.param.entity_lookup_field = <string>
* The field in the data retrieved by the correlation search that is used to look up corresponding entities. For example, host.
* Optional.

action.itsi_event_generator.param.search_type = <string>
* The search type.
* Optional.
* Default: custom

action.itsi_event_generator.param.meta_data = <string>
* The search type of any stored metadata.
* Optional.

action.itsi_event_generator.param.is_ad_at = <boolean>
* Whether this correlation is created by enabling adaptive thresholding or anomaly detection (AT/AD) for KPIs or services.
* Optional.
* If "1", the correlation is created by AT/AD.
* If "0", the correlation is not created by AT/AD.

action.itsi_event_generator.param.ad_at_kpi_ids = <comma-separated list>
* A list of KPIs where AT/AD is enabled.
* Optional.

**savedsearches.conf.example**

```
# This is an example savedsearches.conf. Use this file to configure # saved searches.
# # To use one or more of these configurations, copy the configuration
```
block
# into savedsearches.conf in $SPLUNK_HOME/etc/apps/SA-ITOA/local.
# You must restart Splunk to enable configurations.
#
# To learn more about configuration files (including precedence) please
# see
# the documentation located at
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

[Test ITSI Reporting Search]

cron_schedule = */5 * * * *
disabled = False
dispatch.earliest_time = -5m
dispatch.latest_time = now
enableSched = True
search = | stats count | eval demo="Demo Search" | fields - count

action.itsi_event_generator = 1

action.itsi_event_generator.param.title = "Host $result.host$ is down"
action.itsi_event_generator.param.description = Test if host $result.host$ is down or not
action.itsi_event_generator.param.owner = admin
action.itsi_event_generator.param.status = 1
action.itsi_event_generator.param.severity = 2
action.itsi_event_generator.param.drilldown_search_title = Raw search of seeing $result.host$ events
action.itsi_event_generator.param.drilldown_search_search= index=_internal host="$result.host$"
action.itsi_event_generator.param.drilldown_search_latest_offset = 30
action.itsi_event_generator.param.drilldown_search_earliest_offset = -30
action.itsi_event_generator.param.drilldown_title = Go to deep dive "$result.sourcetype$"
action.itsi_event_generator.param.drilldown_uri = "'/en-US/app/itsi/search/"

[Test ITSI Notable Event Search]

cron_schedule = */5 * * * *
disabled = False
dispatch.earliest_time = -5m
dispatch.latest_time = now
enableSched = True
search = index=_internal | head 4

alert.digest_mode = 0

action.itsi_event_generator = 1

action.itsi_event_generator.param.title = "Host $result.host$ is down"

action.itsi_event_generator.param.description = Test if host $result.host$ is down or not

action.itsi_event_generator.param.owner = admin

action.itsi_event_generator.param.status = 1

action.itsi_event_generator.param.severity = 2

action.itsi_event_generator.param.drilldown_search_title = Raw search of seeing $result.host$ events

action.itsi_event_generator.param.drilldown_search_search = index=_internal host=$result.host$

action.itsi_event_generator.param.drilldown_search_latest_offset = 30

action.itsi_event_generator.param.drilldown_search_earliest_offset = -30

action.itsi_event_generator.param.drilldown_title = Go to deep dive "$result.sourcetype$"

action.itsi_event_generator.param.drilldown_uri = "/en-US/app/itsi/search/"

**searchbnf.conf**

The following are the spec and example files for searchbnf.conf.

**searchbnf.conf.spec**

```yaml
# Version 7.2.0
#
```

515
# This file contain descriptions of stanzas and attribute/value pairs
# for
# configuring search-assistant via searchbnf.conf
#
# There is a searchbnf.conf in $SPLUNK_HOME/etc/system/default/. It
# should
# not be modified. If your application has its own custom python search
# commands, your application can include its own searchbnf.conf to
describe
# the commands to the search-assistant.
#
# To learn more about configuration files (including precedence) please
see
# the documentation located at
#
http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

**GLOBAL SETTINGS**

# Use the [default] stanza to define any global settings.
#  * You can also define global settings outside of any stanza, at the
top
#    of the file.
#  * Each conf file should have at most one default stanza. If there are
#    multiple default stanzas, attributes are combined. In the case of
#    multiple definitions of the same attribute, the last definition in
#    the
#    file wins.
#  * If an attribute is defined at both the global level and in a
#    specific
#    stanza, the value in the specific stanza takes precedence.

**[<search-commandname>-command]**

* This stanza enables properties for a given <search-command>.
* A searchbnf.conf file can contain multiple stanzas for any number of
  commands. * Follow this stanza name with any number of the following
  attribute/value pairs.
* If you do not set an attribute for a given <spec>, the default is
  used.
  The default values are empty.
* An example stanza name might be "geocode-command", for a "geocode"
  command.
* Search command stanzas can refer to definitions defined in others
  stanzas,
  and they do not require "-command", appended to them. For example:
[geocode-command]

    syntax = geocode <geocode-option>*

...  

[geocode-option]

    syntax = (maxcount=<int>) | (maxhops=<int>)

...  

#**************************************************************************
# The possible attributes/value pairs for searchbnf.conf
#**************************************************************************

syntax = <string>
* Describes the syntax of the search command. See the head of
  searchbnf.conf for details.
* Required

simplesyntax = <string>
* Optional simpler version of the syntax to make it easier to
  understand at the expense of completeness. Typically it removes
  rarely used options or alternate ways of saying the same thing.
* For example, a search command might accept values such as
  "m|min|mins|minute|minutes", but that would unnecessarily
  clutter the syntax description for the user. In this can, the
  simplesyntax can just pick the one (e.g., "minute").

alias = <commands list>
* Alternative names for the search command. This further cleans
  up the syntax so the user does not have to know that
  'savedsearch' can also be called by 'macro' or 'savedsplunk'.

description = <string>
* Detailed text description of search command. Description can continue
  on
  the next line if the line ends in "\"
* Required

shortdesc = <string>
* A short description of the search command. The full DESCRIPTION
  may take up too much screen real-estate for the search assistant.
* Required

exampi<index> = <string>

comment<index> = <string>
'example' should list out a helpful example of using the search command, and 'comment' should describe that example.

'example' and 'comment' can be appended with matching indexes to allow multiple examples and corresponding comments.

For example:
- example2 = geocode maxcount=4
  command2 = run geocode on up to four values
- example3 = geocode maxcount=-1
  command3 = run geocode on all values

**usage = public|private|deprecated**
- Determines if a command is public, private, depreciated. The search assistant only operates on public commands.
- Required

**tags = <tags list>**
- List of tags that describe this search command. Used to find commands when the user enters a synonym (e.g. "graph" -> "chart")

**related = <commands list>**
- List of related commands to help user when using one command to learn about others.

#******************************************************************************
# Optional attributes primarily used internally at Splunk
#******************************************************************************

appears-in = <string>
category = <string>
maintainer = <string>
note = <string>
optout-in = <string>
supports-multivalue = <string>

**searchbnf.conf.example**

# Version 7.2.0
#
# The following are example stanzas for searchbnf.conf configurations.
#

##################
# selfjoin
##################
[selfjoin-command]
syntax = selfjoin (<selfjoin-options>)* <field-list>
shortdesc = Join results with itself.
description = Join results with itself. Must specify at least one field
to join on.
usage = public
eample1 = selfjoin id
comment1 = Joins results with itself on 'id' field.
related = join
tags = join combine unite

[selfjoin-options]
syntax = overwrite=<bool> | max=<int> | keepsingle=<int>
description = The selfjoin joins each result with other results that\nhave the same value for the join fields. 'overwrite' controls if\fields from these 'other' results should overwrite fields of the\result used as the basis for the join (default=true). max indicates\the maximum number of 'other' results each main result can join with.\(default = 1, 0 means no limit). 'keepsingle' controls whether or\not\results with a unique value for the join fields (and thus no other\results to join with) should be retained. (default = false)

**service_analyzer_settings.conf**

The following are the spec and example files for

**service_analyzer_settings.conf**

**service_analyzer_settings.conf.spec**

```bash
# This file contains a setting for determining whether or not to
# always display the First Time Run modal in the Service Analyzer.
#
# To set custom configurations, create a drilldownsearch_offset.conf in
# $SPLUNK_HOME/etc/apps/itsi/local/. You must restart Splunk to enable
# configurations.
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
#
# http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles

[settings]

ftr_override = [0|1]
* A special setting that determines whether or not to always display
  the First Time Run (FTR) modal in the Service Analyzer.
```
* If "1", every time you navigate to the Service Analyzer, the First Time Run modal is displayed.
* If "0", the behavior defaults to showing the FTR modal only when services are not present.
* Default: 0 (false)

**service_analyzer_settings.conf.example**

No example

**threshold_labels.conf**

The following are the spec and example files for threshold_labels.conf.

**threshold_labels.conf.spec**

```plaintext
# Copyright (C) 2005-2018 Splunk Inc. All Rights Reserved.
#
# This file contains all possible attribute/value pairs for configuring settings
# for severity-level thresholds. Use this file to configure
# threshold names and color mappings.
#
# To map threshold names and colors, place a threshold_label.conf in
# $SPLUNK_HOME/etc/apps/itsi/local/. For examples, see
# threshold_label.conf.example.
#
# To learn more about configuration files (including precedence) see the
documentation
# located at http://www.splunk.com/base/Documentation/latest/Admin/Aboutconfigurationfiles
#
# CAUTION: You can drastically affect your Splunk installation by
# changing any settings in
# this file other than the colors. Consult technical support
# (http://www.splunk.com/page/submit_issue)
# if you are not sure how to configure this file.

[<name>]

color = <string>
* A valid color code.
* Required.
```
lightcolor = <string>
* A valid color code to display for Episode Review "prominent mode".
* When you view Episode Review in prominent mode, the entire row is
colored
    rather than just the colored band on the side.
* Required.

threshold_level = <integer>
* A threshold level that is used to create an ordered list of the
labels.
* For example, if you set the 'Normal' threshold level to "1", it
appears
    first when the levels are listed in the UI.
* Optional.

health_weight = <integer>
* The weight or importance of this status.
* This value should be between 0 and 1.
* In general, regular levels like Normal and Critical have a weight of
"1", while
    less important levels like Maintenance and Info have a weight of "0".
* Required.

health_min = <integer>
* The minimum threshold value.
* This value must be a number between 0 and 100. 0 and 100 are inclusive
but
    the minimum threshold value is exclusive.
* Required.

health_max = <integer>
* The maximum threshold value.
* This value must be a number between 0 and 100. 0 and 100 are inclusive
but
    the maximum threshold value is exclusive.
* Required.

score_contribution = <integer>
* The number, traditionally from 0 to 100, that this particular level
will
    contribute towards health score calculations.
* Required.

threshold_labels.conf.example

# Copyright (C) 2005-2018 Splunk Inc. All Rights Reserved.
# This is an example threshold_labels.conf. Use this file to
# configure settings for severity-level thresholds.
To use one or more of these configurations, copy the color code into threshold_labels.conf in $SPLUNK_HOME/etc/apps/itsi/local. You must restart Splunk to enable configurations.

To learn more about configuration files (including precedence) please see the documentation located at http://docs.splunk.com/Documentation/ITSI/latest/Configure/ListofITSIconfigurationfiles.

This example alert includes showing raw events at selected time buckets, showing raw events from a displayed time range, showing KPI events for a host, and showing all events for a host.

This file contains examples of brighter severity colors, with "Normal" severity being replaced with "Low" severity.

[info]
color = #6AB7C7
threshold_level = 1

[low]
color = #65A637
threshold_level = 2

[medium]
color = #FAC51C
threshold_level = 3

[high]
color = #F7902B
threshold_level = 4

[critical]
color = #D85D3C
threshold_level = 5

threshold_periods.conf

The following are the spec and example files for threshold_periods.conf.
threshold_periods.conf.spec

# threshold_periods.conf is DEPRECATED and should not be edited.

[<threshold-period-number>]

past = <value>
* Label for how far in the past.

description=<value>
* The description.

relative=<value>
* Relative time range.

threshold_periods.conf.example

No example

transforms.conf

The following are the spec and example files for transforms.conf.

transforms.conf.spec

#   Version 7.2.0
#
# This file contains settings and values that you can use to configure
# data transformations.
#
# Transforms.conf is commonly used for:
#   * Configuring host and source type overrides that are based on regular
#     expressions.
#   * Anonymizing certain types of sensitive incoming data, such as credit
#     card or social security numbers.
#   * Routing specific events to a particular index, when you have
#     multiple
#     indexes.
#   * Creating new index-time field extractions. NOTE: We do not recommend
#     adding to the set of fields that are extracted at index time unless
#     it
is absolutely necessary because there are negative performance
implications.
* Creating advanced search-time field extractions that involve one or
more of the following:
  * Reuse of the same field-extracting regular expression across
  multiple sources, source types, or hosts.
  * Application of more than one regular expression to the same
  source,
  * source type, or host.
  * Using a regular expression to extract one or more values from the
  values of another field.
  * Delimiter-based field extractions, such as extractions where the
  field-value pairs are separated by commas, colons, semicolons,
  bars, or something similar.
  * Extraction of multiple values for the same field.
  * Extraction of fields with names that begin with numbers or
  underscores.
  * NOTE: Less complex search-time field extractions can be set up
    entirely in props.conf.
  * Setting up lookup tables that look up fields from external sources.

All of the above actions require corresponding settings in props.conf.

You can find more information on these topics by searching the Splunk
documentation (http://docs.splunk.com/Documentation).

There is a transforms.conf file in $SPLUNK_HOME/etc/system/default/.
To set custom configurations, place a transforms.conf file in
$SPLUNK_HOME/etc/system/local/.

For examples of transforms.conf configurations, see the
transforms.conf.example file.

You can enable configuration changes made to transforms.conf by running this
search in Splunk Web:

```
| extract reload=t
```

To learn more about configuration files (including precedence) please see
the documentation located at
http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the
top
#    of the file.
# * Each conf file should have at most one default stanza. If there are
#    multiple default stanzas, settings are combined. In the case of
#    multiple definitions of the same setting, the last definition in
#    the
#    file wins.
# * If a setting is defined at both the global level and in a specific
#    stanza, the value in the specific stanza takes precedence.

[<unique_transform_stanza_name>]
* Name your stanza. Use this name when you configure field extractions,
lookup tables, and event routing in props.conf. For example, if you
are
setting up an advanced search-time field extraction, in props.conf you
would add REPORT-class = <unique_transform_stanza_name> under the
[<spec>] stanza that corresponds with a stanza you’ve created in
transforms.conf.
* Follow this stanza name with any number of the following setting/value
pairs, as appropriate for what you intend to do with the transform.
* If you do not specify an entry for each setting, Splunk software uses
the default value.

REGEX = <regular expression>
* Enter a regular expression to operate on your data.
* NOTE: This setting is valid for index-time and search-time field
extraction.
* REGEX is required for all search-time transforms unless you are
setting up
an ASCII-only delimiter-based field extraction, in which case you can
use
DELIMS (see the DELIMS setting description, below).
* REGEX is required for all index-time transforms.
* REGEX and the FORMAT setting:
  * Name-capturing groups in the REGEX are extracted directly to fields.
    This means that you do not need to specify the FORMAT setting for
    simple field extraction cases (see the description of FORMAT,
    below).
  * If the REGEX extracts both the field name and its corresponding
    field
    value, you can use the following special capturing groups if you
    want to
    skip specifying the mapping in FORMAT:
    _KEY_<string>, _VAL_<string>.
* For example, the following are equivalent:
  * Using FORMAT:
    * REGEX  = ([a-z]+)=([a-z]+)
    * FORMAT = $1::$2
  * Without using FORMAT
    * REGEX  = (?<_KEY_1>[a-z]+)=(?<_VAL_1>[a-z]+)
    * When using either of the above formats, in a search-time extraction,
      the regular expression attempts to match against the source text,
      extracting as many fields as can be identified in the source text.
  * Default: empty string

FORMAT = <string>
* NOTE: This option is valid for both index-time and search-time field extraction. However, FORMAT behaves differently depending on whether the extraction is performed at index time or search time.
* This setting specifies the format of the event, including any field names or values you want to add.
* FORMAT for index-time extractions:
  * Use $n (for example $1, $2, etc) to specify the output of each REGEX match.
  * If REGEX does not have n groups, the matching fails.
  * The special identifier $0 represents what was in the DEST_KEY before the REGEX was performed.
  * At index time only, you can use FORMAT to create concatenated fields:
    * Example: FORMAT = ipaddress::$1.$2.$3.$4
    * When you create concatenated fields with FORMAT, "$" is the only special character. It is treated as a prefix for regular expression capturing groups only if it is followed by a number and only if the number applies to an existing capturing group. So if REGEX has only one capturing group and its value is "bar", then:
      * "FORMAT = foo$1" yields "foobar"
      * "FORMAT = foo$bar" yields "foo$bar"
      * "FORMAT = foo$1234" yields "foo$1234"
      * "FORMAT = foo$1\$2" yields "foo$bar\$2"
  * At index-time, FORMAT defaults to <stanza-name>::$1
* FORMAT for search-time extractions:
  * The format of this field as used during search time extractions is as follows:
    * FORMAT = <field-name>::<field-value>(<field-name>::<field-value>)*
where:
* field-name = [string]|$<extracting-group-number>]
* field-value = [string]|$<extracting-group-number>]

* Search-time extraction examples:
  * 1. FORMAT = first::$1 second::$2 third::other-value
  * 2. FORMAT = $1::$2

* If you configure FORMAT with a variable <field-name>, such as in the second example above, the regular expression is repeatedly applied to the source key to match and extract all field/value pairs in the event.
  * When you use FORMAT to set both the field and the value (such as FORMAT = third::other-value), and the value is not an indexed token, you must set the field to INDEXED_VALUE = false in fields.conf. Not doing so can cause inconsistent search results.

* NOTE: You cannot create concatenated fields with FORMAT at search time. That functionality is only available at index time.
  * At search-time, FORMAT defaults to an empty string.

MATCH_LIMIT = <integer>
* Only set in transforms.conf for REPORT and TRANSFORMS field extractions.
  * For EXTRACT type field extractions, set this in props.conf.
  * Optional. Limits the amount of resources that are spent by PCRE when running patterns that do not match.
  * Use this to set an upper bound on how many times PCRE calls an internal function, match(). If set too low, PCRE may fail to correctly match a pattern.
  * Default: 100000

DEPTH_LIMIT = <integer>
* Only set in transforms.conf for REPORT and TRANSFORMS field extractions.
  * For EXTRACT type field extractions, set this in props.conf.
  * Optional. Limits the amount of resources that are spent by PCRE when running patterns that do not match.
  * Use this to limit the depth of nested backtracking in an internal PCRE function, match(). If set too low, PCRE might fail to correctly match a pattern.
  * Default: 1000

CLONE_SOURCETYPE = <string>
* This name is wrong; a transform with this setting actually clones and modifies events, and assigns the new events the specified source type.
  * If CLONE_SOURCETYPE is used as part of a transform, the transform creates a modified duplicate event for all events that the transform is applied
to via
    normal props.conf rules.
* Use this setting when you need to store both the original and a
  modified
  form of the data in your system, or when you need to send the
  original and
  a modified form to different outbound systems.
* A typical example would be to retain sensitive information according
to
  one policy and a version with the sensitive information removed
  according to another policy. For example, some events may have data
  that you must retain for 30 days (such as personally identifying
  information) and only 30 days with restricted access, but you need
  that
  event retained without the sensitive data for a longer time with
  wider
  access.
* Specifically, for each event handled by this transform, a near-exact
  copy
  is made of the original event, and the transformation is applied to
  the
  copy. The original event continues along normal data processing
  unchanged.
* The <string> used for CLONE_SOURCETYPE selects the source type that is
  used
  for the duplicated events.
* The new source type MUST differ from the the original source type. If
  the
  original source type is the same as the target of the
CLONE_SOURCETYPE,
  Splunk software makes a best effort to log warnings to splunkd.log,
  but this
  setting is silently ignored at runtime for such cases, causing the
  transform
  to be applied to the original event without cloning.
* The duplicated events receive index-time transformations & sed
  commands for all transforms that match its new host, source, or source
  type.
* This means that props.conf matching on host or source will
  incorrectly be
  applied a second time.
* Can only be used as part of of an otherwise-valid index-time
  transform. For
  example REGEX is required, there must be a valid target (DEST_KEY or
  WRITE_META), etc as above.

LOOKAHEAD = <integer>
* NOTE: This option is valid for all index time transforms, such as
  index-time field creation, or DEST_KEY modifications.
* Optional. Specifies how many characters to search into an event.
* Default: 4096
  * You may want to increase this value if you have event line lengths
that exceed 4096 characters (before linebreaking).

WRITE_META = [true|false]
* NOTE: This setting is only valid for index-time field extractions.
* Automatically writes REGEX to metadata.
* Required for all index-time field extractions except for those where
  DEST_KEY = _meta (see the description of the DEST_KEY setting, below)
* Use instead of DEST_KEY = _meta.
* Default: false

DEST_KEY = <KEY>
* NOTE: This setting is only valid for index-time field extractions.
* Specifies where Splunk software stores the expanded FORMAT results in
  accordance with the REGEX match.
* Required for index-time field extractions where WRITE_META = false or
  is not set.
* For index-time extractions, DEST_KEY can be set to a number of values
  mentioned in the KEYS section at the bottom of this file.
  * If DEST_KEY = _meta (not recommended) you should also add $0 to the
    start of your FORMAT setting. $0 represents the DEST_KEY value
    before Splunk software performs the REGEX (in other words, _meta).
  * The $0 value is in no way derived *from* the REGEX match. (It
    does not represent a captured group.)
* KEY names are case-sensitive, and should be used exactly as they
  appear in the KEYS list at the bottom of this file. (For example, you would say
  DEST_KEY = MetaData:Host, *not* DEST_KEY = metadata:host .)

DEFAULT_VALUE = <string>
* NOTE: This setting is only valid for index-time field extractions.
* Optional. The Splunk software writes the DEFAULT_VALUE to DEST_KEY if the
  REGEX fails.
* Default: empty string

SOURCE_KEY = <string>
* NOTE: This setting is valid for both index-time and search-time field
  extractions.
* Optional. Defines the KEY that Splunk software applies the REGEX to.
* For search time extractions, you can use this setting to extract one
  or more values from the values of another field. You can use any field
  that is available at the time of the execution of this field extraction
  * For index-time extractions use the KEYS described at the bottom of
    this file.
  * KEYs are case-sensitive, and should be used exactly as they appear
in

the KEYS list at the bottom of this file. (For example, you would say

SOURCE_KEY = MetaData:Host, *not* SOURCE_KEY = metadata:host .)

* If <string> starts with "field:" or "fields:" the meaning is changed.
  Instead of looking up a KEY, it instead looks up an already indexed
  field.
  For example, if a CSV field name "price" was indexed then
  "SOURCE_KEY = field:price" causes the REGEX to match against the
  contents
  of that field. It's also possible to list multiple fields here with
  "SOURCE_KEY = fields:name1,name2,name3" which causes MATCH to be run
  against a string comprising of all three values, separated by space
  characters.
  * SOURCE_KEY is typically used in conjunction with REPEAT_MATCH in
    index-time field transforms.
  * Default: _raw
  * This means it is applied to the raw, unprocessed text of all events.

REPEAT_MATCH = [true|false]

* NOTE: This setting is only valid for index-time field extractions.
  * Optional. When set to true, Splunk software runs the REGEX multiple
    times on the SOURCE_KEY.
  * REPEAT_MATCH starts wherever the last match stopped, and continues
    until
    no more matches are found. Useful for situations where an unknown
    number
    of REGEX matches are expected per event.
  * Default: false

INGEST_EVAL = <comma-separated list of evaluator expressions>

* NOTE: This setting is only valid for index-time field extractions.
  * Optional. When you set INGEST_EVAL, this setting overrides all of the
    other
    index-time settings (such as REGEX, DEST_KEY, etc) and declares the
    index-time extraction to be evaluator-based.
  * The expression takes a similar format to the search-time "|eval"
    command.
  For example "a=b+c*d" Just like the search-time operator, you can
  string multiple expressions together, separated by commas like
  "len=length(_raw), length_category=floor(log(len,2))".
  * Keys which are commonly used with DEST_KEY or SOURCE_KEY (like
    "_raw", "queue", etc) can be used directly in the expression.
  Also available are values which would be populated by default when
  this event is searched ("source", "sourcetype", "host",
  "splunk_server",
  "linecount", "index"). Search-time calculated fields (the "EVAL-"
  settings
  in props.conf) are NOT available.
  * When INGEST_EVAL accesses the "_time" variable, subsecond information
    is
    included. This is unlike regular-expression-based index-time
extractions,
where "_time" values are limited to whole seconds.
* By default, other variable names refer to index-time fields which are
populated in "_meta" So an expression 'event_category=if(_raw LIKE
"WARN %", "warning", "normal")' would append a new indexed field to _meta like
"event_category::warning".
* You can force a variable to be treated as a direct KEY name by
prefixing it with "pd:". You can force a variable to be always
treated as a "_meta" field by prefixing it with "field:" Therefore
the above expression could also be written as
'$_field:event_category$_=if($pd:_raw$ LIKE "WARN %", "warning",
"normal")'
* When writing to a _meta field, the default behavior is to add a new
index-time field even if one exists with the same name, the same way
WRITE_META works for regular-expression-based extractions. For
example, "a=5,
a=a+2" adds two index-time fields to _meta: "a::5 a::7". You can
change this
by using ":=" after the variable name. For example, setting "a=5,
a:=a+2"
causes Splunk software to add a single "a::7" field.
* NOTE: Replacing index-time fields is slower than adding them. It is
best to
only use ":=" when you need this behavior.
* The ":=" operator can also be used to remove existing fields in _meta
by assigning the expression null() to them.
* When reading from an index-time field that occurs multiple times
inside the
_meta key, normally the first value is used. You can override this by
prefixing the name with "mv:" which returns all of the values into a
"multival" object. For example, if _meta contains the keys "v::a v::b"
then
'mvjoin(v,"")' returns "a" while 'mvjoin($mv:v$","",")' returns "a,b".
* Note that this "mv:" prefix does not change behavior when it writes to
a
_meta field. If the value returned by an expression is a multivalue,
it
always creates multiple index-time fields. For example,
'x=mvappend("a","b","c")' causes the string "x::a x::b x::c" to be appended
to the _meta key.
* Internally, the _meta key can hold values with various numeric types.
Splunk software normally picks a type appropriate for the value that the
expression returned. However, you can override this this choice by
specifying
a type in square brackets after the destination field name. For
example,
"my_len[int]=length(source)" creates a new field named "my_len" and
forces it
to be stored as a 64-bit integer inside _meta. You can force Splunk
software
to store a number as floating point by using the type "[float]". You can
request a smaller, less-precise encoding by using "[float32]". If you want to
store the value as floating point but also ensure that the Splunk software
remembers the significant-figures information that the evaluation expression
deduced, use "[float-sf]" or "[float32-sf]". Finally, you can force the
result to be treated as a string by specifying "[string]".
* The capability of the search-time |eval operator to name the destination
  field based on the value of another field (like "| eval {destname}=1")
is NOT available for index-time evaluations.
* Default: empty

DELIMS = <quoted string list>
* NOTE: This setting is only valid for search-time field extractions.
* IMPORTANT: If a value may contain an embedded unescaped double quote character, such as "foo"bar", use REGEX, not DELIMS. An escaped double quote (\") is ok. Non-ASCII delimiters also require the use of REGEX.
* Optional. Use DELIMS in place of REGEX when you are working with ASCII-only delimiter-based field extractions, where field values (or field/value pairs) are separated by delimiters such as colons, spaces, line breaks, and so on.
* Sets delimiter characters, first to separate data into field/value pairs,
  and then to separate field from value.
* Each individual ASCII character in the delimiter string is used as a delimiter to split the event.
* Delimiters must be specified within double quotes (eg. DELIMS="|,;"). Special escape sequences are \t (tab), \n (newline), \r (carriage return), \\ (backslash) and \" (double quotes).
* When the event contains full delimiter-separated field/value pairs, you enter two sets of quoted characters for DELIMS:
  * The first set of quoted delimiters extracts the field/value pairs.
  * The second set of quoted delimiters separates the field name from its corresponding value.
  * When the event only contains delimiter-separated values (no field names), use just one set of quoted delimiters to separate the field values. Then use the FIELDS setting to apply field names to the extracted values.
  * Alternately, Splunk software reads even tokens as field names and odd tokens as field values.
* Splunk software consumes consecutive delimiter characters unless you specify a list of field names.
* The following example of DELIMS usage applies to an event where field/value pairs are separated by '|' symbols and the field names are separated from their corresponding values by '=' symbols:

```plaintext
[pipe_eq]
DELIMS = "|", "="
```
* Default: ""

FIELDS = <quoted string list>
* NOTE: This setting is only valid for search-time field extractions.
* Used in conjunction with DELIMS when you are performing delimiter-based field extraction and only have field values to extract.
* FIELDS enables you to provide field names for the extracted field values, in list format according to the order in which the values are extracted.
* NOTE: If field names contain spaces or commas they must be quoted with " "

To escape, use \\
* The following example is a delimiter-based field extraction where three field values appear in an event. They are separated by a comma and then a space.

```plaintext
[commalist]
DELIMS = ", ,"
FIELDS = field1, field2, field3
```
* Default: ""

MV_ADD = [true|false]
* NOTE: This setting is only valid for search-time field extractions.
* Optional. Controls what the extractor does when it finds a field which already exists.
* If set to true, the extractor makes the field a multivalued field and appends the newly found value, otherwise the newly found value is discarded.
* Default: false

CLEAN_KEYS = [true|false]
* NOTE: This setting is only valid for search-time field extractions.
* Optional. Controls whether Splunk software "cleans" the keys (field names) it extracts at search time. "Key cleaning" is the practice of replacing any non-alphanumeric characters (characters other than those falling between the a-z, A-Z, or 0-9 ranges) in field names with underscores, as well as the stripping of leading underscores and 0-9 characters from field names.
* Add CLEAN_KEYS = false to your transform if you need to extract field
names that include non-alphanumeric characters, or which begin with
underscores or 0-9 characters.
* Default: true

KEEP_EMPTY_VALS = [true|false]
* NOTE: This setting is only valid for search-time field extractions.
* Optional. Controls whether Splunk software keeps field/value pairs when
  the value is an empty string.
* This option does not apply to field/value pairs that are generated by
  Splunk software autokv extraction. Autokv ignores field/value pairs with
  empty values.
* Default: false

CAN_OPTIMIZE = [true|false]
* NOTE: This setting is only valid for search-time field extractions.
* Optional. Controls whether Splunk software can optimize this extraction out
  (another way of saying the extraction is disabled).
* You might use this if you are running searches under a Search Mode setting
  that disables field discovery--it ensures that Software always discovers
  specific fields.
* Splunk software only disables an extraction if it can determine that none of
  the fields identified by the extraction will ever be needed for the successful
  evaluation of a search.
* NOTE: This option should be rarely set to false.
* Default: true

### Lookup tables

# NOTE: Lookup tables are used ONLY during search time

filename = <string>
* Name of static lookup file.
* File should be in $SPLUNK_HOME/etc/system/lookups/, or in
  $SPLUNK_HOME/etc/<app_name>/lookups/ if the lookup belongs to a
  specific app.
* If file is in multiple 'lookups' directories, no layering is done.
* Standard conf file precedence is used to disambiguate.
* Only file names are supported. Paths are explicitly not supported. If
  you specify a path, Splunk software strips the path to use the value after
  the final path separator.
* Splunk software then looks for this filename in
collection = <string>
* Name of the collection to use for this lookup.
* Collection should be defined in
$SPLUNK_HOME/etc/<app_name>/collections.conf
  for some <app_name>
* If collection is in multiple collections.conf file, no layering is done.
* Standard conf file precedence is used to disambiguate.
* Defaults to empty string (in which case the name of the stanza is used).

max_matches = <integer>
* The maximum number of possible matches for each input lookup value (range 1 - 1000).
* If the lookup is non-temporal (not time-bounded, meaning the time_field setting is not specified), Splunk software uses the first <integer> entries, in file order.
* If the lookup is temporal, Splunk software uses the first <integer> entries in descending time order. In other words, only <max_matches> lookup entries are allowed to match. If the number of lookup entries exceeds <max_matches>, only the ones nearest to the lookup value are used.
* Default = 1000 if the lookup is not temporal, default = 1 if it is temporal.

min_matches = <integer>
* Minimum number of possible matches for each input lookup value.
* Default = 0 for both temporal and non-temporal lookups, which means that Splunk software outputs nothing if it cannot find any matches.
* However, if min_matches > 0, and Splunk software gets less than min_matches, it provides the default_match value provided (see below).

default_match = <string>
* If min_matches > 0 and Splunk software has less than min_matches for any given input, it provides this default_match value one or more times until the min_matches threshold is reached.
* Defaults to empty string.

case_sensitive_match = <bool>
* NOTE: To disable case-sensitive matching with input fields and values

535
from events, the KV Store lookup data must be entirely in lower case. The input data can be of any case, but the KV Store data must be lower case.
* If set to false, case insensitive matching is performed for all fields in a lookup table
* Defaults to true (case sensitive matching)

**match_type = <string>**
* A comma and space-delimited list of <match_type>(<field_name>) specification to allow for non-exact matching
* The available match_type values are WILDCARD, CIDR, and EXACT. Only fields that should use WILDCARD or CIDR matching should be specified in this list.
* Default: EXACT

**external_cmd = <string>**
* Provides the command and arguments to invoke to perform a lookup. Use this for external (or "scripted") lookups, where you interface with with an external script rather than a lookup table.
* This string is parsed like a shell command.
* The first argument is expected to be a python script (or executable file) located in $SPLUNK_HOME/etc/<app_name>/bin (or ../etc/searchscripts).
* Presence of this field indicates that the lookup is external and command based.
* Default: empty string

**fields_list = <string>**
* A comma- and space-delimited list of all fields that are supported by the external command.

**index_fields_list = <string>**
* A comma- and space-delimited list of fields that need to be indexed for a static .csv lookup file.
* The other fields are not indexed and not searchable.
* Restricting the fields enables better lookup performance.
* Defaults to all fields that are defined in the .csv lookup file header.

**external_type = [python|executable|kvstore|geo]**
* This setting describes the external lookup type.
* Use 'python' for external lookups that use a python script.
* Use 'executable' for external lookups that use a binary executable, such as a C++ executable.
* Use 'kvstore' for KV store lookups.
* Use 'geo' for geospatial lookups.
  * Default: python

`time_field = <string>`
* Used for temporal (time bounded) lookups. Specifies the name of the field in the lookup table that represents the timestamp.
  * Default: empty string
  * This means that lookups are not temporal by default.

`time_format = <string>`
* For temporal lookups this specifies the 'strptime' format of the timestamp field.
  * You can include subseconds but Splunk software ignores them.
  * Default: %s.%Q (seconds from unix epoch in UTC and optional milliseconds)

`max_offset_secs = <integer>`
* For temporal lookups, this is the maximum time (in seconds) that the event timestamp can be later than the lookup entry time for a match to occur.
  * Default: 2000000000

`min_offset_secs = <integer>`
* For temporal lookups, this is the minimum time (in seconds) that the event timestamp can be later than the lookup entry timestamp for a match to occur.
  * Default: 0

`batch_index_query = <bool>`
* For large file-based lookups, batch_index_query determines whether queries can be grouped to improve search performance.
  * Default is unspecified here, but defaults to true (at global level in `limits.conf`)

`allow_caching = <bool>`
* Allow output from lookup scripts to be cached
  * Default: true

`cache_size = <integer>`
* Cache size to be used for a particular lookup. If a previously looked up value is already present in the cache, it is applied.
  * The cache size represents the number of input values for which to cache output values from a lookup table.
  * Do not change this value unless you are advised to do so by Splunk Support or
a similar authority.
* Default: 10000

max_ext_batch = <integer>
* The maximum size of external batch (range 1 - 1000).
* This setting applies only to KV Store lookup configurations.
* Default: 300

filter = <string>
* Filter results from the lookup table before returning data. Create this filter like you would a typical search query using Boolean expressions and/or comparison operators.
* For KV Store lookups, filtering is done when data is initially retrieved to improve performance.
* For CSV lookups, filtering is done in memory.

feature_id_element = <string>
* If the lookup file is a kmz file, this field can be used to specify the xml path from placemark down to the name of this placemark.
* This setting applies only to geospatial lookup configurations.
* Default: /Placemark/name

check_permission = <bool>
* Specifies whether the system can verify that a user has write permission to a lookup file when that user uses the outputlookup command to modify that file.
  If the user does not have write permissions, the system prevents the modification.
* The check_permission setting is only respected when output_check_permission is set to "true" in limits.conf.
* You can set lookup table file permissions in the .meta file for each lookup file, or through the Lookup Table Files page in Settings. By default, only users who have the admin or power role can write to a shared CSV lookup file.
* This setting applies only to CSV lookup configurations.
* Default: false

replicate = true|false
* Indicates whether to replicate CSV lookups to indexers.
* When false, the CSV lookup is replicated only to search heads in a search head cluster so that input lookup commands can use this lookup on the search heads.
* When true, the CSV lookup is replicated to both indexers and search
heads.
* Only for CSV lookup files.
* Note that replicate=true works only if it is included in replication
  whitelist. See distSearch.conf/[replicationWhitelist] option.
* Default: true

**METRICS - STATSD DIMENSION EXTRACTION**

**Metrics**

[statsd-dims:<unique_transforms_stanza_name>]
* 'statsd-dims' prefix indicates this stanza is applicable only to
  statsd metric
  type input data.
* This stanza is used to define regular expression to match and extract
  dimensions out of statsd dotted name segments.
* By default, only the unmatched segments of the statsd dotted name
  segment
  become the metric_name.

REGEX = <regular expression>
* Splunk software supports a named capturing group extraction format to
  provide
  dimension names of the corresponding values being extracted out. For
  example:
    (?<dim1>group)(?<dim2>group)..

REMOVE_DIMS_FROM_METRIC_NAME = <boolean>
* If set to false, the matched dimension values from the REGEX above
  would also
  be a part of the metric name.
* If true, the matched dimension values would not be a part of metric
  name.
* Default: true

[metric-schema:<unique_transforms_stanza_name>]
* The 'metric-schema' stanza transforms index-time field extractions
  from a
  single log event into metrics.
* Each metric created has its own metric_name and _value.
* The other fields extracted from the log event become dimensions in the
  generated metrics.
* You must provide one of the following two settings:
  METRIC-SCHEMA-MEASURES-<unique_metric_name_prefix> or
METRIC-SCHEMA-MEASURES. These settings determine how values for the metric_name and _value fields are obtained.

METRIC-SCHEMA-MEASURES-<unique_metric_name_prefix> = <measure_field1>, <measure_field2>, ...
* Optional.
* <unique_metric_name_prefix> should match the value of a field extracted from the event.
* <measure_field> should match the name of a field with a numeric value extracted from the event.
* If the value of the 'metric_name' index-time extraction matches with the <unique_metric_name_prefix>, the Splunk platform:
  * Creates a metric with a new metric_name for each <measure_field> where the metric_name value is the <measure_field> prefixed by the <unique_metric_name_prefix>.
  * Saves the corresponding numeric value for each <measure_field> as '_value' within each metric.
  * The Splunk platform saves the remaining index-time field extractions as dimensions in each of the created metrics.
* Default: empty

METRIC-SCHEMA-BLACKLIST-DIMS-<unique_metric_name_prefix> = <dimension_field1>, <dimension_field2>, ...
* Optional
* This configuration enables the Splunk platform to omit unnecessary dimensions when it transforms log data to metrics data. You might set this up if you have high-cardinality dimensions that are unnecessary for your metrics.
* Use this configuration in conjunction with a corresponding METRIC-SCHEMA-MEASURES-<unique_metric_name_prefix> configuration.
* <unique_metric_name_prefix> should match the value of a field extracted from the log event.
* <dimension_field> should match the name of a field in the log event that is not extracted as a <measure_field> in the corresponding METRIC-SCHEMA-MEASURES-<unique_metric_name_prefix> configuration.
* Default: empty

METRIC-SCHEMA-MEASURES = <measure_field1>, <measure_field2>, ...
* Optional.
* This configuration has a lower precedence over METRIC-SCHEMA-MEASURES-<unique_metric_name_prefix> if event has a match for unique_metric_name_prefix
* When no prefix can be identified, this configuration is active
to create a new metric for each `<measure_field>` in the event data.
* The Splunk platform saves the remaining index-time field extractions as
dimensions in each of the created metrics.
* Default: empty

```
METRIC-SCHEMA-BLACKLIST-DIMS = `<dimension_field1>,
<dimension_field2>,...
```
* Optional
* This configuration enables the Splunk platform to omit unnecessary
dimensions
when it transforms log data to metrics data. You might set this up if you
have high-cardinality dimensions that are unnecessary for your metrics.
* Use this configuration in conjunction with a corresponding
`METRIC-SCHEMA-MEASURES` configuration.
* `<dimension_field>` should match the name of a field in the log event
that is
not extracted as a `<measure_field>` in the corresponding `METRIC-SCHEMA-
MEASURES` configuration.
* Default: empty

**KEYS:**

* NOTE: Keys are case-sensitive. Use the following keys exactly as they
appear.

```
queue : Specify which queue to send the event to (can be nullQueue,
indexQueue).
   * indexQueue is the usual destination for events going through the
   transform-handling processor.
   * nullQueue is a destination which causes the events to be
   dropped entirely.
_meta : The raw text of the event.
_time : The timestamp of the event, in seconds since 1/1/1970 UTC.
```

```
MetaData:Host       : The host associated with the event.
   The value must be prefixed by "host::"
_METAData:Index     : The index where the event should be stored.
MetaData:Source     : The source associated with the event.
   The value must be prefixed by "source::"
MetaData:SourceType : The source type of the event.
```
The value must be prefixed by "sourcetype::"

_TCP_ROUTING : Comma separated list of tcpout group names (from outputs.conf)
  Defaults to groups present in 'defaultGroup' for [tcpout].
SYSLOG_ROUTING : Comma separated list of syslog-stanza names (from outputs.conf)
  Defaults to groups present in 'defaultGroup' for [syslog].

* NOTE: Any KEY (field name) prefixed by '_' is not indexed by Splunk software, in general.

[accepted_keys]

<name> = <key>

* Modifies the list of valid SOURCE_KEY and DEST_KEY values. Splunk software checks the SOURCE_KEY and DEST_KEY values in your transforms against this list when it performs index-time field transformations.
* Add entries to [accepted_keys] to provide valid keys for specific environments, apps, or similar domains.
* The 'name' element disambiguates entries, similar to -class entries in props.conf.
* The 'name' element can be anything you choose, including a description of the purpose of the key.
* The entire stanza defaults to not being present, causing all keys not documented just above to be flagged.

transforms.conf.example

# Version 7.2.0
# This is an example transforms.conf. Use this file to create regexes and rules for transforms. Use this file in tandem with props.conf.
# To use one or more of these configurations, copy the configuration block into transforms.conf in $SPLUNK_HOME/etc/system/local/. You must restart Splunk to enable configurations.
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

# Note: These are examples. Replace the values with your own customizations.

# Indexed field:

[netscreen-error]
REGEX = device_id=\[w+\](?<err_code>[^:]+)
FORMAT = err_code::$1
WRITE_META = true

# Override host:

[hostoverride]
DEST_KEY = MetaData:Host
REGEX = \s(\w*)$
FORMAT = host::$1

# Extracted fields:

[netscreen-error-field]
REGEX = device_id=\[w+\](?<err_code>[^:]+)
FORMAT = err_code::$1

# Index-time evaluations:

[discard-long-lines]
INGEST_EVAL = queue=if(length(_raw) > 500, "nullQueue", "")

[split-into-sixteen-indexes-for-no-good-reason]
INGEST_EVAL = index="split_" . substr(md5(_raw),1,1)

[add-two-numeric-fields]
INGEST_EVAL = loglen_raw=ln(length(_raw)),
loglen_src=ln(length(source))

# In this example we only create the new index-time field if the host 
# had a dot in it; assigning null() to a new field is a no-op:

[add-hostdomain-field]
INGEST_EVAL = hostdomain=if(host LIKE "%.%",
replace(host,"^[^\./]+\./",""), null())

# Static lookup table

543
[mylookuptable]
filename = mytable.csv

# one to one lookup
# guarantees that we output a single lookup value for each input value,
if
# no match exists, we use the value of "default_match", which by default
is
# "NONE"
[mylook]
filename = mytable.csv
max_matches = 1
min_matches = 1
default_match = nothing

# Lookup and filter results
[myfilteredlookup]
filename = mytable.csv
filter = id<500 AND color="red"

# external command lookup table
[myexternaltable]
external_cmd = testadapter.py blah
fields_list = foo bar

# Temporal based static lookup table
[staticwtime]
filename = mytable.csv
time_field = timestamp
time_format = %d/%m/%y %H:%M:%S

# Mask sensitive data:
[session-anonymizer]
REGEX = (\w+)SessionId=\w+(\w{4}[&"].*)_S
FORMAT = $1SessionId=###$2
DEST_KEY = _raw

# Route to an alternate index:
[AppRedirect]
REGEX = Application
DEST_KEY = _MetaData:Index
FORMAT = Verbose

# Extract comma-delimited values into fields:
[extract_csv]
DELIMS = ",,"
FIELDS = "field1", "field2", "field3"

# This example assigns the extracted values from _raw to field1, field2 and
# field3 (in order of extraction). If more than three values are
# extracted
# the values without a matching field name are ignored.

[pipe_eq]
DELIMS = "|", "="

# The above example extracts key-value pairs which are separated by '|' while the key is delimited from value by '='.

[multiple_delims]
DELIMS = "|;", "=:"

# The above example extracts key-value pairs which are separated by '|' or
# ';', while the key is delimited from value by '=' or ':'.

###### BASIC MODULAR REGULAR EXPRESSIONS DEFINITION START ###########
# When adding a new basic modular regex PLEASE add a comment that lists
# the fields that it extracts (named capturing groups), or whether it
# provides a placeholder for the group name as:
# Extracts: field1, field2....
#
[all_lazy]
REGEX = .*?

[all]
REGEX = .*

[nspaces]
# matches one or more NON space characters
REGEX = \S+

[alphas]
# matches a string containing only letters a-zA-Z
REGEX = [a-zA-Z]+

[alnums]
# matches a string containing letters + digits
REGEX = [a-zA-Z0-9]+

[qstring]
# matches a quoted "string" - extracts an unnamed variable
# name MUST be provided as in [[qstring:name]]
# Extracts: empty-name-group (needs name)
REGEX = "(?<>[^"]*+)

[sbstring]
# matches a string enclosed in [] - extracts an unnamed variable
# name MUST be provided as in [[sbstring:name]]
# Extracts: empty-name-group (needs name)
REGEX = \[(?<>[^\]*+)\]

[digits]
REGEX = \d+

[int]
# matches an integer or a hex number
REGEX = 0x[a-fA-F0-9]+|\d+

[float]
# matches a float (or an int)
REGEX = \d*\.\d+|[int]

[octet]
# this would match only numbers from 0-255 (one octet in an ip)
REGEX = (?:2(?::?[0-5]|\[0-4][0-9]|\[0-1][0-9]|0-9)|[0-9][0-9]?)

[ipv4]
# matches a valid IPv4 optionally followed by :port_num the octets in the ip
# would also be validated 0-255 range
# Extracts: ip, port
REGEX = (?<ip> [[octet]] (?:\.[[octet]]{3})?\[int:port\])

[simple_url]
# matches a url of the form proto://domain.tld/uri
# Extracts: url, domain
REGEX = (?<url>\w++://(?<domain>[a-zA-Z0-9\-.:]++)(?:/[^s]*)?)

[url]
# matches a url of the form proto://domain.tld/uri
# Extracts: url, proto, domain, uri
REGEX = (?<url>[[alphas:proto]]://(?<domain>[a-zA-Z0-9\-.:]++)(?<uri>[/\s]*)?)

[simple_uri]
# matches a uri of the form /path/to/resource?query
# Extracts: uri, uri_path, uri_query
REGEX = (?<uri>(?<uri_path>[^\s]*)++)(?:\?[<uri_query>\[^\s"]]+)

[url]
# uri = path optionally followed by query
[/this/path/file.js?query=part&other=var]
# path = root part followed by file        [/root/part/file.part]
# Extracts: uri, uri_path, uri_root, uri_file, uri_query, uri_domain
# (optional if in proxy mode)
REGEX = (?<uri>(?:\w++://(?<uri_domain>[^/\s]++))?(?<uri_path>(?<uri_root>/+(?:[^\s?;=/\]*+/+)*)(?<uri_file>[^\s?;=?/\]*+))(?:\?(?<uri_query>[^\s"]+))?)

[hide-ip-address]
# Make a clone of an event with the sourcetype masked_ip_address. The
# clone
# will be modified; its text changed to mask the ip address.
# The cloned event will be further processed by index-time transforms
# and
# SEDCMD expressions according to its new sourcetype.
# In most scenarios an additional transform would be used to direct the
# masked_ip_address event to a different index than the original data.
REGEX = ^(.*?)src=\d+\.\d+\.\d+\.\d+(.*)$
FORMAT = $1src=XXXXX$2
DEST_KEY = _raw
CLONE_SOURCETYPE = masked_ip_addresses

#Set REPEAT_MATCH to true to repeatedly match the regex in the data.
#example sample data - 1483382050 a=1 b=2 c=3 d=4 e=5
#Since REPEAT_MATCH is set to true, the regex will matched for a=1, then
#b=2, then c=3 and so on
#If REPEAT_MATCH is not set, the match will stop at a=1
#Since WRITE_META is set to true, these will added as indexed fields -
a, b, c, d, e
[repeat_regex]
REGEX = ([a-z])=(\d+)$
FORMAT = $1::$2
REPEAT_MATCH = true
WRITE_META = true

REGEX = #([0-9]+) \+\((-?[0-9]+)\)\ \([X]\)
FORMAT = $0 bash_quote_id::$1 bash_quote_score::$2
WRITE_META=true

###### BASIC MODULAR REGULAR EXPRESSIONS DEFINITION END ###############

# Statsd dimensions extraction

# For example, below two stanzas would extract dimensions as
# ipv4=10.2.3.4
# and os=windows from statsd
data=mem.percent.used.10.2.3.4.windows:33|g
[statsd-dims:regex_stanza1]
REGEX = (?<ipv4>\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3})
REMOVE_DIMS_FROM_METRIC_NAME = true

[statsd-dims:regex_stanza2]
REGEX = \S+.(?<os>\w+): REMOVE_DIMS_FROM_METRIC_NAME = true

# In most cases we need only one regex to be run per sourcetype. By default
# Splunk would look for the sourcetype name in transforms.conf in such scenario.
# Hence, there is no need to provide STATSD-DIM-TRANSFORMS setting in props.conf.
[statsd-dims:metric_sourcetype_name]
# In this example, we extract both ipv4 and os dimension using a single regex
REGEX = (?<ipv4>\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3})\.(?<os>\w+): REMOVE_DIMS_FROM_METRIC_NAME = true

# In this metrics example, we start with this log line:
# 01-26-2018 07:49:49.030 -0800 INFO  Metrics - group=queue, name=aggqueue, max_size_kb=1024, current_size_kb=1,
# current_size=3, largest_size=49, smallest_size=0, dc_latitude=37.3187706, dc_longitude=-121.9515042
# The following stanza converts that single event into multiple metrics at
# index-time. It blacklists the dc_latitude and dc_longitude dimensions, which
# means they are omitted from the generated metrics.
[metric-schema:logtometrics]
METRIC-SCHEMA-MEASURES-queue =
max_size_kb, current_size_kb, current_size, largest_size, smallest_size
METRIC-SCHEMA-BLACKLIST-DIMS-queue = dc_latitude, dc_longitude
# Here are the metrics generated by that stanza:
# {'metric_name' : 'queue.max_size_kb', 'value' : 1024, 'name': 'aggqueue'},
# {'metric_name' : 'queue.current_size_kb', 'value' : 1, 'name': 'aggqueue'},
# {'metric_name' : 'queue.current_size', 'value' : 3, 'name': 'aggqueue'},
# {'metric_name' : 'queue.largest_size', 'value' : 49, 'name': 'aggqueue'},
# {'metric_name' : 'queue.smallest_size', 'value' : 0, 'name': 'aggqueue'}

# In the sample log above, group=queue represents the unique metric name prefix. Hence, it needs to be
# formatted and saved as metric_name::queue for Splunk to identify queue as a metric name prefix.
ui-tour.conf

The following are the spec and example files for ui-tour.conf.

ui-tour.conf.spec

#  Version 7.2.0
#
# This file contains the tours available for Splunk Onboarding
#
GLOBAL SETTINGS

# Use the [default] stanza to define any global settings.
# * You can also define global settings outside of any stanza, at the
top of
#   the file.
# * This is not a typical conf file for configurations. It is used to
set/create
#   tours to demonstrate product functionality to users.
# * If an attribute is defined at both the global level and in a
specific
#   stanza, the value in the specific stanza takes precedence.

[<stanza name>]
* Stanza name is the name of the tour

useTour = <string>
* Used to redirect this tour to another when called by Splunk.
* Optional

nextTour = <string>
* String used to determine what tour to start when current tour is
finished.
* Optional

intro = <string>
* A custom string used in a modal to describe what tour is about to be
taken.
* Optional

type = <image || interactive>
* Can either be "image" or "interactive" to determine what kind of tour
it is.
* Required

label = <string>
* The identifying name for this tour used in the tour creation app.
* Optional in general
* Required only if this tour is being to linked from another tour (nextTour)

tourPage = <string>
* The Splunk view this tour is associated with (only necessary if it is linked to).
* Optional

managerPage = <boolean>
* Used to signify that the tourPage is a manager page. This will change the url of
* when the tourPage is rendered to "/manager/{app}/{view}" rather than "/app/{app}/{view}" 
* Optional

viewed = <boolean>
* A boolean to determine if this tour has been viewed by a user.
* Set by Splunk

skipText = <string>
* The string for the skip button (interactive and image)
* Defaults to "Skip tour"
* Optional

doneText = <string>
* The string for the button at the end of a tour (interactive and image)
* Defaults to "Try it now"
* Optional

doneURL = <string>
* The Splunk URL of where the user will be directed once the tour is over.
* The user will click a link/button.
* Helpful to use with above doneText attribute to specify location.
* Splunk link is formed after the localization portion of the full URL.
For example if the link
* is localhost:8000/en-US/app/search/reports, the doneURL will be
"app/search/reports"
* Optional

forceTour = <boolean>
* Used with auto tours to force users to take the tour and not be able to skip 
* Optional

For image based tours

# Users can list as many images with captions as they want. Each new
image is created by
# incrementing the number.

imageName<int> = <string>
* The name of the image file (example.png)
* Required but Optional only after first is set

imageCaption<int> = <string>
* The caption string for corresponding image
* Optional

imgPath = <string>
* The subdirectory relative to Splunk's 'img' directory in which users put the images.
  This will be appended to the url for image access and not make a server request within Splunk.
  EX) If user puts images in a subdirectory 'foo': imgPath = foo.
  EX) If within an app, imgPath = foo will point to the app's img path of
    appserver/static/img/foo
* Required only if images are not in the main 'img' directory.

category = <system || <specific app name>>
* String consisting of either 'system' or the app name the tour images are to be stored.
* If set to 'system', it will revert to Splunk's native img path.
* Required

**For interactive tours**

# Users can list as many steps with captions as they want. Each new step is created by
# incrementing the number.

urlData = <string>
* String of any querystring variables used with tourPage to create full url executing this tour.
* Don't add the "?" to the beginning of this string
* Optional

stepText<int> = <string>
* The string used in specified step to describe the UI being showcased.
* Required but Optional only after first is set

stepElement<int> = <selector>
* The UI Selector used for highlighting the DOM element for corresponding step.
* Optional
stepPosition<int> = <bottom || right || left || top>
* String that sets the position of the tooltip for corresponding step.
* Optional

stepClickEvent<int> = <click || mousedown || mouseup>
* Sets a specific click event for an element for corresponding step.
* Optional

stepClickElement<int> = <string>
* The UI selector used for a DOM element used in conjunction with click above.
* Optional

ui-tour.conf.example

#   Version 7.2.0
#
# This file contains the tours available for Splunk Onboarding
#
# To update tours, copy the configuration block into
# ui-tour.conf in $SPLUNK_HOME/etc/system/local/. Restart the Splunk software to
# see the changes.
#
# To learn more about configuration files (including precedence) see the
# documentation located at
#
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
#
# Image Tour
[tour-name]
type = image
imageName1 = TourStep1.png
imageCaption1 = This is the first caption
imageName2 = TourStep2.png
imageCaption2 = This is the second caption
imgPath = /testtour
context = system
doneText = Continue to Tour Page
doneURL = app/toursapp/home

# Interactive Tour
[test-interactive-tour]
type = interactive
tourPage = reports
urlData = data=foo&moredata=bar
label = Interactive Tour Test
visualizations.conf

The following are the spec and example files for visualizations.conf.

visualizations.conf.spec

```text
#   Version 7.2.0
#
# This file contains definitions for visualizations an app makes available
# to the system. An app intending to share visualizations with the system
# should include a visualizations.conf in
#$SPLUNK_HOME/etc/apps/<appname>/default
#
# visualizations.conf should include one stanza for each visualization to be shared
#
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles
#
#******
# The possible attribute/value pairs for visualizations.conf are:
#******

[<stanza name>]

* Create a unique stanza name for each visualization. It should match the name
  of the visualization
* Follow the stanza name with any number of the following attribute/value
  pairs.
```

554
* If you do not specify an attribute, Splunk uses the default.

disabled = <bool>
  * Optional.
  * Disable the visualization by setting to true.
  * If set to true, the visualization is not available anywhere in Splunk
  * Defaults to false.

allow_user_selection = <bool>
  * Optional.
  * Whether the visualization should be available for users to select
  * Defaults to true

label = <string>
  * Required.
  * The human-readable label or title of the visualization
  * Will be used in dropdowns and lists as the name of the visualization
  * Defaults to <app_name>.<viz_name>

description = <string>
  * Required.
  * The short description that will show up in the visualization picker
  * Defaults to ""

search_fragment = <string>
  * Required.
  * An example part of a search that formats the data correctly for the
  * viz. Typically the last pipe(s) in a search query.
  * Defaults to ""

default_height = <int>
  * Optional.
  * The default height of the visualization in pixels
  * Defaults to 250

default_width = <int>
  * Optional.
  * The default width of the visualization in pixels
  * Defaults to 250

min_height = <int>
  * Optional.
  * The minimum height the visualizations can be rendered in.
  * Defaults to 50.

min_width = <int>
  * Optional.
  * The minimum width the visualizations can be rendered in.
  * Defaults to 50.

max_height = <int>
  * The maximum height the visualizations supports.
* Optional.
* Default is unbounded.

max_width = <int>
* The maximum width the visualizations supports.
* Optional.
* Default is unbounded.

trellis_default_height = <int>
* Default is 400

trellis_min_widths = <string>
* Default is undefined

trellis_per_row = <string>
* Default is undefined

# Define data sources supported by the visualization and their initial fetch params for search results data

data_sources = <csv-list>
* Comma separated list of data source types supported by the visualization.
* Currently the visualization system provides these types of data sources:
  * - primary: Main data source driving the visualization.
  * - annotation: Additional data source for time series visualizations to show discrete event annotation on the time axis.
  * Defaults to "primary"

data_sources.<data-source-type>.params.output_mode = [json_rows|json_cols|json]
* Optional.
* the data format that the visualization expects. One of:
  * - "json_rows": corresponds to SplunkVisualizationBase.ROW_MAJOR_OUTPUT_MODE
  * - "json_cols": corresponds to SplunkVisualizationBase.COLUMN_MAJOR_OUTPUT_MODE
  * - "json": corresponds to SplunkVisualizationBase.RAW_OUTPUT_MODE
* Defaults to undefined and requires the javascript implementation to supply initial data params.

data_sources.<data-source-type>.params.count = <int>
* Optional.
* How many rows of results to request, default is 1000

data_sources.<data-source-type>.params.offset = <int>
* Optional.
* The index of the first requested result row, default is 0

data_sources.<data-source-type>.params.sort_key = <string>
* Optional.
* The field name to sort the results by

data_sources.<data-source-type>.params.sort_direction = [asc|desc]
  * Optional.
  * The direction of the sort
  * - asc: sort in ascending order
  * - desc: sort in descending order
  * Defaults to desc

data_sources.<data-source-type>.params.search = <string>
  * Optional.
  * A post-processing search to apply to generate the results

data_sources.<data-source-type>.mapping_filter = <bool>
data_sources.<data-source-type>.mapping_filter.center = <string>
data_sources.<data-source-type>.mapping_filter.zoom = <string>
supports_trellis = <bool>
  * Optional.
  * Indicates whether trellis layout is available for this visualization
  * Defaults to false

supports_drilldown = <bool>
  * Optional.
  * Indicates whether the visualization supports drilldown (responsive actions triggered when users click on the visualization).
  * Defaults to false

supports_export = <bool>
  * Optional.
  * Indicates whether the visualization supports being exported to PDF.
  * This setting has no effect in third party visualizations.
  * Defaults to false

# Internal settings for bundled visualizations. They are ignored for third party visualizations.
core.type = <string>
core.viz_type = <string>
core.charting_type = <string>
core.mapping_type = <string>
core.order = <int>
core.icon = <string>
core.preview_image = <string>
core.recommend_for = <string>
core.height_attribute = <string>
visualizations.conf.example

No example

web.conf

The following are the spec and example files for web.conf.

web.conf.spec

#   Version 7.2.0
#
# This file contains possible attributes and values you can use to
# configure
# the Splunk Web interface.
#
# There is a web.conf in $SPLUNK_HOME/etc/system/default/. To set
# custom
# configurations, place a web.conf in $SPLUNK_HOME/etc/system/local/.
# For
# examples, see web.conf.example. You must restart Splunk software to
# enable
# configurations.
#
# To learn more about configuration files (including precedence) please
# see
# the documentation located at
#
http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

[settings]
* Set general Splunk Web configuration options under this stanza name.
* Follow this stanza name with any number of the following setting/value
  pairs.
* If you do not specify an entry for each setting, Splunk Web uses the
  default value.

startwebserver = [0 | 1]
* Set whether or not to start Splunk Web.
* 0 disables Splunk Web, 1 enables it.
* Default: 1

httpport = <positive integer>
* The TCP port on which Splunk Web listens for incoming connections.
* Must be present for Splunk Web to start.
* If omitted or 0 the server will NOT start an http listener.
* If using SSL, set to the HTTPS port number.
* Default: 8000

mgmtHostPort = <IP address:port>
* The IP address and host port of the splunkd process.
* Don't include "http[s]://" when specifying this setting. Only include the IP address and port.
* Default: 0.0.0.0:8089

appServerPorts = <positive integer>, <positive integer>, <positive integer> ...
* Port number(s) for the python-based application server to listen on. This port is bound only on the loopback interface -- it is not exposed to the network at large.
* If set to "0", prevents the application server from being run from splunkd. Instead, Splunk Web starts as a separate python-based service which directly listens to the 'httpport'. This is how Splunk 6.1.X and earlier behaved.
* Generally, you should only set one port number here. For most deployments a single application server won't be a performance bottleneck. However you can provide a comma-separated list of port numbers here and splunkd will start a load-balanced application server on each one.
* Set this setting to a non-zero value unless you experience compatibility problems. The new separate application server configuration is faster and supports more configuration options. Also, setting this to "0" can cause problems with new functionality, such as using the Search Head Clustering feature. See the [shclustering] stanza in server.conf for more information.
* Default: 8065

splunkdConnectionTimeout = <integer>
* The amount of time, in seconds, to wait before timing out when communicating with splunkd.
* Must be at least 30. If not
* Values smaller than 30 will be ignored, resulting in the use of the default value
* Default: 30

enableSplunkWebClientNetloc = <boolean>
* Control if the splunk web client can override the client network location
* Default: false

enableSplunkWebSSL = <boolean>
* Toggle between http or https.
* Set to true to enable https and SSL.
* Default: false
privKeyPath = <path>
* The path to the file containing the web server SSL certificate private
  key.
  * A relative path is interpreted relative to $SPLUNK_HOME and may not
  refer 
    outside of $SPLUNK_HOME (e.g., no ../*somewhere).
  * You can also specify an absolute path to an external key.
  * See also 'enableSplunkWebSSL' and 'serverCert'.
  * No default.

serverCert = <path>
* Full path to the Privacy Enhanced Mail (PEM) format Splunk web server
  certificate file.
* The file may also contain root and intermediate certificates, if
  required.
    They should be listed sequentially in the order:
      [ Server SSL certificate ]
      [ One or more intermediate certificates, if required ]
      [ Root certificate, if required ]
  * See also 'enableSplunkWebSSL' and 'privKeyPath'.
  * Default: $SPLUNK_HOME/etc/auth/splunkweb/cert.pem

sslPassword = <password>
* Password that protects the private key specified by 'privKeyPath'.
* If encrypted private key is used, do not enable client-authentication
  on splunkd server. In [sslConfig] stanza of server.conf,
  'requireClientCert' must be 'false'.
* Optional.
* Default: The unencrypted private key.

caCertPath = <path>
* DEPRECATED.
* Use 'serverCert' instead.
* A relative path is interpreted relative to $SPLUNK_HOME and may not
  refer
   outside of $SPLUNK_HOME (e.g., no ../*somewhere).
* No default.

requireClientCert = <boolean>
* Requires that any HTTPS client that connects to the Splunk Web HTTPS
  server has a certificate that was signed by the CA cert installed
  on this server.
* If "true", a client can connect ONLY if a certificate created by our
  certificate authority was used on that client.
* If "true", it is mandatory to configure splunkd with same root CA in
  server.conf.
  This is needed for internal communication between splunkd and
  splunkweb.
* Default: false

sslCommonNameToCheck = <commonName1>, <commonName2>, ...
* Checks the common name of the client's certificate against this list of names.  
  * 'requireClientCert' must be set to "true" for this setting to work.  
  * Optional.  
  * Default: empty string (No common name checking).

sslAltNameToCheck = <alternateName1>, <alternateName2>, ...  
* If this value is set, and 'requireClientCert' is set to true, Splunk Web will verify certificates which have a so-called "Subject Alternate Name" that matches any of the alternate names in this list.  
  * Subject Alternate Names are effectively extended descriptive fields in SSL certs beyond the commonName. A common practice for HTTPS certs is to use these values to store additional valid hostnames or domains where the cert should be considered valid.  
  * Accepts a comma-separated list of Subject Alternate Names to consider valid.  
  * Optional.  
  * Default: empty string (no alternate name checking).

serviceFormPostURL = http://docs.splunk.com/Documentation/Splunk  
* DEPRECATED.  
* This setting has been deprecated since Splunk Enterprise version 5.0.3.

userRegistrationURL = https://www.splunk.com/page/sign_up  
updateCheckerBaseURL = http://quickdraw.Splunk.com/js/  
docsCheckerBaseURL = http://quickdraw.splunk.com/help  
* These are various Splunk.com urls that are configurable.  
* Setting 'updateCheckerBaseURL' to 0 stops Splunk Web from pinging Splunk.com for new versions of Splunk software.

enable_insecure_login = <boolean>  
* Whether or not the GET-based "/account/insecurelogin" REST endpoint is enabled.  
  * Provides an alternate GET-based authentication mechanism.  
  * If "true", the following url is available:  
  * If "false", only the main /account/login endpoint is available  
  * Default: false

simple_error_page = <boolean>  
* Whether or not to display a simplified error page for HTTP errors that only contains the error status.  
  * If set to "true", Splunk Web displays a simplified error page for errors (404, 500, etc.) that only contain the error status.  
  * If set to "false", Splunk Web displays a more verbose error page that contains the home link, message, a more_results_link, crashes, referrer, debug output, and byline  
  * Default: false

login_content = <string>
* Lets you add custom content to the login page.
* Supports any text including HTML.
* No default.

sslVersions = <comma-separated list>
* A comma-separated list of SSL versions to support.
* The versions available are "ssl3", "tls1.0", "tls1.1", and "tls1.2"
* The special version "*" selects all supported versions. The version
  "tls"
    selects all versions tls1.0 or newer
* If you prefix a version with "-", it is removed from the list.
* SSLv2 is always disabled; "-ssl2" is accepted in the version list, but
  does nothing.
* When 'appServerPorts'="0" only supported values are "all", "ssl3,
  tls"  
  and "tls"
* When configured in FIPS mode, "ssl3" is always disabled regardless
  of this configuration.
* NOTE: this setting only takes effect when 'appServerPorts' is set to a
  non-zero value.
* For the default, see $$SPLUNK_HOME/etc/system/default/web.conf.$$

supportSSLV3Only = <boolean>
* When 'appServerPorts' is set to a non-zero value (the default mode),
  this setting is DEPRECATED. SSLv2 is now always disabled.
  The exact set of SSL versions allowed is now configurable via the
  'sslVersions' setting above.
* When 'appServerPorts' is set to 0, this controls whether SSLv2
  connections are disallowed.
* Default (when 'appServerPorts' is set to 0): false

cipherSuite = <cipher suite string>
* If set, uses the specified cipher string for the HTTP server.
* If not set, uses the default cipher string provided by OpenSSL. This
  is
    used to ensure that the server does not accept connections using weak
    encryption protocols.
* Must specify 'dhFile' to enable any Diffie-Hellman ciphers.
  The default can vary. See the cipherSuite setting in
  $$SPLUNK_HOME/etc/system/default/web.conf$$ for the current default.

ecdhCurveName = <string>
* DEPRECATED.
  Use the 'ecdhCurves' setting instead.
* This setting specifies the Elliptic Curve Diffie-Hellman (ECDH) curve
  to
  use for ECDH key negotiation.
* Splunk only supports named curves that have been specified by their
  SHORT name.
* The list of valid named curves by their short and long names
  can be obtained by running this CLI command:
  $$SPLUNK_HOME/bin/splunk cmd openssl ecparam -list_curves$$
* Default: empty string.

dhFile = <path>
* Full path to the Diffie-Hellman parameter file.
* Relative paths are interpreted as relative to $SPLUNK_HOME, and must
  not refer to a location outside of $SPLUNK_HOME.
* This file is required in order to enable any Diffie-Hellman ciphers.
* Default: not set.

root_endpoint = <URI_prefix_string>
* Defines the root URI path on which the appserver will listen
* For example, if you want to proxy the splunk UI at
  http://splunk:8000/splunkui,
  then set root_endpoint = /splunkui
* Default: /

static_endpoint = <URI_prefix_string>
* Path to static content.
* The path here is automatically appended to root_endpoint defined above
* Default: /static

static_dir = <relative_filesystem_path>
* The directory that holds the static content
* This can be an absolute URL if you want to put it elsewhere
* Default: share/splunk/search_mrsparkle/exposed

rss_endpoint = <URI_prefix_string>
* Path to static rss content
* The path here is automatically appended to what you defined in the
  'root_endpoint' setting
* Default: /rss

embed_uri = <URI>
* Optional URI scheme/host/port prefix for embedded content
* This presents an optional strategy for exposing embedded shared
  content that does not require authentication in a reverse proxy/single
  sign on environment.
* Default: empty string, resolves to the client
  
  `window.location.protocol + "//" + window.location.host`

```
embed_footer = <html_string>
```
* A block of HTML code that defines the footer for an embedded report.
* Any valid HTML code is acceptable.
* Default: "splunk>"

```
tools.staticdir.generate_indexes = [1 | 0]
```
* Whether or not the webserver serves a directory listing for static directories.
* Default: 0 (false)

```
template_dir = <relative_filesystem_path>
```
* The base path to the Mako templates.
* Default: "share/splunk/search_mrsparkle/templates"

```
module_dir = <relative_filesystem_path>
```
* The base path to Splunk Web module assets.
* Default: "share/splunk/search_mrsparkle/modules"

```
enable_gzip = <boolean>
```
* Whether or not the webserver applies gzip compression to responses.
* Default: true

```
use_future_expires = <boolean>
```
* Whether or not the Expires header of /static files is set to a far-future date
* Default: true

```
flash_major_version = <integer>
flash_minor_version = <integer>
flash_revision_version = <integer>
```
* Specifies the minimum Flash plugin version requirements
* Flash support, broken into three parts.
* We currently require a min baseline of Shockwave Flash 9.0 r124

```
override_JSON_MIME_type_with_text_plain = <boolean>
```
* Whether or not to override the MIME type for JSON data served up by Splunk Web endpoints with content-type="text/plain; charset=UTF-8"
* If "true", Splunk Web endpoints (other than proxy) that serve JSON data will serve as "text/plain; charset=UTF-8"
* If "false", Splunk Web endpoints that serve JSON data will serve as "application/json; charset=UTF-8"

```
enable_proxy_write = <boolean>
```
* Indicates if the /splunkd proxy endpoint allows POST operations.
* If "true", both GET and POST operations are proxied through to splunkd.
* If "false", only GET operations are proxied through to splunkd.
* Setting to "false" prevents many client-side packages (such as the
Splunk JavaScript SDK) from working correctly.
* Default: true

js_logger_mode = [None | Firebug | Server]
* The JavaScript Logger mode.
* Available modes: None, Firebug, Server
* Mode None: Does not log anything.
* Mode Firebug: Use firebug by default if it exists, or defer to the
  older
  less promiscuous version of firebug lite.
* Mode Server: Log to a defined server endpoint.
* See js/logger.js Splunk.Logger.Mode for mode implementation details
  and if
  you would like to author your own.
* Default: None

js_logger_mode_server_end_point = <URI_relative_path>
* The server endpoint to post JavaScript log messages
* Used when js_logger_mode = Server
* Default: util/log/js

js_logger_mode_server_poll_buffer = <integer>
* The interval, in milliseconds, to check, post, and cleanse the
  JavaScript log buffer
* Default: 1000

js_logger_mode_server_max_buffer = <integer>
* The maximum size threshold, in megabytes, to post and cleanse the
  JavaScript log buffer
* Default: 100

ui_inactivity_timeout = <integer>
* The length of time lapsed, in minutes, for notification when
  there is no user interface clicking, mouseover, scrolling, or
  resizing.
* Notifies client side pollers to stop, resulting in sessions expiring
  at
  the 'tools.sessions.timeout' value.
* If less than 1, results in no timeout notification ever being
  triggered
  (Sessions stay alive for as long as the browser is open).
* Default: 60

js_no_cache = <boolean>
* DEPRECATED.
* Toggles the JavaScript cache control.
* Default: false

cacheBytesLimit = <integer>
* When appServerPorts is set to a non-zero value, splunkd can keep a
  small cache of static assets in memory.
* When the total size of the objects in cache grows larger than this
setting, in bytes, splunkd begins ageing entries out of the cache.
* If set to zero, disables the cache.
* Default: 4194304

```
cacheEntriesLimit = <integer>
```
* When appServerPorts is set to a non-zero value, splunkd can keep a small cache of static assets in memory.
* When the number of the objects in cache grows larger than this, splunkd begins ageing entries out of the cache.
* If set to zero, disables the cache.
* Default: 16384

```
staticCompressionLevel = <integer>
```
* When appServerPorts is set to a non-zero value, splunkd can keep a small cache of static assets in memory.
* Splunkd stores these assets in a compressed format, and the assets can usually be served directly to the web browser in compressed format.
* This level can be a number between 1 and 9. Lower numbers use less CPU time to compress objects, but the resulting compressed objects will be larger.
* There is not much benefit to decreasing the value of this setting from its default. Not much CPU time is spent compressing the objects.
* Default: 9

```
enable_autocomplete_login = <boolean>
```
* Indicates if the main login page lets browsers autocomplete the username.
* If "true", browsers may display an autocomplete drop down in the username field.
* If "false", browsers may not show autocomplete drop down in the username field.
* Default: false

```
verifyCookiesWorkDuringLogin = <boolean>
```
* Normally, the login page makes an attempt to see if cookies work properly in the user's browser before allowing them to log in.
* If you set this to "false", this check is skipped.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* Do not set to "false" in normal operations.
* Default: true

```
minify_js = <boolean>
```
* Whether the static JavaScript files for modules are consolidated and minified.
* Setting this to "true" improves client-side performance by reducing the number of HTTP requests and the size of HTTP responses.

```
minify_css = <boolean>
```
* Indicates whether the static CSS files for modules are consolidated
and
  minified
* Setting this to "true" improves client-side performance by reducing
  the number of HTTP
  requests and the size of HTTP responses.
* Due to browser limitations, disabling this when using Internet
  Explorer
  version 9 and earlier might result in display problems.

**trap_module_exceptions** = <boolean>
* Whether or not the JavaScript for individual modules is wrapped in a
  try/catch
* If "true", syntax errors in individual modules do not cause the UI to
  hang, other than when using the module in question.
* Set to "false" when developing apps.

**enable_pivot_adhoc_acceleration** = <boolean>
* DEPRECATED in version 6.1 and later, use
  'pivot_adhoc_acceleration_mode'
  instead
* Whether or not the pivot interface uses its own ad-hoc acceleration
  when a data model is not accelerated.
* If "true", the pivot interface uses ad-hoc acceleration to make
  reporting
  in pivot faster and more responsive.
* In situations where data is not stored in time order, or where the
  majority
  of events are far in the past, disabling this behavior can improve the
  pivot experience.

**pivot_adhoc_acceleration_mode** = [Elastic | AllTime | None]
* Specifies the type of ad-hoc acceleration used by the pivot interface
  when a
  data model is not accelerated.
* If "Elastic", the pivot interface only accelerates the time range
  specified for reporting, and dynamically adjusts when this time range
  is changed.
* If "AllTime", the pivot interface accelerates the relevant data over
  all
  time. This makes the interface more responsive to time-range changes
  but places a larger load on system resources.
* If "None", the pivot interface does not use any acceleration. This
  means
  any change to the report requires restarting the search.
* Default: Elastic

**jschart_test_mode** = <boolean>
* Whether or not the JSChart module runs in Test Mode.
* If "true", JSChart module attaches HTML classes to chart elements for
  introspection.
* This negatively impacts performance and should be disabled unless you
  are actively using JSChart Test Mode.
To avoid browser performance impacts, the JSChart library limits the amount of data rendered in an individual chart.

\[
\text{jschart_truncation_limit} = \text{<integer>}
\]
* Cross-browser truncation limit.
* If set, takes precedence over the browser-specific limits below

\[
\text{jschart_truncation_limit.chrome} = \text{<integer>}
\]
* Chart truncation limit.
* For Chrome only.
* Default: 50000

\[
\text{jschart_truncation_limit.firefox} = \text{<integer>}
\]
* Chart truncation limit.
* For Firefox only.
* Default: 50000

\[
\text{jschart_truncation_limit.safari} = \text{<integer>}
\]
* Chart truncation limit.
* For Safari only.
* Default: 50000

\[
\text{jschart_truncation_limit.ie11} = \text{<integer>}
\]
* Chart truncation limit.
* For Internet Explorer version 11 only
* Default: 50000

\[
\text{jschart_series_limit} = \text{<integer>}
\]
* Chart series limit for all browsers.
* Default: 100

\[
\text{jschart_results_limit} = \text{<integer>}
\]
* DEPRECATED.
* Use 'data_sources.primary.params.count' in visualizations.conf instead.
* Chart results per series limit for all browsers.
* Overrides the results per series limit for individual visualizations.
* Default: 10000

\[
\text{choropleth_shape_limit} = \text{<integer>}
\]
* Choropleth map shape limit for all browsers.
* Default: 10000

\[
\text{dashboard_html_allow_inline_styles} = \text{<boolean>}
\]
* Whether or not to allow style attributes from inline HTML elements in dashboards.
* If "false", style attributes from inline HTML elements in dashboards will be removed
  to prevent potential attacks.
* Default: true
dashboard_html_allow_iframes = <boolean>
* Whether or not to allow iframes from HTML elements in dashboards.
* If "false", iframes from HTML elements in dashboards will be removed
to prevent potential attacks.
* Default: true

max_view_cache_size = <integer>
* The maximum number of views to cache in the appserver.
* Default: 300

pdfgen_is_available = [0 | 1]
* Specifies whether Integrated PDF Generation is available on this
search head.
* This is used to bypass an extra call to splunkd.
* Default (on platforms where node is supported): 1
* Default (on platforms where node is not supported): 0

version_label_format = <printf_string>
* Internal configuration.
* Overrides the version reported by the UI to *.splunk.com resources
* Default: %s

auto_refresh_views = [0 | 1]
* Specifies whether the following actions cause the appserver to ask
splunkd
to reload views from disk.
* Logging in through Splunk Web
* Switching apps
* Clicking the Splunk logo
* Default: 0

# Splunk bar options
#
# Internal config. May change without notice.
# Only takes effect if 'instanceType' is 'cloud'.
#
showProductMenu = <boolean>
* Used to indicate visibility of product menu.
* Default: False.

productMenuUriPrefix = <string>
* The domain product menu links to.
* Required if 'showProductMenu' is set to "true".

productMenuLabel = <string>
* Used to change the text label for product menu.
* Default: 'My Splunk'
showUserMenuProfile = <boolean>
* Used to indicate visibility of 'Profile' link within user menu.
* Default: false

# Header options
#
x_frame_options_sameorigin = <boolean>
* adds a X-Frame-Options header set to "SAMEORIGIN" to every response served
* by cherrypy
* Default: true

# Single Sign On (SSO)
#
remoteUser = <http_header_string>
* Remote user HTTP header sent by the authenticating proxy server.
* This header should be set to the authenticated user.
* CAUTION: There is a potential security concern regarding the treatment of HTTP headers.
* Your proxy provides the selected username as an HTTP header as specified above.
* If the browser or other HTTP agent were to specify the value of this header, probably any proxy would overwrite it, or in the case that the username cannot be determined, refuse to pass along the request or set it blank.
* However, Splunk Web (specifically, cherrypy) normalizes headers containing the dash and the underscore to the same value. For example, USER-NAME and USER_NAME are treated as the same in Splunk Web.
* This means that if the browser provides REMOTE-USER and Splunk Web accepts REMOTE_USER, theoretically the browser could dictate the username.
* In practice, however, the proxy adds its headers last, which causes them to take precedence, making the problem moot.
* See also the 'remoteUserMatchExact' setting which can enforce more exact header matching when running with 'appServerPorts' enabled.
* Default: 'REMOTE_USER'

remoteGroups = <http_header_string>
* Remote groups HTTP header name sent by the authenticating proxy server.
* This value is used by Splunk Web to match against the header name.
* The header value format should be set to comma-separated groups that
the user belongs to.
* Example of header value: Products,Engineering,Quality Assurance
* No default.

remoteGroupsQuoted = <boolean>
* Whether or not the group header value can be comma-separated quoted entries.
* This setting is considered only when 'remoteGroups' is set.
* If "true", the group header value can be comma-separated quoted entries.
* NOTE: Entries themselves can contain commas.
* Example of header value with quoted entries:
  "Products","North America, Engineering","Quality Assurance"
* Default: false (group entries should be without quotes.)

remoteUserMatchExact = [0 | 1]
* Whether or not to consider dashes and underscores in a remoteUser header to be distinct.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* When set to "1", considers dashes and underscores distinct (so "Remote-User" and "Remote_User" are considered different headers.)
* When set to 0, dashes and underscores are not considered to be distinct, to retain compatibility with older versions of Splunk software.
* Set to 1 when you set up SSO with 'appServerPorts' enabled.
* Default: 0

remoteGroupsMatchExact = [0 | 1]
* Whether or not to consider dashes and underscores in a remoteGroup header to be distinct.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* When set to 1, considers dashes and underscores distinct (so "Remote-Groups" and "Remote_Groups" are considered different headers)
* When set to 0, dashes and underscores are not considered to be distinct, to retain compatibility with older versions of Splunk software.
* Set to 1 when you set up SSO with 'appServerPorts' enabled.
* Default: 0

SSOMode = [permissive | strict]
* Whether SSO behaves in either permissive or strict mode.
* When set to "permissive": Requests to Splunk Web that originate from an untrusted IP address are redirected to a login page where they can log into Splunk Web without using SSO.
* When set to "strict": All requests to Splunk Web will be restricted to those
originating from a trusted IP except those to endpoints that do not require authentication.
* Default: strict

trustedIP = <ip_address>
* The IP address of the authenticating proxy (trusted IP).
* Splunk Web verifies it is receiving data from the proxy host for all SSO requests.
* Set to a valid IP address to enable SSO.
* If 'appServerPorts' is set to a non-zero value, this setting can accept a richer set of configurations, using the same format as the 'acceptFrom' setting.
* Default: not set; the normal value is the loopback address (127.0.0.1).

allowSsoWithoutChangingServerConf = [0 | 1]
* Whether or not to allow SSO without setting the 'trustedIP' setting in server.conf as well as in web.conf.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* If set to 1, enables web-based SSO without a 'trustedIP' setting configured in server.conf.
* Default: 0

testing_endpoint = <relative_uri_path>
* The root URI path on which to serve Splunk Web unit and integration testing resources.
* NOTE: This is a development only setting, do not use in normal operations.
* Default: /testing

testing_dir = <relative_file_path>
* The path relative to $SPLUNK_HOME that contains the testing files to be served at endpoint defined by 'testing_endpoint'.
* NOTE: This is a development only setting, do not use in normal operations.
* Default: share/splunk/testing

ssoAuthFailureRedirect = <scheme>://<URL>
* The redirect URL to use if SSO authentication fails.
* Examples:
  * http://www.example.com
  * https://www.example.com
* Default: empty string; Splunk Web shows the default unauthorized error page if SSO authentication fails.

# Results export server config
export_timeout = <integer>
* When exporting results, the number of seconds the server waits before
closing the connection with splunkd.
* If you do not set a value for export_timeout, Splunk Web uses the
value
  for the 'splunkdConnectionTimeout' setting.
* Set 'export_timeout' to a value greater than 30 in normal operations.
* No default.

#
# cherrypy HTTP server config
#

server.thread_pool = <integer>
* The minimum number of threads the appserver is allowed to maintain.
  * Default: 20

server.thread_pool_max = <integer>
* The maximum number of threads the appserver is allowed to maintain.
  * Default: -1 (unlimited)

server.thread_pool_min_spare = <integer>
* The minimum number of spare threads the appserver keeps idle.
  * Default: 5

server.thread_pool_max_spare = <integer>
* The maximum number of spare threads the appserver keeps idle.
  * Default: 10

server.socket_host = <ip_address>
* Host values may be any IPv4 or IPv6 address, or any valid hostname.
  * The string 'localhost' is a synonym for '127.0.0.1' (or '::1', if your
    hosts file prefers IPv6).
  * The string '0.0.0.0' is a special IPv4 entry meaning "any active
    interface"
    (INADDR_ANY), and "::" is the similar IN6ADDR_ANY for IPv6.
  * Default (if 'listenOnIPV6' is set to "no": 0.0.0.0
  * Default (otherwise): "::"

server.socket_timeout = <integer>
* The timeout, in seconds, for accepted connections between the browser
  and
  Splunk Web
  * Default: 10

listenOnIPV6 = <no | yes | only>
* By default, Splunk Web listens for incoming connections using
  IPv4 only.
* To enable IPv6 support in splunkweb, set this to "yes". Splunk Web
  simultaneously listens for connections on both IPv4 and IPv6
  protocols.
* To disable IPv4 entirely, set to "only", which causes SPLunk Web
to exclusively accept connections over IPv6.
* To listen on an IPv6 address, also set 'server.socket_host' to "::".

max_upload_size = <integer>
* The hard maximum limit, in megabytes, of uploaded files.
* Default: 500

log.access_file = <filename>
* The HTTP access log filename.
* This file is written in the default $SPLUNK_HOME/var/log directory.
* Default: web_access.log

log.access_maxsize = <integer>
* The maximum size, in bytes, that the web_access.log file can be.
* Comment out or set to 0 for unlimited file size.
* Splunk Web rotates the file to web_access.log.0 after the 'log.access_maxsize' is reached.
* See the 'log.access_maxfiles' setting to limit the number of backup files created.
* Default: 0 (unlimited size).

log.access_maxfiles = <integer>
* The maximum number of backup files to keep after the web_access.log file has reached its maximum size.
* CAUTION: Setting this to very high numbers (for example, 10000) can affect performance during log rotation.
* Default (if 'access_maxsize' is set): 5

log.error_maxsize = <integer>
* The maximum size, in bytes, the web_service.log can be.
* Comment out or set to 0 for unlimited file size.
* Splunk Web rotates the file to web_service.log.0 after the max file size is reached.
* See 'log.error_maxfiles' to limit the number of backup files created.
* Default: 0 (unlimited file size).

log.error_maxfiles = <integer>
* The maximum number of backup files to keep after the web_service.log file has reached its maximum size.
* CAUTION: Setting this to very high numbers (for example, 10000) can affect performance during log rotations
* Default (if 'access_maxsize' is set): 5

log.screen = <boolean>
* Whether or not runtime output is displayed inside an interactive TTY.
* Default: true

request.show_tracebacks = <boolean>
* Whether or not an exception traceback is displayed to the user on

574
fatal
  exceptions.
* Default: true

engine.autoreload_on = <boolean>
* Whether or not the appserver will auto-restart if it detects a python
  file
  has changed.
* Default: false

tools.sessions.on = true
* Whether or not user session support is enabled.
  * Always set this to true.

tools.sessions.timeout = <integer>
* The number of minutes of inactivity before a user session is
  expired.
* The countdown for this setting effectively resets every minute through
  browser activity until the 'ui_inactivity_timeout' setting is
  reached.
* Use a value of 2 or higher, as a value of 1 causes a race condition
  with
  the browser refresh, producing unpredictable behavior.
* Low values are not useful except for testing.
* Default: 60

tools.sessions.restart_persist = <boolean>
* Whether or not the session cookie is deleted from the browser when the
  browser quits.
* If set to "false", then the session cookie is deleted from the browser
  upon the browser quitting.
* If set to "true", then sessions persist across browser restarts, assuming
  the 'tools.sessions.timeout' has not been reached.
* Default: true

tools.sessions.httponly = <boolean>
* Whether or not the session cookie is available to running JavaScript
  scripts.
* If set to "true", the session cookie is not available to running
  JavaScript
  scripts. This improves session security.
* If set to "false", the session cookie is available to running
  JavaScript
  scripts.
* Default: true

tools.sessions.secure = <boolean>
* Whether or not the browser must transmit session cookies over an HTTPS
  connection when Splunk Web is configured to serve requests using HTTPS
  (the 'enableSplunkWebSSL' setting is "true".)
* If set to "true" and 'enableSplunkWebSSL' is also "true", then the
browser must transmit the session cookie over HTTPS connections. This improves session security.
* See the 'enableSplunkWebSSL' setting for details on configuring HTTPS session support.
* Default: true

tools.sessions.forceSecure = <boolean>
* Whether or not the secure bit of a session cookie that has been sent over HTTPS is set.
* If a client connects to a proxy server over HTTPS, and the back end connects to Splunk over HTTP, then setting this to "true" forces the session cookie being sent back to the client over HTTPS to have the secure bit set.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* Default: false

response.timeout = <integer>
* The timeout, in seconds, to wait for the server to complete a response.
* Some requests, such as uploading large files, can take a long time.
* Default: 7200 (2 hours).

tools.sessions.storage_type = [file]
tools.sessions.storage_path = <filepath>
* Specifies the session information storage mechanisms.
* Set 'tools.sessions.storage_type' and 'tools.sessions.storage_path' to
  
  use RAM based sessions instead.
* Use an absolute path to store sessions outside of $SPLUNK_HOME.
* Default: storage_type=file, storage_path=var/run/splunk

tools.decode.on = <boolean>
* Whether or not all strings that come into CherryPy controller methods are
  
  decoded as unicode (assumes UTF-8 encoding).
* CAUTION: Setting this to false will likely break the application, as all incoming strings are assumed to be unicode.
* Default: true

tools.encode.on = <boolean>
* Whether or not to encode all controller method response strings into UTF-8 str objects in Python.
* CAUTION: Disabling this will likely cause high byte character encoding to fail.
* Default: true

tools.encode.encoding = <codec>
* Forces all outgoing characters to be encoded into UTF-8.
* This setting only takes effect when 'tools.encode.on' is set to "true".
* By setting this to "utf-8", CherryPy default behavior of observing the Accept-Charset header is overwritten and forces utf-8 output.
* Only change this if you know a particular browser installation must receive some other character encoding (Latin-1 iso-8859-1, etc)
* CAUTION: Change this setting at your own risk.
* Default: utf-8

**tools.proxy.on = <boolean>**
* Used for running Apache as a proxy for Splunk Web, typically for SSO configurations.
* Search the CherryPy website for "apache proxy" for more information.
* For Apache 1.x proxies only, set to "true". This configuration tells CherryPy (the Splunk Web HTTP server) to look for an incoming X-Forwarded-Host header and to use the value of that header to construct canonical redirect URLs that include the proper host name. For more information, refer to the CherryPy documentation on running behind an Apache proxy. This setting is only necessary for Apache 1.1 proxies.
* For all other proxies, you must set to "false".
* Default: false

**tools.proxy.base = <scheme>://<URL>**
* The proxy base URL in Splunk Web.
* Default: empty string

**pid_path = <filepath>**
* Specifies the path to the Process IDentification (pid) number file.
* Must be set to "var/run/splunk/splunkweb.pid".
* CAUTION: Do not change this parameter.

**enabled_decomposers = <intention> [, <intention>]...**
* Added in Splunk 4.2 as a short term workaround measure for apps which happen to still require search decomposition, which is deprecated with 4.2.
* Search decomposition will be entirely removed in a future release.
* A comma-separated list of allowed intentions.
* Modifies search decomposition, which is a Splunk Web internal behavior.
* Can be controlled on a per-app basis.
* If set to an empty string, no search decomposition occurs, which causes some usability problems with Report Builder.
* The current possible values are: addcommand, stats, addterm, addtermgt, addtermlt, setfields, excludefields, audit, sort, plot
* Default: "plot", leaving only the plot intention enabled.

**simple_xml_perf_debug = <boolean>**
* Whether or not Simple XML dashboards log performance metrics to the browser console.
* If set to "true", Simple XML dashboards log some performance metrics to
  the browser console.
* Default: false

job_min_polling_interval = <integer>
* The minimum polling interval, in milliseconds, for search jobs.
* This is the initial wait time for fetching results.
* The poll period increases gradually from the minimum interval to the maximum interval when search is in a queued or parsing state (and not a running state) for some time.
* Set this value between 100 and 'job_max_polling_interval' milliseconds.
* Default: 100

job_max_polling_interval = <integer>
* The maximum polling interval, in milliseconds, for search jobs.
* This is the maximum wait time for fetching results.
* In normal operations, set to 3000.
* Default: 1000

acceptFrom = <network_acl> ...

* Lists a set of networks or addresses from which to accept connections.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* Separate multiple rules with commas or spaces.
* Each rule can be in one of the following formats:
  1. A single IPv4 or IPv6 address (examples: "10.1.2.3", "fe80::4a3")
  2. A Classless Inter-Domain Routing (CIDR) block of addresses (examples: "10/8", "192.168.1/24", "fe80:1234/32")
  3. A DNS name, possibly with a "*" used as a wildcard (examples: "myhost.example.com", "*.splunk.com")
  4. "*", which matches anything
* You can also prefix an entry with '!' to cause the rule to reject the connection. The input applies rules in order, and uses the first one that matches.
  For example, "!10.1/16, *" allows connections from everywhere except the 10.1.*.* network.
* Default: "*" (accept from anywhere)

maxThreads = <integer>
* The number of threads that can be used for active HTTP transactions.
* This setting only takes effect when appServerPorts is set to a non-zero value.
* This value can be limited to constrain resource usage.
* If set to 0, a limit is automatically picked based on estimated server capacity.
* If set to a negative number, no limits are enforced.
* Default: 0
maxSockets = <integer>
* The number of simultaneous HTTP connections that Splunk Web can accept.
* This setting only takes effect when appServerPorts is set to a non-zero value.
* This value can be limited to constrain resource usage.
* If set to 0, a limit is automatically picked based on estimated server capacity.
* If set to a negative number, no limits are enforced.
* Default: 0

keepAliveIdleTimeout = <integer>
* How long, in seconds, that the Splunk Web HTTP server lets a keep-alive connection remain idle before forcibly disconnecting it.
* If this number is less than 7200, it will be set to 7200.
* Default: 7200

busyKeepAliveIdleTimeout = <integer>
* How long, in seconds, that the Splunk Web HTTP server lets a keep-alive connection remain idle while in a busy state before forcibly disconnecting it.
* CAUTION: Too large a value that can result in file descriptor exhaustion due to idling connections.
* If this number is less than 12, it will be set to 12.
* Default: 12

forceHttp10 = auto|never|always
* How the HTTP server deals with HTTP/1.0 support for incoming clients.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* When set to "always", the REST HTTP server does not use some HTTP 1.1 features such as persistent connections or chunked transfer encoding.
* When set to "auto", it limits HTTP 1.1 features only if the client sent no User-Agent header, or if the user agent is known to have bugs in its HTTP/1.1 support.
* When set to "never", it always allows HTTP 1.1, even to clients it suspects might be buggy.
* Default: auto

crossOriginSharingPolicy = <origin_acl> ...
* A list of HTTP Origins for which to return Access-Control-Allow-* (CORS) headers.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* These headers tell browsers that Splunk Web trusts web applications at those sites to make requests to the REST interface.
* The origin is passed as a URL without a path component (for example
"https://app.example.com:8000")
* This setting can take a list of acceptable origins, separated
  by spaces and/or commas
* Each origin can also contain wildcards for any part. Examples:
  *://app.example.com:* (either HTTP or HTTPS on any port)
  https://*.example.com (any host under example.com, including
  example.com itself)
* An address can be prefixed with a '!' to negate the match, with
  the first matching origin taking precedence. For example,
  "!://evil.example.com:* *://*.example.com:*" to not avoid
  matching one host in a domain.
* "*" can also be used to match all origins.
* Default: empty string

allowSslCompression = <boolean>
* Whether or not the server lets clients negotiate SSL-layer data
  compression.
* This setting only takes effect when 'appServerPorts' is set
to a non-zero value. When 'appServerPorts' is zero or not set, this
setting
  is "true".
* If set to "true", the server lets clients negotiate SSL-layer
data compression.
* The HTTP layer has its own compression layer which is usually
  sufficient.
* Default (if 'appServerPorts' is set and not 0): false
* Default (if 'appServerPorts' is 0 or not set): true

allowSslRenegotiation = <boolean>
* Whether or not the server lets clients renegotiate SSL connections.
* This setting only takes effect when 'appServerPorts' is set to a
  non-zero value.
* In the SSL protocol, a client may request renegotiation of the
  connection
    settings from time to time.
* Setting this to "false" causes the server to reject all renegotiation
    attempts, breaking the connection.
* This limits the amount of CPU a single TCP connection can use, but it
  can cause connectivity problems especially for long-lived connections.
* Default: true

sendStrictTransportSecurityHeader = <boolean>
* Whether or not the REST interface sends a "Strict-Transport-Security"
  header with all responses to requests made over SSL.
* This setting only takes effect when 'appServerPorts' is set to a
  non-zero value.
* If set to "true", the REST interface sends a
  "Strict-Transport-Security"
  header with all responses to requests made over SSL.
* This can help avoid a client being tricked later by a
  Man-In-The-Middle
  attack to accept a non-SSL request.
* This requires a commitment that no non-SSL web hosts will ever be run on this hostname on any port. For example, if splunkweb is in non-SSL mode this can break the ability of browser to connect to it.
* Enable this setting with caution.
* Default: false

```
dedicatedIoThreads = <integer>
```
* The number of dedicated threads to use for HTTP input/output operations.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* If set to zero, HTTP I/O is performed in the same thread that accepted the TCP connection.
* If set set to a non-zero value, separate threads run to handle the HTTP I/O, including SSL encryption.
* Typically this does not need to be changed. For most usage scenarios using the same the thread offers the best performance.
* Default: 0

```
replyHeader.<name> = <string>
```
* Adds a static header to all HTTP responses that this server generates.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* For example, "replyHeader.My-Header = value" causes Splunk Web to include the response header "My-Header: value" in the reply to every HTTP request to it.
* No default.

```
termsOfServiceDirectory = <directory>
```
* The directory to look in for a "Terms of Service" document that each user must accept before logging into Splunk Web.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* Inside the directory the TOS should have a filename in the format "<number>.html"
  * <number> is in the range 1 to 18446744073709551615.
  * The active TOS is the filename with the larger number. For example, if there are two files in the directory named "123.html" and "456.html", then 456 will be the active TOS version.
  * If a user has not accepted the current version of the TOS, they must accept it the next time they try to log in. The acceptance times will be recorded inside a "tos.conf" file inside an app called "tos".
  * If the "tos" app does not exist, you must create it for acceptance times to be recorded.
  * The TOS file can either be a full HTML document or plain text, but it must have the ".html" suffix.
* You do not need to restart Splunk Enterprise when adding files to the TOS directory.
  * Default: empty string (no TOS)

appServerProcessShutdownTimeout = <nonnegative integer>[smhd]
* The amount of time splunkd waits for a Python-based application server process to handle outstanding or existing requests.
* This setting only takes effect when 'appServerPorts' is set to a non-zero value.
* If a Python-based application server process "outlives" this timeout, splunkd forcibly terminates the process.
* Default: '10m' (10 minutes).

enableWebDebug = <boolean>
* Whether or not the debug REST endpoints are accessible, for example., /debug/**splat.
  * Default: false

allowableTemplatePaths = <directory> [, <directory>]...
* A comma-separated list of template paths that may be added to the template lookup whitelist.
* Paths are relative to $SPLUNK_HOME.
  * Default: empty string

enable_risky_command_check = <boolean>
* Whether or not checks for data-exfiltrating search commands are enabled.
  * default true

customFavicon = <pathToMyFile, myApp:pathToMyFile, or blank for default>
* Customizes the favicon image across the entire application.
* If no favicon image file, the favicon default: the Splunk favicon.
  * Supported favicon image files are .ico files, and should be square images.
* Place the favicon image file in the default or manual location:
  * Default destination folder:
    * $SPLUNK_HOME/etc/apps/search/appserver/static/customfavicon.
  * Example: If your favicon image is located at $SPLUNK_HOME/etc/apps/search/appserver/static/customfavicon/favicon.ico, set 'customFavicon' to "customfavicon/favicon.ico".
  * Manual location: Place the file in $SPLUNK_HOME/etc/apps/<myApp>/appserver/static/<pathToMyFile>, and set 'customFavicon' to "<myApp:pathToMyFile>".
  * Default: not set, Splunk Web uses the Splunk favicon.

loginCustomLogo = <fullUrl, pathToMyFile, myApp:pathToMyFile, or blank for default>
* Customizes the logo image on the login page.
* If no image file, the logo Default: the Splunk logo.
* Supported images are:
* Full URL image file (secured or not secured), such as https://www.splunk.com/logo.png or http://www.splunk.com/logo.png.
* Image file, such as .jpg or .png. All image formats are supported.
* Place logo image file in default or manual location:
  * Default destination folder:
    $SPLUNK_HOME/etc/apps/search/appserver/static/logincustomlogo.
    * Example: If your logo image is located at
      $SPLUNK_HOME/etc/apps/search/appserver/static/logincustomlogo/logo.png, type
      loginCustomLogo = logincustomlogo/logo.png.
    * Manual location:
      $SPLUNK_HOME/etc/apps/<myApp>/appserver/static/<pathToMyFile>, and type
      loginCustomLogo = <myApp:pathToMyFile>.
* The maximum image size is 485px wide and 100px high. If the image exceeds these limits, the image is automatically resized.
* Default: not set, Splunk Web uses the Splunk logo.

loginBackgroundImageOption = [default | custom | none]
* Controls display of the background image of the login page.
* "default" displays the Splunk default background image.
* "custom" uses the background image defined by the backgroundImageCustomName setting.
* "none" removes any background image on the login page. A dark background color is applied.
* Default: "default".

loginCustomBackgroundColor = <pathToMyFile or myApp:pathToMyFile>
* Customizes the login page background image.
  * Supported image files include .jpg, .jpeg or .png with a maximum file size of 20MB.
  * A landscape image is recommended, with a minimum resolution of 1024x640 pixels.
  * Using Splunk Web:
    * Upload a custom image to a manager page under General Settings.
    * The login page background image updates automatically.
  * Using the CLI or a text editor:
    * Set 'loginBackgroundColorOption' to "custom".
    * Place the custom image file in the default or manual location:
      * Default destination folder:
        $SPLUNK_HOME/etc/apps/search/appserver/static/logincustombg.
        * Example: If your image is located at
          $SPLUNK_HOME/etc/apps/search/appserver/static/logincustombg/img.png, set
          'loginCustomBackgroundColor' to "logincustombg/img.png".
        * Manual location:
          $SPLUNK_HOME/etc/apps/<myApp>/appserver/static/<pathToMyFile>, and set
          'loginCustomBackgroundColor' to
          "<myApp:pathToMyFile>".
        * The login page background image updates automatically.
        * Default: not set (If no custom image is used, the default Splunk background image displays).

loginFooterOption = [default | custom | none]
* Controls display of the footer message of the login page.
  * "default" displays the Splunk copyright text.
  * "custom" uses the footer text defined by the loginFooterText setting.
  * "none" removes any footer text on the login page.
  * NOTE: This option is made available only to OEM customers participating in
    the Splunk OEM Partner Program and is subject to the relevant terms of
    the Master OEM Agreement. All other customers or partners are prohibited
    from
    removing or altering any copyright, trademark, and/or other
    intellectual
    property or proprietary rights notices of Splunk placed on or embedded
    in any Splunk materials.
  * Default: "default".

  loginFooterText = <footer_text>
* The text to display in the footer of the login page.
* Supports any text, including HTML.
* To display, the parameter 'loginFooterOption' must be set to "custom".

loginDocumentTitleOption = [default | custom | none]
* Controls display of the document title of the login page.
* Default: "default".
* "default" displays: "<page_title> | Splunk".
* "none" removes the branding on the document title of the login page:
  "<page_title>".
* "custom" uses the document title text defined by the
  loginDocumentTitleText setting.
* NOTE: This option is made available only to OEM customers participating in
  the Splunk OEM Partner Program and is subject to the relevant terms of
  the
  Master OEM Agreement. All other customers or partners are prohibited
  from
  removing or altering any copyright, trademark, and/or other
  intellectual
  property or proprietary rights notices of Splunk placed on or embedded
  in any Splunk materials.
* Default: "default".

loginDocumentTitleText = <document_title_text>
* The text to display in the document title of the login page.
* Text only.
* To display, the parameter 'loginDocumentTitleOption' must be set to
  "custom".

loginPasswordHint = <default_password_hint>
* The text to display the password hint at first time login on the login
  page.
* Text only.
* Default: "changeme"
appNavReportsLimit = <integer>
* Maximum number of reports to fetch to populate the navigation
drop-down
  menu of an app.
* An app must be configured to list reports in its navigation XML
  configuration before it can list any reports.
* Set to -1 to display all the available reports in the navigation menu.
* NOTE: Setting to either -1 or a value that is higher than the default
  might result in decreased browser performance due to listing large numbers
  of available reports in the drop-down menu.
* Default: 500

[framework]
# Put App Framework settings here
django_enable = <boolean>
* Specifies whether Django should be enabled or not
  * Default: True
  * Django will not start unless an app requires it
django_path = <filepath>
* Specifies the root path to the new App Framework files,
  relative to $SPLUNK_HOME
  * Default: etc/apps/framework
django_force_enable = <boolean>
* Specifies whether to force Django to start, even if no app requires it
  * Default: False

# custom cherrypy endpoints
#

[endpoint:<python_module_name>]
* Registers a custom python CherryPy endpoint.
  * The expected file must be located at:
    $SPLUNK_HOME/etc/apps/<APP_NAME>/appserver/controllers/<PYTHON_MODULE_NAME>.py
  * This module's methods will be exposed at
    /custom/<APP_NAME>/<PYTHON_MODULE_NAME>/<METHOD_NAME>

# exposed splunkd REST endpoints
#
[expose:<unique_name>]
* Registers a splunkd-based endpoint that should be made available to
  the UI under the "/splunkd" and "/splunkd/__raw" hierarchies.
* The name of the stanza does not matter as long as it begins with
  "expose:"
* Each stanza name must be unique.
pattern = <url_pattern>
* The pattern to match under the splunkd /services hierarchy.
* For instance, "a/b/c" would match URIs "/services/a/b/c" and
  "/servicesNS/*/a/b/c",
* The pattern cannot include leading or trailing slashes.
* Inside the pattern an element of "*" matches a single path element.
  For example, "a/*/c" would match "a/b/c" but not "a/1/2/c".
* A path element of "**" matches any number of elements. For example,
  "a/**/c" would match both "a/1/c" and "a/1/2/3/c".
* A path element can end with a "*" to match a prefix. For example,
  "a/elem-*"/b" would match "a/elem-123/c".

methods = <method_lists>
* A comma-separated list of methods to allow from the web browser
  (example: "GET,POST,DELETE").
* Default: "GET"

oidEnabled = [0 | 1]
* Whether or not a REST endpoint is capable of taking an embed-id as a
  query parameter.
* If set to 1, the endpoint is capable of taking an embed-id
  as a query parameter.
* This is only needed for some internal splunk endpoints, you probably
  should not specify this for app-supplied endpoints
* Default: 0

skipCSRFProtection = [0 | 1]
* Whether or not Splunk Web can safely post to an endpoint without
  applying
  Cross-Site Request Forgery (CSRF) protection.
* If set to 1, tells Splunk Web that it is safe to post to this endpoint
  without applying CSRF protection.
* This should only be set on the login endpoint (which already contains
  sufficient auth credentials to avoid CSRF problems).
* Default: 0

web.conf.example

#   Version 7.2.0
#
# This is an example web.conf. Use this file to configure data web
# settings.
#
# To use one or more of these configurations, copy the configuration
# block
# into web.conf in $SPLUNK_HOME/etc/system/local/. You must restart
# Splunk
# to enable configurations.
# To learn more about configuration files (including precedence) please see
# the documentation located at
# http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

# This stanza heading must precede any changes.
[settings]

# Change the default port number:
httpport = 12800
# Also run the python application server on a non-default port:
appServerPorts = 12801

# Turn on SSL:
enableSplunkWebSSL = true
# absolute paths may be used here.
privKeyPath = /home/user/certs/myprivatekey.pem
serverCert = /home/user/certs/mycacert.pem
# NOTE: non-absolute paths are relative to $SPLUNK_HOME
Troubleshooting

Search IT Service Intelligence logs

IT Service Intelligence log files have a prefix of `itsi_`.

- IT Service Intelligence search command logs are located in
  $SPLUNK_HOME/var/run/splunk/dispatch/<session_id>/itsi_search.log.
- All other ITSI logs are located in `$SPLUNK_HOME/var/log/splunk`.

All ITSI logs have a sourcetype of `itsi_internal_log` to make them easy to search.

Steps

1. Run the following Splunk search to search ITSI logs:
   ```splunk
   index = _internal sourcetype=itsi_internal_log
   ```
2. Click the `source` field under Selected Fields to see specific log files.

For Windows deployments, the ITSI search command log, `itsi_search.log`, cannot be searched in Splunk Web. You must open the file on the Windows host using a text editor.

See also

- See ITSI Health Check dashboard for IT Service Intelligence log messages.
- See What Splunk software logs about itself for general information about Splunk logging.

Installation and configuration considerations and issues

This topic contains important considerations and issues that you should be aware of when installing and configuring Splunk IT Service Intelligence (ITSI). This list supplements (and in some cases overlaps) with known issues found in the official ITSI release notes. For more information, see Known issues in *Release Notes*. 

588
**Installation/Upgrade**

**Ubuntu**

When installing ITSI on Ubuntu, use Bash shell. Do not use Dash shell.

- Splunk assumes Bash shell.
- Using Dash shell can result in zombie processes.

**Windows**

When installing ITSI on Windows, rename `inputs.conf.windows` to `inputs.conf`, in both `SA-Utils` and `SA-ThreatIntelligence` default directories.

For example:

```bash
cd $SPLUNK_HOME/etc/apps/SA-Utils/default/
cp inputs.conf inputs.conf.bak
cp inputs.conf.windows inputs.conf
rm inputs.conf.windows
```

**Run script to set the default team to Global**

The new "Teams" feature introduced in ITSI 3.0 requires all services be assigned to a team. When upgrading to ITSI 3.0 or later from a version of ITSI prior to 3.0, the migration script creates the default team, Global, in the kvstore and assigns all existing services to the Global team.

If this step in the migration fails, you will receive the error "Failed to import Team settings" and ITSI will be unusable. If you encounter this error when upgrading to version 3.0, download and install ITSI version 3.0.1. ITSI 3.0.1 contains a Python script called `itsi_reset_default_team.py` in the `apps/SA-ITOA/bin` directory. Running this script manually creates the Global team in the kvstore which completes the migration.

If you encounter this error when upgrading to version 3.0.1 from a version prior to 3.0, manually run the Python script included in version 3.0.1.

To run the `itsi_reset_default_team.py` script in version 3.0.1:

1. Run the following command on any search head in your ITSI deployment:
   ```bash
cd $SPLUNK_HOME/etc/apps/SA-ITOA/bin
   $SPLUNK_HOME/bin/splunk cmd python itsi_reset_default_team.py
   ```
2. Provide the splunkd port number and Splunk user name and password when prompted.
3. After the script has successfully finished, the Global group is created in the kvstore.
4. Restart the Splunk platform.

It is recommended to install version 3.0.1, rather than downloading and running the `itsi_reset_default_team.py` file from the ITSI 3.0.1 installation package. ITSI version 3.0.1 has better error logging than version 3.0 so in the event that the script does not work, version 3.0.1 will provide more information on the reason for the failure.

**Service Analyzer**

Real-time searches might hang when the number of KPIs/Services is set too high.

- You can change the number of tiles shown on the Service Analyzer dashboard. If you set the number of tiles too high (50 or greater) this can cause the two indexed real time searches that generate the tiles to hang and show a "Waiting for data" message. This occurs only on the specific search head.
- This seems to occur as a result of KV Store performance issues in a search head cluster environment.

**Workarounds:**

- Avoid changing the number of tiles setting in a search head cluster environment.
- Use filters to display only the specific services and KPIs that require monitoring.
- Set the number of visible tiles to the lowest number possible.

**Data backfill on Windows**

On Windows, when using data backfill, ITSI might not work as expected and an error message might appear due to insufficient page file quota allocation.

**Workaround:** Double the current page file quota allocation on your Windows machine.
Duplicate entity aliases

Duplicate entity aliases occur when you create entities separately, but they happen to have overlapping aliases. Entity aliases must uniquely identify the entity. For example, host=appserver-01. The same alias field value cannot be used for more than one entity. If more than one entity is using the same alias field value (such as appserver-01), it can lead to a miscalculated service health score value (because extra entities are counted) and incorrect KPI base search results.

How to detect duplicate aliases

Use any of the following methods to detect duplicate entity aliases:

- If Splunk finds any duplicates, a warning message appears in the Messages menu. Click Show duplicates to open the ITSI Health Check dashboard which lists the entities with duplicate aliases.

- Navigate directly to the ITSI Health Check dashboard by clicking Search > Dashboards > ITSI Health Check from the ITSI main menu.

- Scan your entities manually to check for duplicates.

- Run the following search to detect duplicates:

  ```
  | inputlookup itsi_entities | eval original='identifier.values' | mvexpand original | eval key=_key | stats count values(identifier.values) AS entity_aliases values(title) AS entity_title values(key) AS entity_key values(services._key) AS service_keys by original | eval error=if(count>1,"dupe","") | where count>1
  ```

What causes duplicate aliases?

The following issues can lead to duplicate entity aliases:

- The same entity was imported several times with different methods and fields.
- An entity was imported by a specific module, but each time with a slightly different name as the title or alias.
- An entity was manually imported from a search or CSV file with different field values, or a case difference.
Clean up and avoid duplicates

Take the following steps to fix duplicate entity aliases and prevent future ones:

Always back up ITSI before attempting to fix duplicates (Configure > Backup/Restore).

♦ If the duplicates were caused by an automatic import, disable the import in Settings > Data inputs > IT Service Intelligence Asynchronous CSV Loader, then disable the appropriate inputs. Retry the import after you clean up the entities and normalize fields and aliases.
♦ Click Configure > Entities and edit the entity definitions with duplicate aliases. Keep the alias value for one of the entities and edit the other to remove the duplicate alias value.
♦ Merge the duplicates by moving all the fields that differ to one entity, then deleting the extra copy.
♦ Test your entity import searches to ensure there are no conflicts.

Services (or other objects) missing after upgrade

When the objects in ITSI are exported during a backup or migration, if the number of KPIs linked to a service is high, the instance can hit a kvstore memory size limit causing some objects to be dropped from the backup and lost after the upgrade.

Workaround: Increase the KV store bulk get limit in $SPLUNK_HOME/etc/apps/SA-ITOA/local/limits.conf and retry the backup or upgrade. Increase the max_size_per_result_mb value as necessary. The default limit is set to 500MB in ITSI version 3.1.4 and later.

[kvstore]
# The maximum size, in megabytes (MB), of the result that will be returned for a single query to a collection.
# ITSI requires approximately 50MB per 1,000 KPIs. Override this value if necessary.
# Default: 500 MB
max_size_per_result_mb = 500

This action will increase the memory used by the kvstore during operations.