# Table of Contents

## Introduction to Data Fabric Search
- DFS overview .......................................................................................................................................................... 1  
- Big data analysis .................................................................................................................................................. 3  
- Federated search .................................................................................................................................................. 5  
- DFS terminology ................................................................................................................................................ 10  
- Use cases for big data analysis .......................................................................................................................... 11  
- Use cases for federated searches ......................................................................................................................... 12  

## Set up DFS
- Installation quickstart ........................................................................................................................................ 14  
- Licensing information ...................................................................................................................................... 16  
- Hardware requirements .................................................................................................................................. 16  
- Compatibility matrix ........................................................................................................................................ 17  
- Upgrade requirements .................................................................................................................................... 18  
- Port configuration requirements .......................................................................................................................... 18  
- Configure the compute cluster using the app ..................................................................................................... 20  
- Set up a DFS deployment ................................................................................................................................. 20  
- Run DFS searches ............................................................................................................................................ 21  
- Set up service accounts for access control ........................................................................................................ 22  
- Set up a federated search .................................................................................................................................. 23  

## Secure a DFS deployment
- Secure a DFS deployment ................................................................................................................................. 32  

## DFS commands
- dfsjob ................................................................................................................................................................. 42  
- dedup ................................................................................................................................................................. 46  
- eval .................................................................................................................................................................... 48  
- fields ................................................................................................................................................................. 51  
- from ................................................................................................................................................................. 51  
- head ................................................................................................................................................................. 51  
- join ................................................................................................................................................................. 52  
- rename ............................................................................................................................................................. 54  
- reverse ............................................................................................................................................................ 55  
- stats ................................................................................................................................................................. 55  
- sort ................................................................................................................................................................. 56  
- tail ................................................................................................................................................................. 56  
- union ............................................................................................................................................................... 58  
- where ............................................................................................................................................................... 59  
- where ............................................................................................................................................................... 60  

## Troubleshoot DFS
- DFS search caveats ............................................................................................................................................. 62  
- DFS log files ...................................................................................................................................................... 62  
- Troubleshoot DFS ............................................................................................................................................. 65  
- Troubleshoot using Splunk DFS Manager ........................................................................................................ 66  
- Troubleshoot security issues ............................................................................................................................... 69  
- Troubleshoot federated searches ........................................................................................................................ 74  

- Licensing information ...................................................................................................................................... 16  
- Installation quickstart ........................................................................................................................................ 14  
- Hardware requirements .................................................................................................................................. 16  
- Compatibility matrix ........................................................................................................................................ 17  
- Upgrade requirements .................................................................................................................................... 18  
- Port configuration requirements .......................................................................................................................... 18  
- Configure the compute cluster using the app ..................................................................................................... 20  
- Set up a DFS deployment ................................................................................................................................. 20  
- Run DFS searches ............................................................................................................................................ 21  
- Set up service accounts for access control ........................................................................................................ 22  
- Set up a federated search .................................................................................................................................. 23  

- Secure a DFS deployment ................................................................................................................................. 32  

- Secure a DFS deployment ................................................................................................................................. 32  

- DFS overview .......................................................................................................................................................... 1  
- Big data analysis .................................................................................................................................................. 3  
- Federated search .................................................................................................................................................. 5  
- DFS terminology ................................................................................................................................................ 10  
- Use cases for big data analysis .......................................................................................................................... 11  
- Use cases for federated searches ......................................................................................................................... 12  

- DFS commands .......................................................................................................................................................... 42  
- dfsjob ................................................................................................................................................................. 42  
- dedup ................................................................................................................................................................. 46  
- eval ................................................................................................................................................................. 48  
- fields ................................................................................................................................................................. 51  
- from ................................................................................................................................................................. 51  
- head ................................................................................................................................................................. 51  
- join ................................................................................................................................................................. 52  
- rename ............................................................................................................................................................. 54  
- reverse ............................................................................................................................................................ 55  
- stats ................................................................................................................................................................. 55  
- sort ................................................................................................................................................................. 56  
- t
# Table of Contents

**Troubleshoot DFS**

DFS Error Messages................................................................................................................................. 76

**Third-party software**............................................................................................................................... 79

Third-party software.......................................................................................................................................... 79

**Release notes**.............................................................................................................................................. 80

What's new in this release.................................................................................................................................. 80

Known issues.................................................................................................................................................... 80

Fixed issues..................................................................................................................................................... 82

**Appendix 1: DFS REST endpoints**............................................................................................................ 84

fshpasswords..................................................................................................................................................... 84

federated........................................................................................................................................................... 87

**Appendix 2: DFS configuration files**........................................................................................................ 92

Edit configuration files for big data analysis................................................................................................. 92

authorize.conf.................................................................................................................................................... 93

federated.conf.................................................................................................................................................. 112

limits.conf......................................................................................................................................................... 113

savedsearches.conf....................................................................................................................................... 180

server.conf....................................................................................................................................................... 199
Introduction to Data Fabric Search

DFS overview

Data Fabric Search (DFS) is the new extended search platform that leverages the distributed processing power of compute clusters to broaden the scope and capability of Splunk Enterprise. Evolving and diverse data sources obscure visibility and makes it difficult to gather quick insights from data. Massive data stores also increase storage and processing costs. In short, analyzing data becomes harder when volume, variety, and the velocity of data grows at a massive scale.

DFS basically connects massive datasets from different data sources like mainframe computers, databases, IoT devices, and multiple Splunk deployments, which may have different storage and retention policies, into a single view. You can use DFS to search across multiple terabytes of data, billions of events without any performance impact, to gather Enterprise-wide insights into your data.

Traditionally, the Splunk platform runs searches from a single search head. The search query comes in through the search head and the results are returned back to the search head. The multiple reporting processors running on the search head in a single machine often create a performance bottleneck on the search head, which impacts the responsiveness of reporting, monitoring, and alert operations. A DFS job enhances search performance by distributing the search processing load to the compute cluster, so that processing and memory requirements do not cause a bottleneck at the search head. This data processing between the indexers, compute nodes, and search head optimizes the search process.

In short, the search head receives a DFS SPL search query. The search head only breaks up and defines the sequence and location of how the various components of the search will be implemented. The indexers pre-process and send the data to the DFS worker or compute cluster nodes. The compute cluster processes the search and sends the results back to the search head. The search head then applies relevant knowledge objects to the results and sends the final results back to the UI.

Big data challenges addressed by DFS

1. Searches may take too long to run.
   The following search may take around 90 minutes to run and return over a billion events.
   
   ... | stats count by city, org, ip | stats count
   
   High cardinality searches especially can generate millions of rows of results.

2. Memory restrictions may cause a search to fail.
   The following search may fail due to memory restrictions.
   
   ...| stats sum (duration) sum (price) by customer
   
   High cardinality searches especially can use extensive memory resources during event count estimation.

3. Inability to securely investigate data located in different regions.
   The following search may be a three step process that involves:
   1. running a search on index = us;
   2. running a search on index = eu;
   3. combining the two searches in a single report;

   index = us | stats count by cid | join type = inner left = L right = R where L.cid=R.cid [index = eu | stats count by cid]
In contrast, a DFS search can use a single SPL query to search across multiple Splunk deployments or external federated providers, while leveraging existing knowledge objects and maintaining user access permissions. You do not need to move data across deployments to run a DFS search or unlock data to ensure users can access only authorized datasets.

DFS features

DFS has the following features:

- **Big Data Analysis**: Analyzes and explores large amounts of data to search over a billion events within a single Splunk deployment with significant performance improvements. Additionally, DFS also allows you to perform high-cardinality searches where events have very uncommon or unique values. In fact, the higher the data cardinality, the higher the performance.

- **Federated Search**: Conducts searches and joins across multiple indexes on disparate Splunk deployments as seamlessly as if it was a single deployment.

Federated searches use an authorization model that enables the administrator to create service accounts for role-based **user authentication** across multiple Splunk deployments. A federated search head is a Splunk instance that handles search management functions, which directs search requests to federated providers that are remote Splunk Enterprise deployments. Federated searches provide the ability to correlate across a wider data fabric of multiple and disparate Splunk Enterprise deployments to access relevant datasets. The compute cluster applies the search pipeline to the results in a distributed manner. A remote search head is the Splunk Enterprise instance that resides on the remote Splunk deployment and conducts federated searches.

The following diagram illustrates the differences between a distributed search and a distributed search with a DFS compute cluster:
TLS is not enabled by default for data transport within a DFS deployment. For more information on securing your DFS deployment, see Secure a DFS deployment.

**Benefits of DFS**

In the last few years, there has been a tremendous boom in data generation which shows no signs of slowing down. DFS offers the following benefits when processing this “big data”:

**Scalability and complex joins**

DFS helps to aggregate complex datasets of billions of events to generate a quick report in minutes.

**Performance despite high data cardinality**

DFS can improve your ability to conduct high-cardinality searches on large volumes of data without compromising performance. If your dataset contains too many uncommon or unique values e.g. usernames, user IDs, email addresses etc. you can still search and gather insights from your data. In fact, the higher the cardinality of the data, the higher the performance of a DFS search.

**Longer time span analysis**

When you want to conduct searches over massive stretches of time to gather better insights into your data, DFS allows you to do that without impacting performance.

**Extensibility**

With federated search, you can join and search across multiple Splunk deployments seamlessly as if it was a single deployment.

**Role-based data isolation**

By using defined role capabilities, DFS can help you to ensure that data is not compromised across multiple deployments through restricted access to datasets and data sources.

DFS uses the Splunk DFS Manager app to automatically set up the compute cluster for your DFS deployment. To set up your compute cluster using Splunk DFS Manager, see Install Splunk DFS Manager.

**Big data analysis**

The big data analysis feature of DFS leverages cluster computing to process large datasets through the partitioned implementation of the search pipeline using a compute cluster like Apache Spark Core. DFS only uses the lowest primitives of Spark Core for task scheduling and does not use any higher level Spark libraries like Spark SQL. The DFS worker nodes of the compute cluster communicate with the Splunk Enterprise indexers and help to optimize search performance by distributing the processing load of the search among themselves.

**Search pipeline in a DFS deployment**

A DFS deployment comprises of a Splunk Enterprise search head, Splunk Enterprise indexers, and DFS workers in the compute cluster. The DFS master process and the DFS coordinator (DFC) are Java processes that run on the search head. The DFC process helps the DFS workers (DFWs) communicate with the search head. A DFS SPL search is received on the search head. The indexers send the pre-processed search data to the DFWs, which shuffle, compute and reduce the data to send the search results to the DFC. The DFC sends the results to the search process on the search head and displays the results in the UI. For more information on the DFM, DFC and the DFW processes, see DFS terminology.
A DFS search pipeline involves several stages of processing.

**Stage 1: On the search head:**

1. Receive a DFS-enabled search.
2. Define the sequence to run the various components of the search.
3. Set up a compute environment using a DFS compute cluster to run the sequence of operations to complete the search.

**Stage 2: On the indexers**

1. Process the remote portion of the search.
2. Send the intermediate results to the DFS compute cluster.

**Stage 3: On the DFS compute cluster**

1. Process the remaining steps in the run sequence of the search.
2. Send the intermediate results back to the search head.

**Stage 4: On the search head**

1. Apply relevant knowledge objects to the intermediate search results.
2. Send the final results of the search to the user interface.

The following diagram shows the data flow for a DFS search pipeline:
A DFS search is pretty similar to any other Splunk Enterprise SPL search, except for the prefix that is appended to the search in the form of the `dfsjob` command. If you place the `dfsjob` command at the start of the search, the search string is read from left to right and all the supported streaming commands, like `eval`, `fields` and generating commands, like `search` in the pipeline are processed on the indexers. However, when it encounters a transforming command like `stats` that orders the search results into a data table, DFS processes the remaining part of the search pipeline on the compute cluster.

The following query is an example of a big data analysis search:

```
| dfsjob [ search index=network | stats count by ip]
```

This is how a DFS search looks like if broken into the map and reduce phases:

```
| dfsjob [ | search sourcetype="websense::cg::kv"
| addinfo type=count label=prereport_events
| fields keepcolororder=t "bytes_in" "prestats_reserved_*" "psrsvd_*" "url"
| prestats mean(bytes_in) by url | ?
```

Map phase:

```
| search sourcetype="websense::cg::kv"
| addinfo type=prereport label=prereport_events
| fields keepcolororder=t "bytes_in" "prestats_reserved_*" "psrsvd_*" "url"
| prestats mean(bytes_in) by url | ?
```

Reduce phase:

```
| stats allnum=false delim=" " partitions=1 avg(bytes_in) AS avg_bytes_in BY url
| sort avg_bytes_in
| head limit=1000
```

Thus, the map phase of the DFS search is processed on the search peers or indexers. The reduce phase of the DFS search is processed on the DFS workers. The final reduce phase of the DFS search is performed on the search head.

In Splunk distributed search, the map phase of the search is processed on the search peers or indexers that locate the matching events and sort them into "field-value" pairs and send the result to the search head. The reduce phase of the search is performed by the search head, which processes the results based on the commands and aggregates them to return the final set of event results.

The DFS subssearch may only include commands supported by DFS. For more information on DFS supported commands, see Commands supported on DFS.

Federated search

Federated searches provide the ability to correlate searches across a wider data fabric to access relevant datasets and apply the search pipeline to the results in a distributed manner. User access to data on the federated providers is controlled through service accounts that define user roles and authorization levels. For more information on user roles, see Roles for federated searches.

A federated search deployment comprises of a federated search head and remote Splunk Enterprise deployments, that are also known as federated providers. A federated search head is a Splunk DFS instance that handles the search management functions and directs search requests to the remote federated providers.
A federated search works differently from a distributed model of Enterprise search. In Enterprise search, where you run a distributed search across Splunk deployments, your search heads must directly interact with the indexers of the other deployments. This requires you to bypass the security configuration and resource usage of the other deployments. Additionally, you will need to replicate all of the knowledge objects to provide a unified view across all search heads.

Federated search seamlessly searches across all deployments by leveraging the service account model that enables you to better manage the security configurations. You do not need to replicate the knowledge objects and you can correlate and run joins or unions to search across datasets in multiple deployments. Additionally, with a federated search you may run any Splunk SPL command on remote federated providers as part of pre-processing prior to running a federated search, even if the command is not supported by DFS.

The federated search pipeline operates in two stages:

1. The meta stage handles the pre-processing of the SPL search, where remote knowledge objects are leveraged to optimize the query and estimate the event count.
2. The search stage triggers the SPL search.

The following federated searches merge the results of disparate datasets from different geographical regions into one dataset with a single query:

```
| dfsjob | union [ | from federated:networkRemote1 | stats count by ip ] [ | from federated:networkRemote2 | stats count by ip ] [search index=networkLocal | stats count by ip | stats count

| dfsjob | union [ | from federated:buttercup_mobile_americas] [ | from federated:buttercup_mobile_asia pacific] [ | from federated:buttercup_mobile_europe] [ | from federated:buttercup_mobile_africa] [ | from federated:buttercup_mobile_middleeast] | stats sum(totalCount) by stats, clientip
```

For more information on federated search use cases, see Use cases for federated search. A federated search can run in two different modes based on the processing location of the intermediate search results (search data) and its impact on search performance and usability.

**Modes to run a federated search**

- **Sh mode**: In this mode, the intermediate search results or search data are sent from the search heads of the remote deployments to the DFS workers of the compute cluster. In this mode, the processing load on the search head may impact the search performance.

**Supported versions on sh mode:**

- Federated search head must be a Splunk Enterprise deployment of version of 7.2 or higher.
- Remote federated providers must be Splunk Enterprise deployments of version 7.2 or higher.

The following diagram shows the data flow for a federated search pipeline running in sh mode:
**Fabric mode:** In this mode, the intermediate search results from the indexers of the remote federated providers are sent directly to the DFS workers on the compute cluster. Running the search on the indexers of the remote deployment and reducing it on the DFS workers ensures optimal performance of a federated search at scale.

Standalone Splunk Enterprise deployments are not supported as remote providers in the fabric mode.

**Supported versions on fabric mode:**

- Federated search head must be a Splunk Enterprise deployment of version 8.0.
- Remote federated providers must be Splunk Enterprise deployments of version 8.0.

Distributed search groups are currently not supported in federated search.

The following diagram shows the data flow for a federated search pipeline running in fabric mode:
Benefits of running federated searches in fabric mode

You do not need to log in to the indexers of the remote federated provider, when you run federated searches in fabric mode. This can be useful when you inherit service accounts that do not allow you to access remote federated providers. Additionally, the federated searches are more optimized in the fabric mode because there is no processing bottleneck on the remote search head.

Commands supported by federated searches

The list of commands supported by federated search in the sh and fabric mode is different. Additionally, you must be aware of some caveats to running DFS and federated searches beyond understanding the list of supported commands in the two modes. For more information on the search caveats, see Search caveats.

Commands supported by the fabric mode

Use the following list to identify commands that are supported by federated searches in the fabric mode:

- dedup
- eval
- fields
- from
- head
- join
- rename
- reverse
- sort
- stats
Commands supported by the sh mode

Use the following list to identify commands that are supported by federated searches in the sh mode:

- chart
- dedup
- eval
- fields
- from
- head
- join
- regex
- reltime
- rename
- reverse
- search
- sort
- stats
- table
- tail
- union
- where

To run federated searches using a more extensive list of commands, use the sh mode.

Any Splunk SPL command may be run on remote federated providers as part of pre-processing prior to sending the results to the DFS workers for further processing. For a list of Enterprise Search SPL commands, see Search commands in the Splunk Enterprise Search Reference.

Set the mode for federated search

The version of the Splunk Enterprise deployment determines the mode used to run federated searches. Federated searches are run in fabric mode by default for federated search heads and remote search heads running version 8.0. Federated searches corresponding to older Splunk Enterprise deployments fallback to sh mode.

You may prefer to run federated searches in sh mode even if your Splunk Enterprise deployment is upgraded to version 8.0 in the following instances:

1. To restrict access to the indexers in the remote deployments
2. To use a more extensive list of commands supported by the sh mode
3. To avoid upgrading all search heads and indexers in your DFS deployment to version 8.0

You can switch the mode to run federated searches by specifying the `executionMode` setting to `sh` or `fabric` in the `federated.conf` configuration file. Contact your account team for assistance because there may be some performance implications when you switch modes.

For information on the compatibility matrix to run federated searches and modes, see Compatibility matrix.
DFS terminology

Splunk Data Fabric Search (DFS) is a new search platform that leverages the distributed processing power of an external compute engine to broaden the scope and capability of the Splunk Platform.

DFS has the following features:

- **Big Data Analysis**
  Analyzes and explores large amounts of data to search over a billion events within a single Splunk deployment with significant performance improvements.

- **Federated Search**
  Conducts searches and joins across multiple indexes on disparate Splunk deployments as seamlessly as if it was a single deployment.

To understand DFS, you must know the following terms:

**DFS compute cluster**
DFS uses cluster computing to process large datasets through a partitioned implementation of the search pipeline. DFS supports compute clusters using a distributed computing framework like Apache Spark Core.

**DFS coordinator (DFC)**
A Java management process started on the search head for each DFS search to manage the different stages of a single DFS search. This process exits once the DFS Search is complete. DFC partitions the search so that different parts of the search and computation is run on the indexer, compute cluster, and the search head.

**DFS master (DFM)**
A Java management process located on the search head that is initiated by the `splunkd` service and contains all the port and resource usage information for DFS. A single DFM process is required for each instance of Splunk Enterprise.

**DFS worker (DFW)**
Refers to the worker nodes of the compute cluster that communicate with the Splunk indexers and run the job on the compute cluster. DFWs help to optimize search performance by distributing the processing load of the search among themselves. DFWs are launched by compute cluster worker nodes when a DFC process is initiated. A DFC process creates the same number of DFWs as the number of nodes in the compute cluster.

**Federated provider**
Refers to the remote deployments that are leveraged in a federated search to correlate datasets and searches across a wider data fabric. Splunk Enterprise deployments are currently the only supported federated providers.

**Federated search head**
A Splunk DFS instance that handles the search management functions and directs search requests to remote Splunk Enterprise deployments.

**Cardinality**
Refers to the uniqueness of data values for any given field contained in a dataset. For example, identification numbers, email addresses, user names, or USER_ID. Low cardinality refers to values that are very similar, which means that the dataset contains a lot of duplicate or repeat values. For example, status flags, boolean values, or gender.
Service account

Accounts that define user roles and authorization levels to secure access to data within DFS and federated search deployments.

Virtual CPU (vCPU)

One or more vCPUs or virtual central processing units can be assigned to every Virtual Machine (VM). Each vCPU is seen as a single physical CPU core by the VM's operating system. The free version of DFS is licensed to have access to a number of vCPUs based on the daily ingestion capacity of the corresponding Splunk Enterprise license. The Splunk DFS Manager app logs the vCPU usage for DFS searches.

Use cases for big data analysis

The following examples demonstrate how to use Splunk Data Fabric Search (DFS) to perform searches over massive volumes of enterprise data, which makes it easier to gather faster insights and take timely action. Additionally, you can run high performance searches over high cardinality datasets (where fields have unique or uncommon values). DFS searches may take longer to start than Enterprise search. However, once the DFS search is processed and starts running, it scales through an astronomical number of events very fast. These examples assume that you have applied the DFS license, set up a DFS compute cluster, and configured and secured DFS. A free version of DFS is included with Splunk Enterprise 8.0, or later, term licenses with at least 1 TB of daily ingestion capacity.

Combining datasets over different timespans

As a security analyst, you want to quickly identify the overlapping IP address between two different intervals of time to investigate a threat. This example illustrates how you can use DFS to quickly report the overlapping IP addresses between two separate one hour intervals of time.

| dfsjob | union | search index=<source_indextype1> earliest=-2h latest=-1h | eval marker=1 | stats count by marker, clientip | search index=<source_indextype2> earliest=-1h latest=now | eval marker=2 | stats count by marker, clientip | stats dc(marker) as dc_marker by clientip | where dc_marker > 1 stats count

Using a DFS search helps to scale the same Enterprise SPL search over billions of records in disparate datasets.

Searches over massive volumes of data

As a business analyst in a technology company, you want to use DFS to optimize the search performance and quickly generate a weekly report on web traffic. This example illustrates how you can aggregate statistics using fields. You can run a DFS search over a billion events and generate a report in minutes that normally takes hours to complete, giving you faster insight into your data.

Use the dfsjob command to run the search as follows:

| dfsjob | search index=idx_oar | stats count by ipCity, ipCompany, ipMasked | stats count |

Instead of the following SPL search:

index=idx_oar | stats count by ipCity, ipCompany, ipMasked | stats count
**Expand analysis to longer timeframes**

As a site reliability engineer at a consumer technology company, you want to use DFS to run reports for root cause analysis. This example illustrates how to gather insights from large datasets that range up to a billion events and span over multiple hours or days of the week without splitting the search into multiple jobs by time range and aggregating the output of multiple jobs.

Use the `dfsjob` command to run a DFS search as follows to help analyze network traffic issues within your network in a timely manner:

```bash
| dfsjob [search index= idx_network | stats avg(bytes_in) avg(bytes_out) avg(packet_in) avg(packet_out) by ip,port,protocol]
```

Instead of the following SPL search:

```spl
index = idx_network | stats avg(bytes_in), avg(bytes_out), avg(packet_in), avg(packet_out) by ip,port,protocol
```

**Searches over high cardinality data**

As a business analyst in a technology company, you want to use DFS to easily evaluate recurring customer issues, identify the duration of an average customer call to improve customer retention rates, and enhance customer conversion to your product offerings. You can search across the call data records (CDR) and other transaction records that can have tens of millions of subscriber ID fields or text messages with extremely unique values to gather greater visibility into your data. This example illustrates how you can leverage DFS to quickly run searches on high cardinality datasets.

Use the `dfsjob` command to run the search as follows:

```bash
| dfsjob [search index= cdr | stats sum(Duration) sum(SalesPrice) by CLI]
```

where,

- `cdr` = Call Data Record
- `CLI` = Caller Line Identifier or Subscriber ID

Instead of the following SPL search:

```spl
index=cdr | stats sum(Duration) sum(SalesPrice) by CLI
```

**Use cases for federated searches**

The following examples demonstrate how DFS federated search capability seamlessly allows you to search across multiple Splunk Enterprise deployments while maintaining user access permissions, regulating system resources, and leveraging existing knowledge objects for faster data processing. Additionally, the federated search capability allows you to have more control over your data architecture and store datasets in multiple Splunk Enterprise deployments. These examples assume that you have configured and secured DFS, installed the license, and set up user accounts with federated search capabilities.
Join data across disparate datasets

As a site reliability engineer at a media technology company, you want to schedule a report to identify temporary access errors by correlating data from two separate Splunk Enterprise deployments. Using federated search, you can seamlessly combine massive "join" operations on two separate data sources with a single query.

Use the `dfsjob` command to run the search as follows:

```
| dfsjob [search index= idx_logs | stats count by reqid | join type=inner usetime=false left=L right=R where L.reqid = R.reqid ] [| from federated: deployment_backend_logs]
```

Search across multiple Splunk Enterprise deployments

As a business analyst in a telecommunications company, you may want to generate a report by conducting a federated search across multiple Splunk Enterprise deployments that contain subscriber data from different silos in the business like video, wireless, and Internet. This example illustrates how to use federated search to identify the video, mobile, and Internet customers that were affected by an outage. Using distributed search across these deployments can potentially bypass user access permissions and overload resources on other deployments. However, a federated search allows you to perform a search across multiple Splunk Enterprise deployments through a single search request without replicating data across multiple deployments and maintain user access permissions.

Use the `dfsjob` command to run a federated search across three Splunk Enterprise deployments as follows:

```
|dfsjob [ union [| from federated: buttercup_mobile] [| from federated: buttercup_video][| from federated: buttercup_internet] | stats sum(totalCount) by status, clientip]
```
Set up DFS

Installation quickstart

You need a Splunk Enterprise search license to use DFS. If you have at least 1 TB of daily ingestion capacity using Splunk Enterprise, you can run a DFS search for free. For more information on the free vCPUs to run DFS searches, see Licensing information. To identify your current ingestion capacity, navigate to the Search and Reporting app, click on Settings. Under System, click on Licensing. The licensing page indicates the daily ingestion capacity provided by your specific license.

DFS uses the Splunk DFS Manager app to automatically set up the compute cluster for your DFS deployment. If you use the app to deploy the compute cluster to run DFS searches, you can leverage the environment variables and some configuration settings that are packaged with the app. Additionally, DFS workers can be configured using the app.

If you are using the DFS Manager app to configure your compute cluster, you must configure the security settings on the DFS workers similar to the other indexers (distributed search peers). For more information on configuring security settings, see Secure a DFS deployment.

To download the app from Splunkbase, see Splunk DFS Manager. To set up your compute cluster using Splunk DFS Manager, see Splunk DFS Manager User Manual. For installation instructions on the supported deployments, see Install Splunk DFS Manager.

Splunk only provides support to enable or maintain a compute cluster that is deployed using the Splunk DFS Manager app. If you bring your own compute cluster, Splunk is not responsible for support or maintenance of the compute cluster.

DFS does not support Splunk multi-site deployment architecture.

Use the following flowchart to guide you through the basic deployment process for DFS using the Splunk DFS Manager
Use the following table to verify that you meet all the requirements to deploy DFS:

<table>
<thead>
<tr>
<th>Number</th>
<th>Task</th>
<th>Description</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Install Splunk Enterprise</td>
<td>Set up search heads and indexers.</td>
<td>Install Splunk Enterprise</td>
</tr>
<tr>
<td>2</td>
<td>Meet hardware requirements for DFS</td>
<td>Verify hardware requirements for DFS.</td>
<td>Hardware requirements for DFS</td>
</tr>
<tr>
<td>3</td>
<td>Check compatibility matrix</td>
<td>Use the compatibility matrix to identify supported versions for Splunk Enterprise, compute cluster, and Splunk DFS Manager app for your deployment.</td>
<td>Compatibility matrix</td>
</tr>
<tr>
<td>4</td>
<td>Meet upgrade requirements for DFS</td>
<td>Verify upgrade requirements for Splunk Enterprise to run DFS.</td>
<td>Upgrade requirements</td>
</tr>
<tr>
<td>5</td>
<td>Review the release notes</td>
<td>Review the list of new features, known issues, and fixed issues.</td>
<td>New Features</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Known issues</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Fixed issues</td>
</tr>
<tr>
<td>6</td>
<td>Meet port requirements</td>
<td>Ensure ports are available to set up the compute cluster.</td>
<td>Port requirements</td>
</tr>
<tr>
<td>7</td>
<td>Edit configuration files for DFS</td>
<td>Some settings are automatically configured using the Splunk DFS Manager app. However, a few settings must be manually configured.</td>
<td>Edit configuration files for DFS</td>
</tr>
</tbody>
</table>

For more information on compatible issues with DFS, see the Compatibility matrix in the Data Fabric Search manual.
Licensing information

A free basic edition of Splunk Data Fabric Search (DFS) is included with Splunk Enterprise 8.0, or later, term licenses with at least 1 TB of daily ingestion capacity. The free version of DFS is licensed to have access to a number of virtual central processing units or vCPUs based on the daily ingestion capacity of the corresponding Splunk Enterprise license.

One or more vCPUs can be assigned to every virtual machine (VM). Each vCPU is seen as a single physical CPU core by the VM's operating system. To identify your current ingestion capacity, navigate to the Search and Reporting app, click on Settings. Under System, click on Licensing. The licensing page indicates the daily ingestion capacity provided by your specific license.

Use the following table to identify the free vCPUs that you can use to run DFS searches based on your daily data ingestion capacity:

<table>
<thead>
<tr>
<th>Daily TB data ingestion</th>
<th>Free vCPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 1TB</td>
<td>0</td>
</tr>
<tr>
<td>1TB - 2TB</td>
<td>32</td>
</tr>
<tr>
<td>2TB - 5TB</td>
<td>48</td>
</tr>
<tr>
<td>5TB - 10TB</td>
<td>64</td>
</tr>
<tr>
<td>10TB - 20TB</td>
<td>80</td>
</tr>
<tr>
<td>20TB - 50TB</td>
<td>96</td>
</tr>
<tr>
<td>50TB+</td>
<td>112</td>
</tr>
</tbody>
</table>

The Splunk DFS Manager app logs the vCPU usage for DFS searches. To learn more about pricing for additional vCPUs for DFS, please contact your account representative. Customers can use the free version of DFS with the included free vCPU entitlement or purchase a license for additional vCPUs to increase the total number of vCPUs to which DFS has access. If vCPU usage for DFS searches exceeds the licensed limit, a warning message is displayed as follows: "Your search exceeds allocated vCPU. Please contact our account team to buy additional vCPUs." However, the search is not stopped if the vCPU limit is exceeded.

The free basic version of Splunk DFS is only available to customers on a Splunk Enterprise term license. Customers with Splunk Enterprise perpetual licenses are not entitled to use the free version of Splunk DFS.

The licensing model for Splunk DFS edition is based on vCPUs purchased in addition to free vCPU's. The Splunk DFS edition is available for purchase to every customer with Splunk Enterprise version 8.0 or higher with a term license. The Splunk DFS edition license is ideally suited to customers who want to use DFS for large scale big data analysis and federated searches across Splunk Enterprise deployments.

Hardware requirements

To run DFS searches, you must install the Splunk DFS Manager app to configure your compute cluster automatically. For more information on the Splunk DFS Manager app, see Splunk DFS Manager Overview.

Plan your hardware capacity before deploying DFS. You can use the Splunk DFS Manager app to monitor core and memory usage when running DFS searches. However, contact your account representative to explore and identify the optimal vCPU limit for your specific DFS deployment. For more information on licensing, see Licensing.
Use the following table for guidelines on the hardware requirements to deploy DFS:

<table>
<thead>
<tr>
<th>Hardware</th>
<th>General specifications</th>
<th>Minimum specifications</th>
<th>Optimal specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory</td>
<td>Ratio of 1 vCPU to 8 GB or higher for each node in the compute cluster</td>
<td>64 GB</td>
<td>128 GB</td>
</tr>
<tr>
<td>vCPU</td>
<td>8-16 vCPU on each node in the compute cluster</td>
<td>8</td>
<td>16</td>
</tr>
<tr>
<td>Network speed</td>
<td>10 Gb/s or higher</td>
<td>10 Gb/s</td>
<td>10 Gb/s</td>
</tr>
<tr>
<td>Local disk storage</td>
<td>500 GB for each node</td>
<td>500 GB or more (1200+ IOPS) for each node</td>
<td>500 GB or more (SSD) for each node</td>
</tr>
</tbody>
</table>

For optimal IOPS and performance, use SSD hard drives but it is not required.

**Compatibility matrix**

The tables in this topic list versions of Data Fabric Search (DFS) and compatible Splunk Enterprise versions, Splunk products and apps versions, operating system versions, and compute cluster versions for customer-deployed environments.

You must install and configure the Splunk DFS Manager app to deploy your compute cluster automatically. Apache Spark is currently the only supported DFS compute cluster.

<table>
<thead>
<tr>
<th>Splunk Enterprise</th>
<th>Data Fabric Search version</th>
<th>Splunk DFS Manager app version</th>
<th>Compute cluster (Spark) version</th>
<th>Operating system version</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.3.x</td>
<td>1.0.0 Limited Availability Release (LAR)</td>
<td>Not applicable</td>
<td>Spark 2.3.0</td>
<td>Linux x86-64 architecture</td>
</tr>
<tr>
<td>8.0.x</td>
<td>1.1.0 GA Release</td>
<td>1.0 and higher</td>
<td>Spark 2.3.3</td>
<td>Linux x86-64 architecture</td>
</tr>
</tbody>
</table>

For the support status of Splunk Enterprise, see the Splunk Software Support Policy. For information on Splunk Enterprise upgrade requirements to use DFS, see Upgrade requirements.

DFS also supports backward compatibility with some Splunk Enterprise versions for DFS deployments that run federated searches. Use the following table to identify the compatible Splunk Enterprise versions that can be used as federated search heads and remote federated providers. The table also provides guidance on how the Splunk Enterprise versions you use in your DFS deployment impacts the mode for running federated searches:

<table>
<thead>
<tr>
<th>Federated search mode</th>
<th>Federated search head Splunk Enterprise version</th>
<th>Remote federated provider Splunk Enterprise version</th>
<th>Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fabric mode (Default)</td>
<td>8.0.0</td>
<td>8.0.0</td>
<td>1. Search at scale across multiple remote federated providers 2. Indexers distribute data to the DFS workers</td>
</tr>
<tr>
<td>Search head mode</td>
<td>7.3 or higher</td>
<td>7.2 or higher</td>
<td>1. Perform complex correlations on data across remote</td>
</tr>
</tbody>
</table>

17
The mode to run federated searches is determined by the lowest version of the search head and indexers on a deployment. If the federated search head and the remote search heads in your deployment are all running on version 8.0, federated searches will automatically run in the fabric mode. If the remote search head runs on version 7.3 and the federated search head runs on version 8.0, the federated search runs using the sh mode. Additionally, federated searches fail to run when the remote deployment contains search heads and indexers running on different versions of Splunk Enterprise software. For example, if the search head of the remote federated provider runs on version 8.0 and the indexer on the remote federated provider runs on version 7.3, the mode to run federated searches must be manually set to sh using the configuration files. Contact Splunk Services to switch the mode for federated searches in your deployment. For more information on setting the mode to run federated searches, see Set the mode for federated searches.

DFS currently supports only a limited set of Enterprise search commands that makes it incompatible with Enterprise Search app.

## Upgrade requirements

Configure the following upgrade requirements to run an optimal data fabric search (DFS) and federated search:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Upgrade requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Big data analysis</td>
<td>Upgrade both search heads and indexers to the latest Splunk Enterprise version 7.3 or higher.</td>
</tr>
<tr>
<td>Federated search</td>
<td>Upgrade the federated search head to the latest Splunk Enterprise version 7.3. Upgrade the federated providers to Splunk Enterprise version 7.2 or higher.</td>
</tr>
</tbody>
</table>

If you are using the Splunk DFS Manager app to install the compute cluster, upgrade both search heads and indexers to Splunk Enterprise version 8.0.x.

For information on hardware requirements, see **Hardware requirements for DFS**.

## Port configuration requirements

To deploy DFS, you must ensure that certain ports are available so that the Splunk Enterprise deployment may interact with your compute cluster and run DFS searches.

Only the DFS master port is an internal port. All other ports are external and enable communication between the search head to the indexer, the compute cluster master to the DFS worker, or between the DFS workers. Only the DFS search head and the DFS workers may access the compute cluster ports. You can block all other nodes from accessing the compute cluster master port and the compute cluster UI port.

You can configure the ports using configuration files like `limits.conf` or `server.conf`. For more information on configuring ports using the configuration files, see **Edit configuration files for DFS**.

You can also configure some of the ports using the UI. For more information on configuring ports using the Splunk DFS Manager app UI, see **Change port settings**.
Ensure that the following ports on your compute cluster are available to enable firewall access to run data fabric searches (DFS):

<table>
<thead>
<tr>
<th>Port name</th>
<th>Port location</th>
<th>Default value</th>
<th>Recommended values</th>
<th>Description</th>
<th>Restart information</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFS master port</td>
<td></td>
<td>9000</td>
<td>9000</td>
<td>Used internally to communicate with splunkd and the compute cluster master.</td>
<td>Restart splunkd.</td>
</tr>
<tr>
<td>DFC coordinator port or DFS search control port</td>
<td></td>
<td>17000</td>
<td>17000 + maximum number of concurrent searches</td>
<td>Sets the listening port for the DFC coordinator process started on each search head for each DFS search to manage the different stages of the DFS search using the compute cluster.</td>
<td>Restart splunkd.</td>
</tr>
<tr>
<td>Compute cluster port</td>
<td></td>
<td>8008</td>
<td>8008</td>
<td>Communicates with the compute cluster master and the DFS workers.</td>
<td>Restart compute cluster.</td>
</tr>
<tr>
<td>Compute cluster WebUI port</td>
<td></td>
<td>8009</td>
<td>8009</td>
<td>Used to get statistics and resource information.</td>
<td>Restart compute cluster.</td>
</tr>
<tr>
<td>Spark history server port</td>
<td></td>
<td>8010</td>
<td>8010</td>
<td>Used to get statistics about compute cluster jobs corresponding to DFS jobs.</td>
<td>Restart Spark.</td>
</tr>
<tr>
<td>DFS worker ports or Receiving data ports</td>
<td></td>
<td>17500+</td>
<td>1750 - 17549</td>
<td>Communicates with Splunk indexers to run the search job on the compute cluster.</td>
<td>Restart splunkd.</td>
</tr>
</tbody>
</table>

The following diagram shows port requirements for DFS:
Configure the compute cluster using the app

Use the Splunk DFS Manager app to set up your compute cluster. The app bundles your compute cluster and configures it for your DFS deployment automatically. You can use the app irrespective of your deployment scenario to deploy the compute cluster on a standalone search head, search head cluster, or an indexer cluster. For more information on installing the Splunk DFS Manager app, see Install Splunk DFS Manager.

Splunk only provides support to enable or maintain a compute cluster that is deployed using the Splunk DFS Manager app. If you configure your own compute cluster, Splunk is not responsible for the support or maintenance of the compute cluster.

If you are using the DFS Manager app to configure your compute cluster, you must configure the security settings on the DFS workers similar to the other indexers (distributed search peers). For more information on configuring security settings, see Secure a DFS deployment.

Set up a DFS deployment

Prerequisites

- You must have a minimum of one Splunk Enterprise deployment.
- Ensure that you meet all upgrade requirements. For more information on upgrade requirements for DFS, see Upgrade requirements.
- Identify the free vCPUs that you can use to run DFS searches based on your daily data ingestion capacity. For more information, see Licensing information.
- Edit configuration settings in the server.conf and the limits.conf configuration files. For more information on editing configuration files to set up big data analysis, see Edit configuration files for big data analysis.
Steps

1. Install Splunk Enterprise. For more information, see Install Splunk Enterprise in the Splunk Enterprise Search Tutorial manual.
2. Install Splunk Enterprise on the indexers that you plan to use as DFS workers.
   - Use a dedicated indexer or indexer cluster to set up DFS workers.
3. Install Splunk DFS Manager app on the search heads and indexers. For more information on installing the app, see Install Splunk DFS Manager.
4. Configure Splunk DFS Manager app. For more information on the settings that can be configured using the app, see Splunk DFS Manager app homepage and Change compute cluster settings.
5. Add workers using the Splunk DFS Manager app. For more information on adding workers, see Add workers.
6. Run a DFS search. For more information on running a DFS search, see Run DFS searches.

Run DFS searches

If your Splunk platform deployment has Data Fabric Search (DFS) functionality enabled, you can use DFS to perform searches over extremely large volumes of high cardinality data. You can also use DFS to run federated searches over one or more additional Splunk platform deployments. However, DFS processing only takes place when you use Search Processing Language (SPL) to invoke the DFS functionality.

You can construct a DFS search by including the dfsjob command as a wrapper around the SPL search. Run DFS searches using the search bar on Splunk Web.

Use dfsjob to run a DFS search

Place the dfsjob command at the start of a search. The dfsjob command reads the search string left to right and processes all supported streaming or generating commands in the search pipeline that runs on the indexers. When it encounters a transforming command, like stats in the search pipeline, DFS processes the remaining part of the search pipeline on the compute cluster. When the DFS process reaches the end of the SPL search, it returns the search results.

If the DFS process encounters an unsupported command, it returns an error message.

A DFS search is composed of three phases:

```
| dfsjob [ | <map-phase> | <reduce-phase> ] | <sh-phase>
```

The following DFS search can be broken down into the map phase, reduce phase, and a final reduce phase as follows:

```
| dfsjob [ | search sourcetype="websense::cg::kv" | stats avg(bytes_in) as avg_bytes_in by url | sort – avg_bytes_in | head 1000]
```

The map phase is processed on the indexers as follows:
The reduce phase is processed by the DFS workers as follows:

```bash
| search sourcetype="websense::cg::kv" | addinfo type=count label=prereport_events | fields keepcolorder=true "bytes_in" "prestats_reserved_*" "psrsvd_*" "url" | prestats mean(bytes_in) by url | ...
```

The final reduce phase or `<ah-phase>` is processed on the search head.

For more information on the `dfsjob` command, see [dfsjob](#).

### Supported commands on DFS

As a first generation product, DFS 1.x does not yet support all the Splunk Enterprise SPL commands. The current list of DFS supported commands are built and developed to handle scale and performance for processing large amounts of data. Support for more commands will be added in future versions.

Use the following list to identify commands that are supported by DFS along with information on their usage with DFS:

- `dedup`
- `eval`
- `fields`
- `from`
- `head`
- `join`
- `rename`
- `reverse`
- `sort`
- `stats`
- `tail`
- `union`
- `where`

Some of the commands supported by DFS operate differently from standard SPL search commands. For information on search caveats, see [DFS search caveats](#). Additionally, refer to the relevant topics on the DFS supported commands for more information.

### Set up service accounts for access control

Set up service accounts and define user roles and authorization levels to secure access to data within your DFS deployments.

You need to set up service accounts to run federated searches only. You do not need to set up service accounts to run big data analysis searches.

For federated search deployments, you must additionally set up service accounts on the federated providers to limit access to data. For more information on using access control to secure data, see [Use access controls to secure data](#). For more information on configuring role based user access, see [Add users and roles](#).
Set up users with federated search capabilities using Splunk Web

The federated search administrator can create service accounts with usernames and passwords on the federated providers. The administrator also specifies the set of indexers and other resources that the users on the federated providers can access from the remote deployment. Ensure that the username and password on the remote federated providers match the local Splunk user accounts. For more information on roles that can run and manage federated searches, see Roles to run federated searches.

On the federated search head:

Add a new role fsh_admin with admin inheritance and fsh_search and fsh_manage capabilities. For more information on adding and editing roles with Splunk Web, see Add and edit roles. For more information on role inheritance, see Role inheritance in the About role-based user access topic for more information. For more information on capabilities, see About defining roles with capabilities for more information.

On the federated providers:

Create users on the federated providers and assign them user roles. For more information on how to create and edit users, see Add and edit users. You can associate federated providers with user roles only by editing configuration files. This functionality is currently not available in the UI. For information on associating the federated providers with user roles, see Set up a federated search.

Roles to run federated searches

The following table identifies the various capabilities assigned to roles that run and manage federated searches:

<table>
<thead>
<tr>
<th>Roles</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fsh_search</td>
<td>User role authorized to:</td>
</tr>
<tr>
<td></td>
<td>• define federated searches in the savedsearches.conf file</td>
</tr>
<tr>
<td></td>
<td>• run a federated search using remote Splunk Enterprise deployments</td>
</tr>
<tr>
<td></td>
<td>• retrieve credentials associated with a service account on a remote</td>
</tr>
<tr>
<td></td>
<td>deployment</td>
</tr>
<tr>
<td></td>
<td>Enabled by default for all administrator roles.</td>
</tr>
<tr>
<td>fsh_manage</td>
<td>Administrator role authorized to manage federated search settings:</td>
</tr>
<tr>
<td></td>
<td>• add remote Splunk Enterprise deployments to the federated.conf file</td>
</tr>
<tr>
<td></td>
<td>• create, edit, or delete federated deployment entries</td>
</tr>
<tr>
<td></td>
<td>• add and manage users that are allowed to run federated searches</td>
</tr>
</tbody>
</table>

Set up a federated search

Set up a federated search using Splunk Web, which is the recommended method. Alternatively, you can use REST API endpoints or edit configuration files to set up federated searches.

- Set up federated searches using Splunk Web
- Set up federated searches using REST API endpoints
- Set up federated searches using configuration files
You may choose to change the mode to run federated searches. However, note that this may impact the search performance and usability. Therefore, only change the mode for your federated search after contacting your account representative. For more information on changing the mode, see Set the mode for the federated search.

Set `tls_enabled=true` in the `[dfs_security]` stanza of the `server.conf` configuration file to enable security on the indexers of the remote federated providers. You may disable this setting to optimize performance.

**Set up federated searches using Splunk Web**

Use Splunk Web to set up a federated search because it automatically updates all the configuration files with the information required to run the search. Perform the following tasks to set up a federated search using the Splunk Web UI.

1. Add federated providers
2. Add federated providers to user roles
3. Create a federated search
4. Edit a federated search
5. Edit permissions for a federated search

**Add federated providers**

Add federated providers to access relevant datasets from multiple remote Splunk Enterprise deployments and conduct more insightful searches. Running a federated search may result in errors if you add federated providers on a search head that is not configured to be a federated search head.

**Prerequisites**

- Enable DFS.
- Add the `fsh_search` and the `fsh_manage` capabilities to the role associated with the federated provider.

**Steps**

1. In the Splunk Search and Reporting app, click on Settings. The Settings menu lists the configuration pages for Knowledge objects, System and licensing, Data, Users, Authentication, and Distributed environment settings.
2. Under Distributed environment, click on Data Fabric. The Data Fabric Search page displays the federated providers that are currently connected to the DFS configuration and includes information on their names, types, user accounts, IP addresses, and roles. Additionally, you can filter the federated providers and edit or remove federated providers from the DFS configuration. If you do not have existing federated providers in your configuration, no federated providers are listed in the Data Fabric Search page.
   For information on editing federated providers, see Edit federated providers.
   For information on deleting federated providers, see Delete federated providers.
3. In the Data Fabric Search page, click on Add federated provider.
4. In the Add Federated Provider page, add information on the Splunk Enterprise deployment to which you want to connect.
   Information on the federated provider includes the following items:
   - IP address of the search head
   - User account
   - Account password
   - Applications that you may access
   - Roles that you may assign to the user to run federated searches
An incorrect IP address or port number throws an error when you run the federated search.

Only roles that have the `fsh_manage` and the `fsh_search` capabilities are displayed in the list. If the role you want to associate with the federated provider is not displayed, then first create the role or add the federated search capability to an existing role to display the role in the list. For more information on user roles, see Roles to run federated searches.

You will not be able to add the federated provider to your configuration, if you do not select a role. Additionally, the name, user account, and role fields for the federated provider cannot be empty and will generate an error message.

5. Click **Add**.

**Edit federated providers**

Edit roles, IP addresses, passwords, or applications associated with the federated providers that are included in the DFS configuration.

**Prerequisites**

- Ensure that you have `fsh_manage` capabilities.
- Enable DFS.

**Steps**

1. In the Splunk Search and Reporting app, click on **Settings**.
   The **Settings** menu lists the configuration pages for Knowledge objects, System and licensing, Data, Users, Authentication, and Distributed environment settings.
2. Under **Distributed environment**, click on **Data Fabric**.
   The **Data Fabric Search** page displays the federated providers that are currently connected to the DFS configuration and includes information on their user accounts, IP addresses, roles.
3. In the **Data Fabric Search** page, click on the **Edit** button next to the federated provider to which you want to make changes.
   The **Data Fabric Search** page displays the federated providers that are currently connected to the DFS configuration and includes information on their names, types, user accounts, IP addresses, and roles.
4. In the **Edit Federated Provider** page, edit information on the Splunk Enterprise deployment for which you want to make changes.
   Information that you can edit for the federated provider includes the IP address, password, the applications that you may access, and the roles that you may assign to the user to run federated searches.
5. Click **Save**.

**Delete federated providers**

Delete federated providers from the DFS configuration if you do not want to access data from a specific Splunk Enterprise deployment.

**Prerequisites**

- Ensure that you have `fsh_manage` capabilities.
- Enable DFS.

**Steps**

25
1. In the Splunk Search and Reporting app, click on Settings. The Settings menu lists the configuration pages for Knowledge objects, System and licensing, Data, Users, Authentication, and Distributed environment settings.

2. Under Distributed environment, click on Data Fabric.

3. In the Data Fabric Search page, identify the Splunk Enterprise deployment that you want to remove from the DFS configuration.

The Data Fabric Search page displays the federated providers that are currently connected to the DFS configuration and includes information on their names, types, user accounts, IP addresses, and roles.

4. Click on the Delete button next to the Splunk Enterprise that you want to remove.

5. Click Delete again to confirm the removal of the federated provider from the DFS configuration.

Deleting a federated provider may result in the federated searches that were created using that federated provider to not work.

**Add federated providers to user roles**

Add federated providers to user roles in your DFS configuration to enable specific access permissions for each user when they access the federated provider.

**Prerequisites**

- Ensure that you have fsh_manage capabilities.
- Enable DFS.

Only additional roles can be added using this procedure.

**Steps**

1. In the Splunk Search and Reporting app, click Settings > Users and Authentication > Roles. The Roles page displays the list of roles by name, capabilities, and access to applications.

2. On the Roles page, click on New Role.

3. On the New Roles page, type the name of the new role that you want to create.

4. Click on Resources and scroll down to Federated providers to add federated providers to the user role.

5. Click Save.

To verify that the federated provider is added to the role, go to the Add Federated Provider page and check the list for the roles assigned to the federated provider. The new role is displayed next to the federated provider in the Role column.

If you want to add federated providers to a role, the role must have fsh_search capabilities.

**Create a federated search**

Create a new federated search by performing the following tasks:

**Steps**

1. In the Search and Reporting app, click on Settings > Search, reports, and alerts.

2. In the Search, Reports, and Alerts page, click on New Federated Search.

3. In the Create Federated Search page, add information for the following fields:

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
</table>

26
DFS only supports transforming searches for remote deployments currently. For more information on commands supported by federated searches in sh mode, see Commands supported by the sh mode. For more information on commands supported by federated searches in fabric mode, see Commands supported by the fabric mode. Additionally only transforming searches are supported. For more information on transforming searches, see Only transforming searches are supported.

**Edit a federated search**

Edit a federated search to make changes to the description of the federated search, the SPL search syntax, or the federated providers that can be accessed by the federated search.

**Steps**

1. In the Search and Reporting app, click on **Settings > Search, reports, and alerts**.
2. In the **Search, Reports, and Alerts** page, click **Edit > Edit search** next to the saved federated search that you want to modify.
3. In the **Edit Federated Search** page, edit information for the following fields:

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description (optional)</td>
<td>Information about the search</td>
</tr>
<tr>
<td>Search</td>
<td>Syntax of the SPL search</td>
</tr>
<tr>
<td>Federated Provider</td>
<td>Remote Splunk Enterprise deployment that you access to run the federated search</td>
</tr>
</tbody>
</table>

4. Click **Save**.

**Edit permissions for a federated search**

Edit permissions to run a federated search and control access to the federated providers by making changes to the read or write capabilities for different user roles.

**Steps**

1. In the Search and Reporting app, click on **Settings > Search, reports, and alerts**.
2. In the **Search, Reports, and Alerts** page, click **Edit** next to the saved federated search that you want to modify.
3. In the **Edit Permissions** page, you can select read or write permissions for applications running on the federated providers.
4. Click **Save**.
**Delete a federated search**

Delete a federated search by performing the following tasks:

**Steps**

1. In the Search and Reporting app, click on Settings > Search, reports, and alerts.
2. In the Search, Reports, and Alerts page, click **Delete** next to the saved federated search that you want to delete.
3. Click **Delete** to confirm deleting the federated search.

**Set up federated searches using REST API endpoints**

Use REST API endpoints to specify information on the federated providers, set up the federated search, and create user roles authorized to run federated searches.

**Prerequisites**

- One federated search head
- One indexer
- One compute cluster master
- One DFS worker
- One federated provider deployment that comprises one search head and one indexer
- Deploy DFS on the federated search head
- Authorize the administrator role to manage federated search settings

For information on infrastructure requirements, see [Infrastructure requirements for DFS](#).

For information on the hardware specifications to set up Spark, see [Hardware requirements](#).

For information on port configuration requirements for Spark, see [Port configuration requirements](#).

The DFS compute cluster must be installed and configured prior to configuring a federated search deployment.

As an authorized user, you can run federated searches across multiple Splunk Enterprise deployments, which are also known as federated providers. For information on setting up user accounts and user roles for a federated search, see [Set up user accounts for access control](#).

All federated searches are transforming searches. Role-based user access control is enforced when you run federated searches.

**Steps**

1. Log in to the federated search head.
   
   ```bash
   source <path to splunk>/bin/setSplunkEnv
   ```
   
   or
   
   ```bash
   cd $SPLUNK_HOME/bin
   ```

2. Review that DFS is enabled on the federated search head by running a DFS search on the search bar of Splunk Web. For example, you can run the following SPL search:

   ```bash
   | dfsjob [ search index-network | stats count by ip]
   ```
If the DFS search runs successfully, DFS is enabled on the federated search head.

If the DFS search does not run, validate your DFS search setup. For more information on setting up a DFS search, see Set up a Data Fabric Search.

3. Install licenses for the federated search on the federated search head. For information on installing the license, see Install DFS license.

4. Create a custom federrated.conf configuration file at the following location: $SPLUNK_HOME/etc/system/local/.

5. Specify the federated providers using the following command:

   ./splunkd rest --auth=fsh_admin:mychanged1 POST /services/dfs/federated
   "name=remote_splunk_deployment_0&ip=10.224.6.219&splunk.port=8089&splunk.serviceAccount=remotefshuser"

   The following sample stanza appears in the federated.conf file:

   [remote_splunk_deployment_0]
   ip = 10.224.6.219
   splunk.port = 8089
   splunk.serviceAccount = remotefshuser

   For more information on using the federated.conf endpoint to create and maintain the DFS federated providers, see /services/dfs/federated.

6. Verify that there is an account for remotefshuser on the remote search head.

7. Define the federated search in the savedsearches.conf configuration file using the following command:

   ./splunkd rest --auth=fsh_admin:mychanged1 POST /servicesNS/nobody/system/saved/searches
   "name=remote_splunk_deployment_0_search&federated.provider=remote_splunk_deployment_0&search=search something"

   Result: The following sample stanza appears in the savedsearches.conf file:

   [remote_splunk_deployment_0_search]
   federated.provider = remote_splunk_deployment_0
   search = search
   index = _internal | stats count

8. Create the roles in the authorize.conf configuration file using the following command:

   ./splunkd rest --auth=fsh_admin:mypassword1 POST /services/authorization/roles/
   "name=remotefsh_user1&capabilities=fsh_search&imported_roles=user&federatedProviders=remote_splunk_deployment_0"

   You can set up multiple user accounts on the federated providers.

   For more information on setting up users with federated search capabilities, see Set up users with federated search capabilities using Splunk Web. The following sample stanza appears in the authorize.conf file:

   [role_remote_splunk_deployment_0]
   importRoles = user
   fsh_search = enabled
   federatedProviders=remote_splunk_deployment_0

   For more information on roles to run federated searches, see Roles to run federated searches.

9. Restart Splunk Enterprise so that the CLI changes are loaded on Splunk Web with Enterprise.

   ~$SPLUNK_HOME/bin/splunk restart

   If you set up the environment variables in a local file, reload the environment variables by sourcing the file.

10. Create, update, delete, or retrieve information on user credentials and passwords pertaining to the remote user accounts associated with the federated providers.

    ./splunkd rest --auth=fsh_admin:mypassword1 POST /services/storage/fshpasswords
    "provider=remote_splunk_deployment_0&name=remotefshuser&password=remotepwd1"

    Result: The following sample stanza appears in the fshpasswords.conf file:

    [credential:remote_splunk_deployment_0:remotefshuser:]
    password = <password>

    Only users with the fsh_manage role can create, edit, or update user credentials using the
    /services/storage/fshpasswords REST API endpoint. Adding or editing user credentials associated with a
    remote deployment using the REST endpoint /storage/fshpasswords creates the fshpasswords.conf file. For
more information, see [https://docs.splunk.com/Documentation/DFS/1.1.1/DFS/fshpasswords
/ storage/fshpasswords].

For more information on setting up user accounts, see Set up user accounts for access control.

**Set up federated searches using configuration files**

You can also set up a federated search by editing configuration files. However, you will still need to use REST endpoints to construct the federated search. For more information on using REST endpoints to set up a federated search, see Set up federated searches using REST API endpoints.

*Configuration files to set up a federated search*

The following table lists the configuration files that you must edit to set up a federated search across multiple Splunk deployments:

<table>
<thead>
<tr>
<th>Configuration file</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>federated.conf</td>
<td>Allows authorized users to add information on the federated providers to run federated searches</td>
</tr>
<tr>
<td>savedsearches.conf</td>
<td>Contains the settings for the federated datasets used in a federated search</td>
</tr>
<tr>
<td>authorize.conf</td>
<td>Allows authorized users to add roles to manage or perform federated searches</td>
</tr>
<tr>
<td>fshpasswords.conf</td>
<td>Contains information on service account passwords for the federated providers</td>
</tr>
</tbody>
</table>

*Edit configuration files for a federated search*

Set up a federated search by manually editing the **stanza**s in specific **configuration files**. For more information on DFS-specific configuration files, see DFS-specific configuration files.

For more information on editing configuration files, see How to edit a configuration file in the Splunk Enterprise *Admin Manual*.

**Prerequisites**

For optimal performance, upgrade the federated providers to version 7.3. For more information on compatibility requirements, see Compatibility matrix. For more information on upgrade requirements, see Upgrade requirements.

**Steps**

1. Access the `limits.conf` file.

   The `limits.conf` configuration file is located in `$SPLUNK_HOME/etc/system/default/`. See limits.conf in the Splunk Enterprise *Admin Manual*.

   You can place a copy of the `limits.conf` configuration file in `$SPLUNK_HOME/etc/system/local/` or create a new `limits.conf` configuration file in `$SPLUNK_HOME/etc/system/local/`.

2. Edit the `dfs_meta_phase_exec_timeout` setting in the `[dfs]` stanza in the `limits.conf` file, which sets the time (in seconds) to wait for various phases to complete during a federated search. The default value is 300.

3. Access the `authorize.conf` file.

   The `authorize.conf` configuration file is located in `$SPLUNK_HOME/etc/system/default/`. See authorize.conf in the Splunk Enterprise *Admin Manual*.

   You can place a copy of the `authorize.conf` configuration file in `$SPLUNK_HOME/etc/system/local/` or create a new `authorize.conf` configuration file in `$SPLUNK_HOME/etc/system/local/`
4. Using the table, edit the [role_<roleName>] stanza in the authorize.conf file to:
   - Add a role to manage the federated search
   - Add a role to perform a federated search

   You can also use Splunk Web to add new roles for federated searches. For more information on adding or editing user roles in authorize.conf file, see Add and edit roles in the Splunk Enterprise Security Manual.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>importRoles</td>
<td>Roles authorized to perform federated searches</td>
<td>admin, user, or power</td>
</tr>
<tr>
<td>fsh_search</td>
<td>Enables the federated search</td>
<td>enabled</td>
</tr>
<tr>
<td>federatedProviders</td>
<td>Name of remote deployment</td>
<td>&lt;remote-deployment-name&gt;</td>
</tr>
<tr>
<td>fsh_manage</td>
<td>Toggle to enable federated search management</td>
<td>enabled</td>
</tr>
</tbody>
</table>

   You can add only one federated provider for each role.

5. Set up a custom federated.conf configuration file at the following location: $SPLUNK_HOME/etc/system/local/. For more information, see federated.conf in this manual.

6. Create a federated provider stanza in the federated.conf configuration file.

7. Using the following table, specify information on the federated providers in the [federated-provider] stanza that you want to include in the federated search.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Remote deployment or federated provider type</td>
<td>splunk</td>
</tr>
<tr>
<td>ip</td>
<td>IP address of the federated provider</td>
<td>&lt;ip_address&gt;</td>
</tr>
<tr>
<td>splunk.app</td>
<td>Name of the Splunk app on the remote Splunk Enterprise deployment where search is performed</td>
<td>search</td>
</tr>
<tr>
<td>splunk.port</td>
<td>Management port for the remote Splunk Enterprise deployment</td>
<td>&lt;port_number&gt;</td>
</tr>
<tr>
<td>splunk.serviceAccount</td>
<td>Authorized user on the remote Splunk Enterprise deployment</td>
<td>&lt;fsh_remoteuser&gt;</td>
</tr>
</tbody>
</table>

   The security credentials associated with this account are managed by the fshpasswords.conf file.

8. Create a uniquely named stanza in the savedsearches.conf configuration file.

9. Using the following table, define the datasets that you want to include in the federated search in the [<unique_dataset_name>] stanza. For more information, see the savedsearches.conf in the Splunk Enterprise Admin Manual.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>search</td>
<td>Provide a search string in the Splunk search processing language (SPL)</td>
<td>&lt;spl&gt;;</td>
</tr>
<tr>
<td>federated.provider</td>
<td>Identifies the federated provider on which the search runs</td>
<td>&lt;remote-deployment-stanza&gt;</td>
</tr>
</tbody>
</table>

10. Edit information on user credentials for remote user accounts associated with the federated providers by creating the [credential:<federated-provider>:<username>] stanza and editing the password field in the fshpasswords.conf file. The fshpasswords.conf file is created only when you add or edit user credentials associated with a remote deployment using the REST endpoint /storage/fshpasswords. For more information, see /storage/fshpasswords.

11. Restart Splunk Enterprise to enable the modifications in the configuration files.
Secure a DFS deployment

Secure a DFS deployment

Security is not enabled by default in a DFS deployment. Perform the following tasks to secure your DFS deployment:

1. Secure a Splunk Enterprise Deployment
2. Secure a DFS compute cluster
3. Secure network flows in DFS

For more information on troubleshooting error messages during security setup, see Troubleshoot security issues.

Secure a Splunk Enterprise deployment

Splunk Enterprise software is configured to use a set of default digital certificates to provide a basic level of security. While these certificates can discourage casual snoopers, they could still leave you vulnerable because the root certificate is the same in every Splunk download and anyone with the same root certificate can perform authentication. For more information on default certificates, see About the default certificates.

You can customize your security settings and use your own certificates by editing the server.conf configuration file.

To ensure a more secure configuration, generate your own certificates and configure your Splunk Enterprise using the following instructions in the Securing Splunk Enterprise manual:

- Secure your deployment server and clients using certificate authentication
- How to self-sign certificates

Additionally, you must also set the following configuration setting in the [SSLConfig] stanza in the server.conf configuration file on all Splunk Enterprise nodes.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>caCertFile</td>
<td>&lt;Path to your root CA certificate&gt;</td>
</tr>
</tbody>
</table>

This value must be the same as the value for sslRootCAPath

For example: /opt/eserv/certificate/myCACertificate.pem

DFS security mode 0 only works with Splunk default certificates. If you plan to change the default certificate, you can only use security mode 1 or higher.

Secure a DFS compute cluster

The current version of DFS only supports Spark as the compute cluster. To secure the communication flows within the Spark compute cluster, you must enable the security settings within the Spark cluster. Spark uses Transport Layer Security (TLS) protocol and the commons-crypto library to protect data in transit.

Set the following Spark configuration fields in the spark-defaults.conf file located at: $SPARK_HOME/conf/ to enable Spark security on all compute nodes. For more information on securing your Spark cluster, see Spark Security.
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>spark.authenticate</td>
<td>true</td>
</tr>
<tr>
<td>spark.authenticate.secret</td>
<td>&lt;SharedKey&gt;</td>
</tr>
<tr>
<td>spark.network.crypto.enabled</td>
<td>true</td>
</tr>
<tr>
<td>spark.ssl.fs.enabled</td>
<td>true</td>
</tr>
<tr>
<td>spark.ssl.protocol</td>
<td>TLSv1.2</td>
</tr>
<tr>
<td>spark.ssl.keyStore</td>
<td>&lt;KeyStoreFilePath&gt; For example: /opt/eserv/certificatepack/valid/certificate/master/ks.jks</td>
</tr>
<tr>
<td></td>
<td>The path provided is an example.</td>
</tr>
<tr>
<td>spark.ssl.keyPassword</td>
<td>&lt;PrivateKeyPassword&gt;</td>
</tr>
<tr>
<td>spark.ssl.keyStorePassword</td>
<td>&lt;KeyStorePassword&gt;</td>
</tr>
<tr>
<td>spark.ssl.trustStore</td>
<td>&lt;TrustStoreFilePath&gt; For example: /opt/eserv/certificatepack/valid/certificate/master/ts.jks</td>
</tr>
<tr>
<td>spark.ssl.trustStorePassword</td>
<td>&lt;TrustStorePassword&gt;</td>
</tr>
</tbody>
</table>

Spark shuffle is secured because the commons-crypto library is enabled, which is needed to keep your data secure. SSL related settings are needed when you use mode 2 in DFS security. DFS only supports Spark security in the `spark.ssl.fs` configuration namespace, which is the file download client that is used to download JAR files and files from HTTPS-enabled servers.

### Secure network flows in the DFS deployment

A secure DFS deployment secures the data flows between:

- the Splunk Enterprise **indexers** and the DFS compute cluster
- the Splunk Enterprise **search head** and the DFS compute cluster
- the federated search head and the search heads in the remote deployments
- the search heads in the remote deployments and the DFS compute cluster

The flows between the indexers or the search heads in the remote deployments and the compute cluster are secured using a two-way SSL digital certificate authentication over a TLS connection. Thus, the client certificate of the remote search heads or indexers and the server certificate of the DFS workers will mutually authenticate each other to secure network communication.

To secure the data flow between the compute nodes and use TLS to secure your DFS deployment, edit the **[dfs_security]** stanza settings in the `server.conf` file.

To apply the changes you made to the configuration settings, you must restart Splunk Enterprise.

For more information, see the `server.conf` file.

Ensure that the configuration settings `tls_enabled` is set as specified in the following table on all Splunk Enterprise nodes. Additionally, ensure that the `tls_protocol` setting that you used is also included with Splunk Enterprise `sslVersions` that is specified in the **[sslConfig]** stanza.

For **[dfs_security]**:
The following configurations are optional for all security levels. Enabling this configuration helps to verify the common name and alternative name on the DFS workers with the certificates from the indexer.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>tls_enabled</td>
<td>Toggle to enable Transport Layer Security (TLS)</td>
</tr>
<tr>
<td>tls_protocol</td>
<td>Supported TLS protocol for DFS</td>
</tr>
</tbody>
</table>

The common names or alternative names are typically the host and domain names. For example: `examplehost.splunk.com` or `splunk.com`. If the common names on the DFS workers pass the verification test, the alternative names are not verified. Also, if you only configure `verify_search_peer_to_dfw_client_certificate = true`, without configuring `search_peer_to_dfw_common_name_list` and `search_peer_to_dfw_alt_name_list`, the search runs without verifying common names or alternative names.

**Configure DFS security modes**

There are four different security modes available in DFS. The difference between each mode is how certificates are generated and distributed. You can choose to implement a security mode based on your configuration preferences.

- Mode 0: (Default) Splunk security certificates
- Mode 1: Push customer configured certificates
- Mode 2: Compute cluster security certificates
- Mode 3: Node-specific certificates

Changes to the configuration settings must only be made on the search head to configure DFS security modes.

DFS security uses the following fields in the `server.conf` configuration file to specify the security mode:

- `override_default_certificate`
- `use_spark_security_configs`
- `use_node_specific_certificates`

If you enable more than one of these security fields, the highest DFS security mode always takes precedence.

**Mode 0: Default Splunk security certificates**

In Mode 0, DFS will generate the Keystore and the Truststore based on the Splunk default certificates and push those files to the DFS compute cluster. In this mode, all Splunk nodes are expected to use the Splunk default certificate to ensure the DFS compute cluster can establish a successful connection with Splunk Enterprise.

You must remember that these certificates only discourage casual snoopers and could still leave you vulnerable. This is because the root certificate is the same in every Splunk Enterprise download and anyone with the same root certificate can perform authentication. For more information, see About the default certificates.

If you still want to use mode 0, make sure all your Splunk nodes are using Splunk default certificates. If you decided to change, follow instructions in Secure a Splunk Enterprise deployment first and continue with mode 1 or higher mode.
The `legacy_ca_certificate_folder` is only required when you use federated search features and include an older version of Splunk Enterprise (6.5 and earlier) as a remote deployment. In such cases, you must copy the older version of the Splunk default certificate authority (CA) file manually to the `legacy_ca_certificate_folder` that you configured. The Splunk default certificate authority (CA) file is located in the following location: `SPLUNK_HOME/etc/auth/cacert.pem`.

For more information about each configuration, see Data fabric search security configuration in the `server.conf` configuration file.

**Mode 1: Push customer configured certificates**

In mode 1, DFS does not generate the Keystore and the Truststore automatically. Instead, DFS uses the Keystore and the TrustStore that you configured on the search head and enables you to push those certificates to the DFS compute cluster. In this mode, you are expected to create your own certificate. For more information on generating certificates, see How to generate certificates for DFS.

Certificates that you configured and the certificates on Splunk nodes must all be signed by the same Certificate Authority (CA) in this mode.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>tls_enabled</td>
<td>true</td>
</tr>
<tr>
<td>override_default_certificate</td>
<td>Default value: false</td>
</tr>
<tr>
<td>use_spark_security_configs</td>
<td>Default value: false</td>
</tr>
<tr>
<td>use_node_specific_certificates</td>
<td>Default value: false</td>
</tr>
<tr>
<td>legacy_ca_certificate_folder</td>
<td><code>&lt;Folder path&gt;</code></td>
</tr>
</tbody>
</table>

For more information about each configuration, see Data fabric search security configuration in the `server.conf` configuration file.
Mode 2: Use compute cluster security settings

In mode 2, DFS neither generates nor distributes your certificates. Instead, DFS directly uses the certificates you already configured in your computer cluster security configurations. For more information on securing a DFS compute cluster, see Secure a DFS computer cluster. For information on how to generate certificates, see How to generate certificates for DFS.

Spark certificates and Splunk Enterprise node certificates must all be signed by the same Certificate Authority (CA) in this mode.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>tls_enabled</td>
<td>true</td>
</tr>
<tr>
<td>override_default_certificate</td>
<td>Default value: false</td>
</tr>
<tr>
<td>use.spark_security_configs</td>
<td>true</td>
</tr>
<tr>
<td>use_node_specific_certificates</td>
<td>Default value: false</td>
</tr>
</tbody>
</table>

For more information about each configuration, see Data fabric search security configuration in the server.conf configuration file.

Mode 3: Node specific certificates

In mode 3, DFS uses the certificates that you configured on each Splunk search head and DFS workers nodes based on the configuration settings on the search head. You must manually distribute the certificates to the search head and all the DFS workers in advance. Different nodes may have different certificates as long as they are signed by the same Certificate Authority (CA) as is the case in mode 1. You can also use a separate set of certificates in Spark as in mode 2, which means your DFS compute cluster may use certificates signed by a Certificate Authority (CA) that is not used on Splunk Enterprise nodes.

Certificates configured for DFS and Splunk node certificates both need to be signed by the same Certificate Authority (CA) in this mode. Certificates on all DFS workers must be configured under the same path as those configured on the search head.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>tls_enabled</td>
<td>true</td>
</tr>
<tr>
<td>override_default_certificate</td>
<td>Default value: false</td>
</tr>
<tr>
<td>use.spark_security_configs</td>
<td>Default value: false</td>
</tr>
<tr>
<td>use_node_specific_certificates</td>
<td>true</td>
</tr>
<tr>
<td>dfc_keystore_path</td>
<td>&lt;DFCKeystorePath&gt;</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>/opt/eserv/certificatepack/valid/certificate/master/ks.jks</td>
</tr>
<tr>
<td>dfc_truststore_path</td>
<td>&lt;DFCTruststorePath&gt;</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>/opt/eserv/certificatepack/valid/certificate/master/ts.jks</td>
</tr>
<tr>
<td>dfc_key_password</td>
<td>&lt;DFCKeystorePassword&gt;</td>
</tr>
<tr>
<td>dfc_key_password</td>
<td>&lt;DFCPrivateKeyPassword&gt;</td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>dfc_truststore_password</td>
<td><code>&lt;DFCTruststorePassword&gt;</code></td>
</tr>
<tr>
<td>dfw_keystore_path</td>
<td><code>&lt;DFWKeystorePath&gt;</code></td>
</tr>
<tr>
<td>dfw_truststore_path</td>
<td><code>&lt;DFWTruststorePath&gt;</code></td>
</tr>
<tr>
<td>dfw_keystore_password</td>
<td><code>&lt;DFWKeystorePassword&gt;</code></td>
</tr>
<tr>
<td>dfw_key_password</td>
<td><code>&lt;DFWPrivateKeyPassword&gt;</code></td>
</tr>
<tr>
<td>dfw_truststore_password</td>
<td><code>&lt;DFWTruststorePassword&gt;</code></td>
</tr>
</tbody>
</table>

For more information about each configuration, see Data fabric search security configuration in the `server.conf` configuration file.

**How to generate certificates for DFS**

DFS uses certificates stored in the Keystore and the Truststore to secure network communications over TLS. Keystore contains the private key and Truststore contains all the certificates and Certificate Authority (CA) that is trusted.

*Create a self-signed Certificate Authority (CA)*

For information on creating the root certificate, see Create the root certificate.

If you already created a root certificate for your Splunk Enterprise, prepare the `myCAPrivateKey.key` and `myCACertificate.pem` to generate the Keystore and Truststore for DFS.

All Splunk Enterprise and DFS node certificates must be signed by the same Certificate Authority (CA).

**Generate KeyStore and TrustStore**

You can use keytool for KeyStore and TrustStore creation and management. For more information on keytool, see keytool.

1. Generate a public and private key pair in KeyStore using Java KeyTool for each node in the compute cluster.

   ```
   keytool -genkeypair \
   -alias <Alias> \
   -keyalg <keyalg> \
   -keysize <keysize> \
   -dname CN=<CommonName>,OU=<OrganizationalUnit>,O=<Organization>,L=<Locality/City>,ST=<State/ProvinceName>,C=<CountryName> \
   -keypass <PrivateKeyPassword> \
   -keystore <FilePathForKeyStore> \
   -storepass <KeyStorePassword> \
   -storetype <KeystoreType>
   ```
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>genkeypair</td>
<td>Generates a key pair and adds an entry to the KeyStore</td>
</tr>
<tr>
<td>alias</td>
<td>Short name of the key pair in the KeyStore</td>
</tr>
<tr>
<td>keyalg</td>
<td>Key generation algorithm</td>
</tr>
<tr>
<td></td>
<td>Usually RSA is used and DSA is the default.</td>
</tr>
<tr>
<td>keysize</td>
<td>For RSA: 2048</td>
</tr>
<tr>
<td></td>
<td>• For DSA: 1024</td>
</tr>
<tr>
<td>dname</td>
<td>Distinguished names to identify entities</td>
</tr>
<tr>
<td></td>
<td>For example: CN=&lt;User Name&gt;,OU=&lt;Organization unit&gt;,O=&lt;Organization&gt;,L=&lt;City Name&gt;,ST=&lt;State Name&gt;,C=&lt;Country name&gt;</td>
</tr>
<tr>
<td>keypass</td>
<td>Private key password used to protect the private key of the generated key pair</td>
</tr>
<tr>
<td>keystore</td>
<td>File path and name of the KeyStore</td>
</tr>
<tr>
<td></td>
<td>For example: /usr/local/eserv/dfs/keystore.jks</td>
</tr>
<tr>
<td>storepass</td>
<td>KeyStore password used to protect the integrity of the KeyStore</td>
</tr>
<tr>
<td>storetype</td>
<td>Keystore type</td>
</tr>
<tr>
<td></td>
<td>Default: JKS</td>
</tr>
<tr>
<td></td>
<td>PKCS12</td>
</tr>
</tbody>
</table>

The private key password is used encrypt the private key. KeyStorePassword is used to protect all contents in the keystore including the encrypted private key.

For example:

```
keytool -genkeypair \
    -alias TestServer \
    -keyalg RSA \
    -keysize 2048 \
    -dname "CN=TestDFSServer,OU=DFS,O=Splunk,L=SanFrancisco,ST=CA,C=US" \
    -keypass 123456 \
    -keystore ks.jks \
    -storepass 123456 \
    -storetype PKCS12
```

2. Create a self-signed certificate by exporting a certificate using the public key of the key pair.

```
keytool -export \
    -alias <Alias> \
    -keystore <KeyStoreFilepath> \
    -file <CertificateFilepath> \
    -storepass <KeyStorePassword>
```

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>Short name for the key pair in the KeyStore</td>
</tr>
<tr>
<td>keystore</td>
<td>File path for the public and private key pair. For example: /usr/local/eserv/dfs/keystore.jks</td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
</tr>
<tr>
<td>----------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>file</td>
<td>File path to store certificate file (.cer). For example: /usr/local/eserv/dfs/certificate.cer</td>
</tr>
<tr>
<td>storepass</td>
<td>Password to access the KeyStore</td>
</tr>
</tbody>
</table>

For example:

```
keytool -export \n-alias TestServer \n-keystore ks.jks \n-file cer.cer \n-storepass 123456
```

3. Generate the certificate signing request (CSR).

```
keytool -certreq \n-alias <Alias> \n-keystore <KeyStoreFilePath> \n-file <CertificateSigningRequestFilePath> \n-storepass <KeyStorePassword>
```

For example:

```
keytool -certreq \n-alias TestServer \n-keystore ks.jks \n-file csr.csr \n-storepass 123456
```

4. Sign the certificate with a CA.

```
openssl x509 \n-CA <CACertificate> \n-CAkey <CAPrivateKey> \n-CAcreateserial \n-req -in <CertificateSigningRequestFilePath> \n-out <SignedCertificate> \n-days <ValidDays> \n-passin pass:<CAPrivateKeyPassword>
```

For example:

```
$SPLUNK_HOME/bin/splunk cmd openssl x509 \n-CA myCACertificate.pem \n-CAkey myCAPrivateKey.key \n-CAcreateserial \n-req -in csr.csr \n-out signCer.cer \n-days 365 \n-passin pass:password
```

5. Add certificate authority (CA) in KeyStore.

```
keytool -import \n-noprompt \n-alias <CAAlias> \n-file <CACertificate> \n-keystore ks.jks 
```
keytool -import \\n-noprompt \\n-alias TestCA \\n-file myCACertificate.pem \\n-keystore ks.jks \\n-storepass 123456

6. Add the signed certificate to the key pair in the KeyStore.

keytool -import -trustcacerts \\n-alias <Alias> \\n-file <SignedCertificate> \\n-keystore <KeyStoreFilePath> \\n-storepass <KeyStorePassword>

For example:

keytool -import -trustcacerts -noprompt \\n-alias TestServer \\n-file signCer.cer \\n-keystore ks.jks \\n-storepass 123456

7. Add the signed certificate to the TrustStore.

keytool -import -noprompt \\n-alias <Alias> \\n-file <SignedCertificate> \\n-keystore <TrustStoreFilePath> \\n-storepass <TrustStorePassword>

For example:

keytool -import -noprompt \\n-alias TestServer \\n-file signCer.cer \\n-keystore ts.jks \\n-storepass 123456

The Truststore filepath is different from the Keystore filepath.

8. Add the CA certificate to the TrustStore.

keytool -import -noprompt \\n-alias <CAAlias> \\n-file <CACertificate> \\n-keystore <TrustStoreFilePath> \\n-storepass <TrustStorePassword>

For example:

keytool -import -noprompt \\n-alias TestCA \\n-file myCACertificate.pem \\n-keystore ts.jks \

-storepass 123456
DFS commands

dfsjob

The `dfsjob` command uses Data Fabric Search (DFS) processing to enable you to run searches over high-cardinality data volumes containing billions of events. DFS processing also expands the range of the `join` and `union` commands, enabling them to support multiple data sources efficiently through partitioned processing of the search pipeline.

The `dfsjob` command requires a Splunk platform environment in which Data Fabric Search functionality is enabled, licensed, and properly configured.

You can only use the `dfsjob` command once in a search.

Syntax

dfsjob [<subsearch>]

`subsearch`

Syntax: `"[ subsearch ]"`
Description: A secondary search that specifies the search that the `dfsjob` command submits to the Data Fabric Search process. DFS currently supports only transforming searches.

The search enclosed by the `dfsjob` command must meet these conditions:

- Be enclosed in square brackets
- Must be a transforming search
- Start with a supported search command such as `search`, `from`, or `union`

The subsearch can have streaming commands before the DFS-supported commands.

Ordinarily, subsearches have strict limits on the number of events they can return and the length of time they can run. However, the DFS process removes these limitations when you use subsearches in the context of the `dfsjob` command.

See About subsearches, in the Splunk Enterprise Search Manual.

Optional arguments

None.

Usage

When DFS functionality is licensed, enabled, and properly configured for your Splunk platform implementation, use the `dfsjob` command to indicate that a particular search should use DFS processing. The DFS process is designed for searches that are high in cardinality and data volume.
**The dfsjob map-reduce process**

Data Fabric Search is similar to Splunk platform features like distributed search and parallel reduce search in that it uses a map-reduce process to efficiently search over large high-cardinality datasets. In the map phase, the search process locates event data that matches the search and organizes it into field-value pairs. During the reduce phase, the search process sorts the mapped results through the commands in your search and aggregates them.

Searches with the dfsjob command carry out a complex map-reduce operation in at least two broad phases: a mapping phase, and a cycled map/reduction phase. A third reduction and aggregation phase can also take place if the SPL search requires it. Each of these phases happens in a different part of the Splunk Enterprise architecture.

The following table describes the various processing locations and processing stages of a DFS search:

<table>
<thead>
<tr>
<th>Search phase</th>
<th>Description</th>
<th>Processing location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapping</td>
<td>The DFS process locates event data matching the SPL search and sorts it into field-value pairs.</td>
<td>The indexer layer of your Splunk Enterprise deployment. An indexer layer search is known as a remote search.</td>
</tr>
<tr>
<td>Cycled map/reduction</td>
<td>The DFS process runs mapped results through the commands in the dfsjob subsearch, using as many cycles of map/reduction as necessary.</td>
<td>Nodes of a compute cluster outside of your Splunk Enterprise deployment. The results are partitioned and processed in parallel by DFS worker processes on the compute cluster nodes.</td>
</tr>
<tr>
<td>Reduction and aggregation</td>
<td>This phase only takes place if you have SPL on the right side of the dfsjob subsearch. In it, the Splunk search process runs the reduced results from the compute cluster through that SPL outside of the subsearch. When the last reduction is complete, the Search process aggregates the results.</td>
<td>The search head layer of your Splunk Enterprise deployment.</td>
</tr>
</tbody>
</table>

The cyclic map/reduction phase is where the DFS process gets its productivity gains. During this phase, the search results are partitioned across the nodes of the compute cluster and are processed in parallel by DFS worker processes. This parallelization of reduction work that otherwise would be done entirely on the search head layer results in faster completion times for high-cardinality searches that aggregate extremely large numbers of search results.

**Proper organization of a dfsjob search**

Carefully organize your dfsjob command searches so that the bulk of the search work is handled by the subsearch. The subsearch contains the map and intermediate reduce portions of the search. The subsearch may only contain commands that are supported by the DFS process.

Here is an example dfsjob search:

```
| dfsjob [search index=airline700
| | stats count by ArrDelay]
| | chart max(ArrDelay)
```

The following table shows you how that example search is broken up into three segments, one for each broad phase of the search:

<table>
<thead>
<tr>
<th>Process</th>
<th>Part of search processed</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapping (remote search)</td>
<td>First part of the subsearch, up to the first DFS-supported command</td>
<td>search index=airline700</td>
</tr>
</tbody>
</table>
## Command type

The `dfsjob` command is an **orchestrating command**, which means that it controls how a search runs. It does not focus on the events processed by the search. The `dfsjob` command performs the following actions:

- Determines whether the search can be supported by DFS and rejects the search if it cannot.
- Sends supported searches to the Data Fabric Controller, which breaks the search down into the remote search and the `dfsjob` search.
- Determines how the `dfsjob` search is fed into, partitioned by, and retrieved from the compute cluster.
- Generates a directed acyclic graph (DAG), which determines how the DFS workers in the compute cluster process the search through multiple rounds of reduction.

For more information about command types, see Types of commands in the Splunk Enterprise Search Reference.

### Ordered search results from a `dfsjob` search

Certain SPL search commands that the `dfsjob` command supports explicitly return results in a sorted order. As a result of the partitioning that takes place when the `dfsjob` command is run, Splunk Enterprise loses the sorting order. If you require the results of a `dfsjob` search to be sorted in that exact order, use `sort` to perform the sorting at the search head. There is an additional performance cost to event sorting after the `dfsjob` command partitions events on the nodes of the compute cluster.

The following SPL search provides ordered results:

```
[search index=airline700 | stats count by ArrDelay]
```

If you want to get that same event ordering while also adding `dfsjob` to the search to speed it up, add `sort` to the search:

```
| dfsjob [search index=airline700
| stats count by ArrDelay
| sort 0 str(ArrDelay)]
```

The `sort` command is supported by `dfsjob`, so it can be included in the `dfsjob` command subsearch as long as the commands that precede it are also supported by `dfsjob`.

### Examples

Detailed examples are also included in the documentation for the various commands supported by DFS. For more information, see Commands supported b DFS.

---

<table>
<thead>
<tr>
<th>Process</th>
<th>Part of search processed</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycled map/reduction (DFS search)</td>
<td>Remainder of the subsearch, starting with the first DFS-supported command</td>
<td>stats count by ArrDelay</td>
</tr>
<tr>
<td></td>
<td>Only DFS-supported commands are allowed.</td>
<td></td>
</tr>
<tr>
<td>Final reduction and aggregation (final search)</td>
<td>Any search processing language (SPL) that follows the subsearch</td>
<td>chart max(ArrDelay)</td>
</tr>
<tr>
<td></td>
<td>This SPL can contain commands that are not supported by DFS.</td>
<td></td>
</tr>
</tbody>
</table>
1. Speed up a search on a large high cardinality dataset

In this example, the `dfsjob` command is applied to a `stats` search that is running over an extremely large high cardinality dataset. The `dfsjob` command reduces the completion time for the search.

```
dfsjob [index-airlinesdata_0 OR index-airlinesdata_1 OR index-airlinesdata_2 OR index-airlinesdata_3 OR index-airlinesdata_4
 | stats count by ArrDelay, ArrTime, AirTime
 | sort ArrDelay]
```

The DFS workers process the `stats count by ArrDelay, ArrTime, AirTime | sort ArrDelay` portion of the search in parallel on the nodes of the compute cluster, lowering the completion time for the search.

2. Unsupported commands follow the `dfsjob` command

Here the `dfsjob` command is used to run a `stats` command search on an index of domestic airline data. It then processes the results through the `top` command, `field` command, and `sort` command.

```
dfsjob [search index=airline_domestic
 | stats count by DepTime, ArrTime]
 | top limit=100 DepTime ArrTime
 | fields DepTime ArrTime
 | sort DepTime
```

The `dfsjob` command causes the `stats` command reduction work to take place entirely on the compute cluster. Then the results are sent to the search head, where they are processed through the `top` command, `field` command, and `sort` command.

Note that the `top` command is not supported by DFS. Therefore, the `sort` command is processed on the search head as it is outside of the `dfsjob` subsearch. This enables the DFS SPL search to run without problems.

3. Run a `dfsjob` search that includes streaming commands in the subsearch

This search includes a `rex` command before the first DFS-supported command. It is run on the indexers, as part of the initial remote search.

```
dfsjob [search index=airline700
 | rex field=ccnumber mode=ed "s/\d{4}-\{3\}/XXXX-XXXX-XXXX-/g"
 | stats count by ArrDelay]
```

4. Use `dfsjob` to run a federated search that merges datasets from two Splunk Enterprise deployments

This example shows a `dfsjob` search where the `union` command is used to merge together datasets from two different Splunk Enterprise deployments.
| dfsjob |
| union[ from federated:deployment_1_search_1 |
| {} from federated:deployment_2_search_1] |

The work of the union command merge is entirely processed by the DFS workers on the nodes of the compute cluster.

5. Use dfsjob to run a federated search that joins datasets from two different Splunk deployments

Here dfsjob is used to perform a join on datasets from two Splunk deployments. The join takes place on the ArrDelay field.

| dfsjob |
| from federated:deployment_1_search_1 |
| join left=L right=R where L.ArrDelay=R.ArrDelay |
| {} from federated:deployment_2_search_1 |
| sort -L.ArrDelay |
| head 100 |

The work of the join command and the following sort command are processed by the DFS workers on the compute cluster.

Commands supported by dfsjob

The dfsjob command subsearch supports only the following commands:

- dedup
- eval
- fields
- from
- head
- join
- rename
- reverse
- sort
- stats
- tail
- union
- where

dedup

A DFS search using the dedup command can operate differently from the SPL search command in that the "consecutive" argument is not supported or the search results display differently. Additionally, the dedup command can return different results if the values of the duplicate field are the same for multiple events.

Thus, if there are two events, one with a=1, b=2, and a second event with a=1, b=3 and the dedup field is "a", the DFS search can return either event.

Using a DFS search with the dedup command with the following input data in fields "a" and "b" may return different search results:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>
Therefore, search results may be returned as expected.

Alternatively, the search results may also be displayed differently:

The search results are displayed based on the order in which the events were processed on a distributed cluster and are both correct. If you want to display a specific value for the "b" field (largest or smallest) in the search results, you may add a `sort` command before the `dedup` command as follows:

`| dfsjob [ |search a < 3 | sort b | dedup a]`

However, this will drop the efficiency of the search significantly.

Following are examples of a DFS search using the `dedup` command, where events that contain an identical combination of values for fields like `count` or `date_minute` or `date_day` are removed from the search results.

```
| dfsjob [search index="_internal" | stats count by date_hour | dedup count]

| dfsjob [search index=ansible_access_combined earliest=0 latest=now | stats count by date_minute date_mday | dedup date_minute date_mday]
```

For general information on the `dedup` command, see `dedup` in the Splunk Enterprise Search Reference.

**Example**

**Unsupported "consecutive=true" argument**

For DFS, the optional argument "consecutive" is not supported with the `dfsjob` command. The `dedup` command acts as if the argument "consecutive" is set to "false". In fact, running a `dfsjob` command with "consecutive=false" works as expected. Thus, the following DFS search returns an error:

```
| dfsjob [search index=ansible_access_combined earliest=0 latest=now | stats count by date_minute date_mday | dedup consecutive=true date_minute date_mday]
```
**eval**

A DFS search offers limited support for certain functions and operators with the `eval` command. A function that is placed before a transforming command like `stats` is supported by DFS, even if it is not in the list of supported functions. Also, if we include a value as an input for a function instead of a field, the function is supported by DFS.

Following are some examples of DFS searches using the `eval` command:

| dfsjob [search index="_internal" | stats count by date_hour | eval result = if(count>10000, "over", "under")]
| dfsjob [search index="_internal" | stats count( eval(date_hour=11) ) as count_of_10]
| dfsjob [search index="_internal" | eval mark_field=1 | stats count by date_hour,mark_field]

For general information on the `eval` command, see [eval in Splunk Enterprise Search Reference](https://docs.splunk.com/Documentation/Splunk/9.0.0/).  

**Evaluation operators**

The following table lists the evaluation operators supported by DFS:

<table>
<thead>
<tr>
<th>Type</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic</td>
<td>+ - * / %</td>
</tr>
<tr>
<td>Concatenation</td>
<td>.</td>
</tr>
<tr>
<td>Boolean</td>
<td>AND OR NOT XOR &lt; &gt; &lt;= &gt;= != = == LIKE</td>
</tr>
</tbody>
</table>

**Evaluation functions**

The following table lists the evaluation functions supported by DFS:

<table>
<thead>
<tr>
<th>Category</th>
<th>Supported functions and syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comparison and conditional functions</td>
<td>case()</td>
<td>Accepts alternating conditions and values. Returns the first value for which the condition evaluates to TRUE.</td>
</tr>
<tr>
<td></td>
<td>cidrmatch(&quot;X&quot;,Y)</td>
<td>Returns TRUE or FALSE based on whether an IP address matches a CIDR notation.</td>
</tr>
<tr>
<td></td>
<td>coalesce(X,...)</td>
<td>This function takes an arbitrary number of arguments and returns the first value that is not NULL.</td>
</tr>
<tr>
<td></td>
<td>false()</td>
<td>Returns FALSE.</td>
</tr>
<tr>
<td></td>
<td>if(X,Y,Z)</td>
<td>If the condition X evaluates to TRUE, returns Y, otherwise returns Z.</td>
</tr>
<tr>
<td></td>
<td>in(FIELD, VALUE-LIST)</td>
<td>The function returns TRUE if one of the values in the list matches a value in the field you specify.</td>
</tr>
<tr>
<td></td>
<td>like(TEXT, PATTERN)</td>
<td>Returns TRUE if TEXT matches PATTERN.</td>
</tr>
<tr>
<td></td>
<td>match(SUBJECT, &quot;REGEX&quot;)</td>
<td>Returns TRUE or FALSE based on whether REGEX matches SUBJECT.</td>
</tr>
<tr>
<td></td>
<td>null()</td>
<td>This function takes no arguments and returns NULL.</td>
</tr>
<tr>
<td>Category</td>
<td>Supported functions and syntax</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>nullif(X,Y)</td>
<td>This function is used to compare fields. The function takes two arguments, X and Y, and returns NULL if X = Y. Otherwise it returns X.</td>
<td></td>
</tr>
<tr>
<td>searchmatch(X)</td>
<td>Use this function to return TRUE if the search string (X) matches the event.</td>
<td></td>
</tr>
<tr>
<td>true()</td>
<td>Returns TRUE.</td>
<td></td>
</tr>
<tr>
<td>validate(X,Y,...)</td>
<td>Use this function to return the string Y corresponding to the first expression X that evaluates to FALSE. This function is the opposite of the case function.</td>
<td></td>
</tr>
<tr>
<td>Conversion functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tonumber(NUMSTR,BASE)</td>
<td>Converts a string to a number.</td>
<td></td>
</tr>
<tr>
<td>tostring(X,Y)</td>
<td>Converts the input, such as a number or a Boolean value, to a string.</td>
<td></td>
</tr>
<tr>
<td>Mathematical functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>abs()</td>
<td>Returns the absolute value.</td>
<td></td>
</tr>
<tr>
<td>ceiling(X) or ceil(X)</td>
<td>Rounds the value up to the next highest integer.</td>
<td></td>
</tr>
<tr>
<td>exact(X)</td>
<td>Returns the result of a numeric eval calculation with a larger amount of precision in the formatted output.</td>
<td></td>
</tr>
<tr>
<td>exp(X)</td>
<td>Returns the exponential function e^x.</td>
<td></td>
</tr>
<tr>
<td>floor(X)</td>
<td>Rounds the value down to the next lowest integer.</td>
<td></td>
</tr>
<tr>
<td>ln(X)</td>
<td>Returns the natural logarithm.</td>
<td></td>
</tr>
<tr>
<td>log(X,Y)</td>
<td>Returns the logarithm of X using Y as the base. If Y is omitted, base 10 is used.</td>
<td></td>
</tr>
<tr>
<td>pi()</td>
<td>Returns the constant π to 11 digits of precision.</td>
<td></td>
</tr>
<tr>
<td>pow(X,Y)</td>
<td>Returns X to the power of Y, x^y.</td>
<td></td>
</tr>
<tr>
<td>round(X,Y)</td>
<td>Returns X rounded to the amount of decimal places specified by Y. The default is to round to an integer.</td>
<td></td>
</tr>
<tr>
<td>sqrt(X)</td>
<td>Returns the square root of the value.</td>
<td></td>
</tr>
<tr>
<td>urlencode(X)</td>
<td>Replaces URL escaped characters with the original characters.</td>
<td></td>
</tr>
<tr>
<td>Multivalue eval functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>split(X,&quot;Y&quot;)</td>
<td>Returns a mv field splitting X by the delimited character Y.</td>
<td></td>
</tr>
<tr>
<td>Text functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>len(X)</td>
<td>Returns the count of the number of characters (not bytes) in the string.</td>
<td></td>
</tr>
<tr>
<td>lower(X)</td>
<td>Converts the string to lowercase.</td>
<td></td>
</tr>
<tr>
<td>substr(X,Y,Z)</td>
<td>Returns a substring from X based on the starting position Y and the length Z.</td>
<td></td>
</tr>
<tr>
<td>upper(X)</td>
<td>Returns the string in uppercase.</td>
<td></td>
</tr>
<tr>
<td>urlencode(X)</td>
<td>Replaces URL escaped characters with the original characters.</td>
<td></td>
</tr>
<tr>
<td>Informational functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>isbool(X)</td>
<td>Returns TRUE if the field value is Boolean.</td>
<td></td>
</tr>
<tr>
<td>isint(X)</td>
<td>Returns TRUE if the field value is an integer.</td>
<td></td>
</tr>
<tr>
<td>isnotnull(X)</td>
<td>Returns TRUE if the field value is not NULL.</td>
<td></td>
</tr>
<tr>
<td>isnull(X)</td>
<td>Returns TRUE if the field value is NULL.</td>
<td></td>
</tr>
<tr>
<td>isnum(X)</td>
<td>Returns TRUE if the field value is a number.</td>
<td></td>
</tr>
<tr>
<td>isstr(X)</td>
<td>Returns TRUE if the field value is a string.</td>
<td></td>
</tr>
<tr>
<td>typeof(X)</td>
<td>Returns a string that indicates the field type, such as Number, String, Boolean, and so forth.</td>
<td></td>
</tr>
</tbody>
</table>
Examples

A function that is placed before a transforming command like `stats` is supported by DFS, even if it not in the list of supported functions. Also, if we include a value as an input for a function instead of a field, the function is supported by DFS. For example, the `trim()` function is not in the list of supported functions, yet can be used in a DFS SPL search in the following two scenarios:

**Functions before a transforming command like `stats` are supported**

For example: When the `trim()` function appears before a transforming command like `stats` in a DFS SPL search, then the `trim()` function runs on the indexer. Thus, the `trim()` function is supported by DFS because it reads the `eval` command as a streaming command in the search pipeline. However, when the `trim()` function appears after the transforming command like `stats`, DFS processes it on the compute cluster and returns an error message.

The following DFS search with `trim()` is supported as `eval n=trim(newfield, "Z")` runs on the indexer.

```
| dfsjob [search index=<source_type> | eval newfield="ZZZZabcZZ" | eval n=trim(newfield, "Z") | stats count by newfield,n]
```

**Eval functions that evaluate to a constant are supported**

For example: The input value for the `trim()` function must be optimized so that `trim()` can be supported by a DFS SPL search even though it appears after a transforming command like `stats`.

Thus, the following search is supported because the input value for the `trim()` function is pre-calculated to the following value "ZZZZabcZZ":".

```
| dfsjob [search index=<source_type> | stats count | eval n=trim("ZZZZabcZZ", "Z")]
```

However, the following DFS search with `trim()` is not supported because the input for the `trim()` function is a field "newfield", which requires further calculation. Additionally, the `trim()` function appears after the `stats` transforming command.

```
| dfsjob [search index=<source_type> | stats count | eval newfield="ZZZZabcZZ" | eval n=trim(newfield, "Z")]
```

**Informational function `isstr(X)` may behave differently**

Informational function `isstr(X)` for `eval` command may behave differently than Enterprise Search. Informational functions are used to check the type of field-value.

In Enterprise search, if `typeof(X)` returns a "Number", the results returned by the `isstr(X)` and `isnum(X)` functions are true.

For a DFS search, if the field value `typeof(X)` is a "Number", then, the results returned by the `isnum(X)` function is true and the `isstr(X)` function is false.

Thus, in the following DFS search, if a field value is a number:
The search results will be as follows: $a$="yes", $b$="no", $c$="Number"

fields

A DFS search using the `fields` command operates in the same manner as an Enterprise search.

Following are examples of DFS searches using the `fields` command:

In this SPL search, a field like `date_mday` is included in the search results:

```
| dfsjob [search index="_internal"
| fields date_mday
| stats count by date_mday ]
```

In this SPL search, a field like `date_mday` is removed the search results:

```
|dfsjob [ search index="_internal"
| stats count by date_mday,date_hour
| fields - date_mday]
```

For general information on the `fields` command, see `fields` in the Splunk Enterprise Search Reference.

from

The `from` command combines the datasets to construct a federated search and allows you to search multiple indexes across federated providers. Information on the remote subsearches and the remote deployments is saved in the `federated.conf` and the `savedsearches.conf` configuration files.

You can set up a federated search using the UI. For more information on using Splunk Web to add, edit, or delete federated providers, see Set up a federated search using Splunk Web. This is the recommended method to set up a federated search.

You can also set up a federated search by modifying specific configuration files. For more information on editing configuration files to set up a federated search, see Set up a federated search using configuration files.

Some command combinations are not supported with the `from` command. The `from` command in a standard search supports multiple dataset types including saved searches, data models, and lookups. However, with DFS functionality enabled, the `from` command may return an error with these dataset types.

Thus, the following DFS search fails because the `inputlookup` command is not supported:

```
[dfsjob | from inputlookup:geo_attr_countries.csv | stats count by continent]
```

The search translates to the following:

```
... | inputlookup geo_attr_countries.csv | stats count by continent
```

For general information on the `from` command, see from in the Splunk Enterprise Search Reference.
Syntax

| from federated:<dataset_name>

Required arguments

<dataset_name>

Syntax: <dataset_name>

Description: The name of the local dataset that is queried from the DFS search head to access information on the remote dataset that resides in the remote deployment. The search defined in the dataset is run on the remote deployment.

Examples

Construct a federated search

Construct a federated search using two partitioned datasets from two remote Splunk Enterprise deployments: networkRemote1 and networkRemote2. The search runs on the dataset search index=networkLocal and adds the counts from the deployments.

<table>
<thead>
<tr>
<th>dfsjob</th>
<th>union</th>
<th>from federated:networkRemote1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>stats count by ip</td>
</tr>
<tr>
<td></td>
<td>from federated:networkRemote2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>stats count by ip</td>
<td></td>
</tr>
<tr>
<td></td>
<td>search index=networkLocal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>stats count by ip</td>
<td></td>
</tr>
<tr>
<td></td>
<td>stats sum(count) by ip</td>
<td></td>
</tr>
</tbody>
</table>

head

A DFS search using the head command operates differently from standard search. Random results are displayed every time a DFS search is run. To get consistent results, sort the results in the specific column using the sort command.

Following are some examples of DFS searches using the head command:

| dfsjob [search index=._internal |
| where date_hour>11 |
| stats count by date_hour |
| head 20] |

| dfsjob [search index=":audit" |
| stats count by date_hour |
| where count>100 |
| head 20] |
For general information on the `head` command, see head in Splunk Enterprise Search Reference.

**Examples**

1. **Use sort with head command to order search results**

   You can run a DFS search with the `head` command to return the first $N$ events returned by the distribute compute cluster as follows:

   ```plaintext
   | dfsjob [search index="index_name" sourcetype="sourcetype_name" | head N]
   ```

   However, you can limit the number of results returned by a DFS search as follows:

   ```plaintext
   | dfsjob [search index="index_name" sourcetype="sourcetype_name"
   | stats count by <field>
   | sort <field>
   | head N
   |
   ```

   Also, the search results displayed may be in random order. Including the `sort` command before the `head` command in the search can display the results in order. The `stats` command included in the DFS search helps to improve performance by reducing data that must be processed by the search.

2. **All optional arguments of the head command are supported with DFS**

   You can use all optional arguments for the `head` command with `dfsjob` to customize the results returned when running a DFS SPL search. Optional arguments that you can use include:

   ```plaintext
   1. limit
   2. eval-expression
   3. keeplast
   4. null
   ```

   For example, if you want to return search results by the number of items per year, only since 2010, you can use the `head` command in the following search:

   ```plaintext
   | dfsjob [search index="_internal"
   | stats count by date_year
   | sort - date_year
   | head date_year>2010
   |
   ```

   An alternative syntax for this search can be:

   ```plaintext
   | dfsjob [search index="_internal"
   | stats count by date_hour
   | sort date_hour
   | head limit=5
   |
join

A DFS search using the `join` command operates differently from standard search, when combining the contents of two datasets. Unlike standard search, DFS does not apply subsearch arguments with the `join` command. Instead, DFS breaks the search into a remote search and a DFS search.

A DFS search with a left (or outer) join operates like a SQL query and pulls in events that are common to the datasets along with the remaining non-matching events from the second dataset. In standard search, the results of a left (or outer) join includes all the events in the main search and only those values in the subsearch that have matching field values.

A DFS search using the `join` command displays different event counts from a standard search. In DFS, the `max` option, which specifies the maximum number of subsearch results with the `join` command, does not support a non-zero value and defaults to 0.

A DFS search using the `join` command also ignores the following options, if specified: `usetime`, `earlier`, and `overwrite`. DFS behaves as if `usetime` is set to `false` because a DFS join operation runs until it exhausts all results. The `earlier` option is ignored with DFS because it is a sub-option to `usetime=true`. Also, the `overwrite` option works as if it is set to `true`.

DFS does not have the same limitations that are imposed by standard search with the `join` command. These limitations with standard search joins are specified in the `limits.conf` configuration file and include the maximum number of subsearches that can be joined, the maximum search time for the subsearch, and the maximum time to wait for the subsearch to complete.

Outer joins, which are accomplished by including `type=outer` in conjunction with `join` in a search, behave differently in data fabric searches.

The `join` command is not supported for a federated search by federated providers running on Splunk Enterprise 6.5 and below.

Following is an example of a DFS search using the `join` command:

```plaintext
| dfsjob [ search index="_audit"
|  stats count by host
|  join left=L right=R where L.host=R.host [search index=_internal
|  stats count by host
| ]
| ]
```

For general information on the `join` command, see `join` in the Splunk Enterprise Search Reference.

Examples

1. A left (or outer) `join` option operates like an SQL query

A DFS search does not support subsearch options with the `join` command. You can run a DFS search with the `join` command that includes events common to "index_name_1" and "index_name_2" along with the remaining non-matching events from "index_name_2".

```plaintext
| dfsjob [ search index="index_name_1"
|  stats count by field_X
```
| join type=outer max=0 left=L right=R
where L.field_X = R.field_X [ search "index_name_2"
| stats count by field_X
]

However, a similar standard search includes all the events in the main search and only those values in the subsearch that have matching field values.

2. The \texttt{max=<int>} argument in DFS joins defaults to 0

In the following DFS search example, the \texttt{join} command is used to combine the results from a main search with the results from a subsearch, \texttt{search vendors}. The result sets are joined on the \texttt{product_id} field, which is common to both sources. In DFS, the \texttt{max=<int>} argument defaults to 0, therefore the DFS search automatically joins each matching subsearch row with the corresponding main search row.

| dfsjob... | join product_id [search vendors]

\textbf{rename}

A DFS search using the \texttt{rename} command operates in the same manner as standard search.

Following are some examples of DFS searches using the \texttt{rename} command:

| dfsjob [search index="_internal" | rename date_hour as hour | stats count by hour ]
| dfsjob [search index="_internal" | stats max(date_hour) as lastest_hour by date_mday | rename date_mday as day_number]

For general information on the \texttt{rename} command, see rename in the Splunk Enterprise Search Reference.

\textbf{reverse}

A DFS search using the \texttt{reverse} command functions the same as in standard search.

Following is an example of a DFS search using the \texttt{reverse} command:

| dfsjob [search index="_internal" | stats count by host | reverse ]

For general information on the \texttt{reverse} command, see reverse in the Splunk Enterprise Search Reference.

\textbf{sort}

A DFS search using the \texttt{sort} command may display results differently from Enterprise search sort when sorting DFS search results.

Following are examples of DFS searches using the \texttt{sort} command:

| dfsjob [search index="_internal" | stats count by date_hour | sort count ]
For general information on the `sort` command, see sort in the Splunk Enterprise Search Reference.

**Examples**

1. **Command combinations with sort and stats**

A DFS search does not display expected results when the `sort` and `stats` commands are combined together in an SPL search.

The following standard search displays sorted results by fields.

```
index=<index_type> earliest=0 latest=now | sort <field_name1>, <field_name2> | stats count by <field_name1> <field_name2>
```

The following standard search displays sorted results by the client IP addresses and bytes.

```
index=access_combined earliest=0 latest=now | sort bytes, clientip | stats count by bytes clientip
```

However, a similar DFS search does not provide results in sorted order due to the distributed implementation of the SPL search. Additionally, the `sort` command must be placed after `stats` in a DFS search.

```
| dfsjob [search index=access_combine earliest=0 latest=now | stats count by bytes clientip | sort bytes, clientip]
```

2. **Sort command is processed by the big data analysis portion for federated searches**

The sort operation is processed on the "reduce" part of the map-reduce process. For scaled federated searches, it is processed on the big data analysis portion of the SPL search.

In this example, the federated search does not sort results by bytes as expected using the `sort` command.

```
[federated:fed_search_1]<br>
search = search index=access_combined | head 10 | sort - bytes<br>
deployment = remote_deployment_1<br>
estimatedResultCount = 100000<br>
```

Instead, you can use the big data analysis feature to sort the results.

```
| dfsjob [[from federated:<fed_search_1> | sort - bytes]
```

**stats**

A DFS search using the `stats` command operates slightly differently from a standard search due to the partitioned processing of the SPL search. The search results for certain `stats` operators in a DFS environment may differ slightly from the search results run in a Splunk Enterprise environment because the order in which the partitions are processed is non-deterministic both when data is moved from the indexers to the DFS workers as well as when data is moved between DFS workers.
Unsupported stats options in DFS

The delim=<> option to specify a custom delimiter is not supported by the stats command in DFS.

Unsupported stats functions in DFS

The earliest_time(), latest_time(), and rate() functions are not supported by the stats command in DFS.

Different results for stats functions in DFS

The results of the following stats functions can vary slightly from Splunk Enterprise when used with DFS search queries:

<table>
<thead>
<tr>
<th>Function type</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event order functions</td>
<td>first(), last()</td>
</tr>
<tr>
<td>Multivalue stats and chart functions</td>
<td>list (x)</td>
</tr>
<tr>
<td>Aggregate functions</td>
<td>estdc(), estdc_error(), upperperc&lt;int&gt;(), perc&lt;int&gt;(), median(), mode()</td>
</tr>
<tr>
<td>Time functions</td>
<td>earliest(), latest()</td>
</tr>
</tbody>
</table>

Some of the stats functions display different results because of one of the following reasons:

- DFS search is processed through a partitioned implementation of the search pipeline
- Ambiguity in the dataset.
  For example: The first() function obtains the first event that is received through the search. However, in a distributed partitioned implementation, multiple events may be received first - one on each of the nodes in the compute cluster. Similarly, the mode() function in DFS may have multiple values that share the most frequent value and any one of them may be returned in the search results. Similarly, for the functions earliest() and latest() DFS search results may differ due to events with the same timestamp.

Stats command behaves differently when combined with some other commands

The head and the tail commands also deviate from expected behavior when combined with the stats command in a DFS search. For more information on how the tail command operates in DFS, see tail. For more information on how the head command operates in DFS, see head.

For general information on the stats command, see stats in Splunk Enterprise Search Reference.

Examples

1. The mode() function displays different elements with the same highest value

In standard search, the mode() returns the most frequent value of the field X. However, when multiple values for a particular variable occur the same number of times, the results of the mode() function can vary from standard search results.

For example, if multiple values have the same number of occurrences for the field X as shown in the table, the following DFS search might return different search results from a standard search:

<table>
<thead>
<tr>
<th>X</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>57</td>
</tr>
<tr>
<td>X</td>
<td>Count</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>A</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
</tr>
<tr>
<td>D</td>
<td>4</td>
</tr>
</tbody>
</table>

\[
\text{dfsjob [ index = \textless indextype\textgreater \mid \text{stats mode}(X) \text{ BY } X]}
\]

The DFS search might return "A" while standard search returns "B".

2. **The earliest() and the latest() functions might behave differently**

In standard search, the `earliest()` function is used to locate the first value based on time order and the `latest()` function is used to locate the last value based on time order. However, in DFS the `earliest()` and the `latest()` functions might behave differently when multiple events have the same timestamp.

3. **Abbreviated form of the percentile aggregate function is not recognized**

DFS does not recognize the abbreviated form of the percentage function. Therefore, the following DFS search with the abbreviated `perc()` returns an error:

\[
\text{dfsjob [ search index=\text{network\_data} \mid \text{stats stdev(bytes\_in) min(bytes\_in)} \mid \text{max(bytes\_in) p90(bytes\_in)} \mid \text{by dst\_ip}}
\]

However, the following DFS search with the non-abbreviated version of the `perc()` returns expected results:

\[
\text{dfsjob [ search index=\text{network\_data} \mid \text{stats stdev(bytes\_in) min(bytes\_in)} \mid \text{max(bytes\_in) perc\text{90}(bytes\_in)} \mid \text{by dst\_ip}}
\]

**tail**

A DFS search using the `tail` command operates differently from standard search. Random results are displayed due to the partitioned implementation of the DFS search, even though the DFS search results are accurate. A standard search using the `tail` command displays the same search results every time a search is run.

Following is an example of a DFS search using the `tail` command:

\[
\text{dfsjob [search index=_\text{audit} \mid \text{stats count by host \mid tail 1}}
\]

For general information on the `tail` command, see `tail` in the Splunk Enterprise Search Reference.

**Examples**
1. *Use the sort with the tail command to order search results*

You may run a DFS search with `tail` command to return the last `N` number of specified results in reverse order as follows:

```
| dfsjob [search index="index_name" sourcetype="sourcetype_name" | tail N]
```

However, the search results displayed may be in random order. Including the `sort` command before the `tail` command in the search can display the results in order. The `stats` command included in the DFS search helps improve performance by reducing data that must be processed by the search.

```
| dfsjob [search index="index_name" sourcetype="sourcetype_name" stats count by <field> | sort <field> | tail N]
```

2. *DFS does not support streaming searches with tail command*

The following DFS search with the `tail` command will throw an exception error because DFS does not support streaming searches.

```
| dfsjob [ search index="index_name" | tail ]
```

3. *Use sort with tail command to display the same search results*

A DFS search using the `tail` command may display different test results each time a search is run. This is because in DFS the search requests are distributed into partitions. Therefore, though the DFS search results are still accurate, they may not display similarly every time a DFS search is run. This behavior is different from a traditional Enterprise search with a `tail` command where the same search results are displayed every time because the search flow is very definitive. To get the same results every time you run a DFS search with a `tail` command, use the `sort` command to organize your results.

```
| dfsjob [search index="index_name" sourcetype="sourcetype_name" stats count by <field> | sort <field> | tail N]
```

**union**

A DFS search using the `union` command operates differently from a standard search.

Following are some examples of DFS searches using the `union` command:

```
| dfsjob [ | union [search index="_internal"
| stats count as internal_count ][ search index="_audit"
| stats count as audit_count ]]
```

Using the `union` command, you can combine the results from two or more datasets. Following is an example of identifying overlapping IP address between two different intervals of time

```
| dfsjob [ | union [ search index=<source_indextype1> earliest=-2h latest=-1h
| eval marker=1
| stats count by marker, clientip ]
| search index=<source_indextype2> earliest=-1h latest=now
```
For general information on the union command, see union in the Splunk Enterprise Search Reference.

A standard search imposes certain limitations with the union command that are specified in the limits.conf configuration file. These include the maximum number of subsearches that can be combined, the maximum search time for the subsearch, and the maximum time to wait for the subsearch to complete. However, DFS does not have such limitations and ignores any values included for the following arguments: maxtime, maxout, and timeout. Using these arguments with the union command in a DFS search generates a UI warning, though the DFS search runs as expected. Thus, the subsearch arguments for the union command are not supported in DFS.

The search results using the union command in a DFS environment may differ from the search results run in a Splunk Enterprise environment. Standard streaming searches using the union command display results in a time order. For other search pipelines within standard search, the results of the union command are ordered serially based on its occurrence in the SPL search. For more information on how union commands are processed, see How the command is processed in the Splunk Enterprise Search Reference. However, in DFS the order of the search results is undefined because individual search pipelines using the union command are run in parallel.

where

A DFS search using the where command operates differently from a standard search and offers limited support for certain functions and operators.

Following are some examples of DFS searches using the where command:

| dfsjob [search index=_internal|where date_hour>11|stats count by date_hour|
| dfsjob [search index=_audit|stats count by date_hour |where count>100|

For general information on the where command, see where in the Splunk Enterprise Search Reference.

For more information on the evaluation functions supported by the where command, see Evaluation functions. For more information on the evaluation operators supported by the where command, see Evaluation operators.

Examples

1. Filter search results

Use the following DFS search to count the number of events with a combination of name and id, then filter the results when the count is greater than three.

| dfsjob [ search index="index_type" | stats count by name,id | where count>3]

2. Filter search results early for better performance

The following DFS search does not filter search results early:

| dfsjob [search index="index_name" | stats count by src,dst | where like(src, "10.9.165.%") OR cidrmatch("10.9.165.0/25", dst)]
To filter results earlier, use the following DFS search:

```
dfsjob [ search index="index_name" | where like(src, "10.9.165.%") OR cidrmatch("10.9.165.0/25", dst)|
        stats count by src,dst]
```

3. **DFS does not support streaming searches with the where command**

The following DFS search with the `where` command throws an exception error because DFS does not support streaming searches.

```
dfsjob [ search index="index_name" | where distance/time > 100]
```
Troubleshoot DFS

DFS search caveats

To optimize search performance, you must be aware of the following caveats when running big data analysis and federated searches:

Maximum number of concurrent searches

DFS allows you to run multiple concurrent searches. You can specify the maximum number of concurrent searches that can run on each search head by setting the value of the `dfc_num_slots` parameter in the `limits.conf` configuration file. The value must be in the range of four to ten concurrent searches. However, running multiple concurrent searches can impact the performance of the search. Thus, it is recommended that you use the default value of 4 to specify the number of concurrent searches. Additionally, if you already have a previously scheduled search and specify the maximum number of concurrent searches, the previously scheduled search will be overridden and an error will display on the search dashboard.

Search results display differently

DFS searches may display different search results from a standard search due to the partitioned execution of the search across DFS workers. Additionally, the order in which the results are displayed with a DFS search is different from a standard search.

For example: A DFS search using the `tail` command behaves differently from a standard search and does not display the same search results every time, even though the results are accurate. Use the `tail` command with the `sort` command to order the search results.

```
|dfsjob [search index="index_name" sourcetype="sourcetype_name" stats count by <field> | sort <field> | tail N]
```

A DFS search using the `head` command may display results in random order that is different from a similar standard search. Use the `head` command with the `sort` command to order search results. For example:

```
|dfsjob [search index="index_name" sourcetype="sourcetype_name" stats count by <field> | sort <field> | head N]
```

Also, a DFS search using the `sort` command may display the fields with same sort value in random order.

Only transforming searches are supported

Federated or big data analysis DFS searches do not support streaming searches or event based searches and may return an incorrect number of events.

Examples of supported big data analysis searches:

```
|dfsjob [search index=_internal | where log_level=INFO | stats count]
```

Examples of supported federated search queries:
Examples of unsupported big data analysis searches:

```
|dfsjob [search index=_internal | head 10 | stats count]
|dfsjob [search index=_internal | head 10]
|dfsjob [search index=_internal | tail 10]
|dfsjob [search index=_internal | where log_level=INFO]
```

Examples of unsupported federated search queries:

```
|dfsjob | from federated: remote_dataset | [remote_dataset] search = search index=ansible_access_combined | head 10 [remote_dataset] search = search index=ansible_access_combined | tail 10 [remote_dataset] search = search index=ansible_access_combined | head 10 | sort by bytes
```

**Federated searches across local indexes are not supported**

For federated searches, searches across local standalone Splunk Enterprise deployments and remote Splunk Enterprise deployments are not supported. Remote standalone Splunk Enterprise deployments are supported. However, fabric mode does support remote standalone Splunk Enterprise deployment. Additionally, for big data analysis, standalone Splunk Enterprise deployments are not supported.

**Use headN option to limit search results**

DFS only supports reporting searches by default. However, you can use the `headN` option with the `noop` command in streaming and non-reporting federated searches to retrieve a subset of events from the dataset on the remote federated providers to preview the data characteristics. This option may be used when you do not have the ability to explore the dataset as DFS only supports transforming searches. The `headN` option is useful in some situations to identify the characteristics of a large dataset e.g. to identify fields in the dataset.

To retrieve 2000 events from the following search:

```
| from federated:small_2000
```

where `small_2000` is defined as:

```
| search index=<source_type>
```

Use the `headN` option with the `noop` command to cap the search limit as follows:

```
|dfsjob | from federated:small_2000 | noop headN=2000
```

**Datamodels with dfsjob**

DFS does not support searches using datamodels that contain a `dfsjob` command.

Thus, if you define a datamodel "d2" as follows:

```
d2 = |dfsjob [search index = air* | stats by host]
```
The following SPL search using the datamodel will not return expected results because of the embedded `dfsjob` command defined in the datamodel.

```
|dfsjob [ | from datamodel:dm2]
```

translates to:

```
|dfsjob [ dfsjob [search index = air* | stats count by host]]
```

**Datamodels with extracted fields**

When you create a datamodel with a dataset that contains a transforming search, it returns no search results. Also, when you create a datamodel with `_time` as an extracted field, no search results are returned. If `_time` is not selected as an extracted field, search results are returned but a few extra empty columns are displayed.

To remove the extra empty columns from displaying in the search results, edit the configuration file `datamodels.conf` to set the `strict_fields` setting to `false`.

**Search command support with child datatsets**

Search commands are supported by the `dfsjob` command. However, the `search` command is not supported by DFS within a child dataset defined in a datamodel.

Thus, the following search:

```
|dfsjob [ | from datamodel:dm.rsSplunkdTransformingChild]
```

displays an error message indicating that the `search` command is not supported because it translates to:

```
| dfsjob [search index=ansible_access_combined | stats count by status |search some_filter |...]
```

**Eval command with curly brackets `{}`**

A federated search in direct mode does not support the `eval` command that uses curly brackets `{}`. However, a regular DFS search supports the `eval` command that uses curly brackets `{}`.

Thus the following SPL search throws as error:

```
|dfsjob [ | from federated:my_dep_l_search_with_eval_curly_brackets]
```

if the search `[my_dep_l_search_with_eval_curly_brackets]` is defined in `savedsearches.conf` as follows:

```
search=search index=<index_type> | stats count by host|eval {host}=count
federated.provider=remote_deployment_1
```

DFS does not support using curly brackets when using the value of another field as the name of the destination field.

**Self-join operations using same datasets**

Self join operations on the same dataset displays no search results in the fabric mode.
If we want to perform a `join` operation with the same dataset `<test>` using the following DFS SPL search, the federated search in fabric mode yields no results:

```
|dfsjob [|from federated:<test> |join left = L right = R where L.host=R.host [|from federated:<test>]]
```

**dfsjob command cannot be used in a subsearch**

A `dfsjob` command may not be used in a sub-search as DFS does not support sub-searches in the main search. Thus, the following search returns an error:

```
|dfsjob [search index = _audit [search index=_internal |head 1 | fields host] | stats count]
```

However, a generating command like `union` or `from` can be the first command to be used in sub-search for DFS searches to run correctly.

Thus, the following DFS search returns an error:

```
|dfsjob [search index=_audit|stats count by host] | union [search index=_audit|regex host=idx1|stats count by host]
```

However, if the `union` command is included inside the `dfsjob` command, the DFS search runs correctly.

For example:

```
|dfsjob [|union [search index=_audit|stats count by host] [search index=_audit|regex host=idx1|stats count by host]]
```

For a list of known issues, see DFS known issues.

### DFS log files

To troubleshoot your DFS deployment, use the Job Inspector to check the following log files in order to identify error messages:

<table>
<thead>
<tr>
<th>Log file</th>
<th>Location</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFS master (DFM) log</td>
<td>Search head</td>
<td>$SPLUNK_HOME/var/log/splunk/dfs.log</td>
</tr>
<tr>
<td>DFS coordinator (DFC) log</td>
<td>Search head</td>
<td>$SPLUNK_HOME/var/run/splunk/dispatch/&lt;remote search id&gt;/dfs.log</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>In the Job Inspector, navigate to Search Job Properties &gt; Additional Info to review the dfs.log file.</strong></td>
</tr>
<tr>
<td>DFS worker (DFW) log</td>
<td>DFS worker</td>
<td>On the DFS worker: $SPARK_HOME/work/&lt;search id&gt;/dfs.log</td>
</tr>
<tr>
<td></td>
<td></td>
<td>On the search head: $SPLUNK_HOME/dispatch/&lt;sid&gt;/phase_dfs/remote_logs</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>In the Job Inspector, navigate to Search Job Properties &gt; Additional Info to review the phase_dfs-&lt;ip address&gt;.log file.</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>One log file exists for each DFS worker.</td>
</tr>
<tr>
<td>Search head search log</td>
<td>Search head</td>
<td>$SPLUNK_HOME/var/run/splunk/dispatch/&lt;search id&gt;/search.log</td>
</tr>
</tbody>
</table>
In the Job Inspector, navigate to **Search Job Properties > Additional Info** to review the `search.log` file.

Log file contains basic information about the DFS job and job cancellation. The log file also identifies if the job is a valid DFS job.

<table>
<thead>
<tr>
<th>Log file</th>
<th>Location</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indexer search log</td>
<td>Indexer</td>
<td><code>$SPLUNK_HOME/var/run/splunk/dispatch/&lt;remote search id&gt;/search.log</code></td>
</tr>
</tbody>
</table>

In the Job Inspector, navigate to **Search Job Properties > Additional Info** to review the `phase-<ip address>.log` file.

If the log files are not available using the Job Inspector, log in to the corresponding system to check the log files. The DFM log file can be viewed only by logging in to the search head.

The following screenshot shows the location of the DFS logs in the Job Inspector.

In the Job Inspector, navigate to **Search Job Properties > Additional Info > dfs**, to check how your DFS search is broken down. The reduced search runs on the search head while the remote search runs on the indexers.

**Troubleshoot DFS**

If you have problems setting up your DFS deployment or running a DFS search, it can be due to several reasons. Use the following table to identify the common issues that might cause your DFS search to fail and to locate information on troubleshooting those issues:

<table>
<thead>
<tr>
<th>Issue</th>
<th>Troubleshooting information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Licensing issues.</td>
<td>Allocated vCPU usage limit for DFS searches has been exceeded. For more information on licensing requirements to run DFS searches, see Licensing information.</td>
</tr>
<tr>
<td>Set up issues with the DFS compute cluster.</td>
<td>For information on verifying if your compute cluster is set up correctly, see Troubleshoot the compute cluster.</td>
</tr>
<tr>
<td>Networking communications issues. Example: The indexers are unable to communicate with the DFS workers.</td>
<td>For information on troubleshooting network issues, see Troubleshoot network communication issues.</td>
</tr>
<tr>
<td>Invalid search due to commands unsupported by DFS.</td>
<td>Only some search commands are supported by DFS searches. For information on the commands supported by DFS, see Run DFS searches.</td>
</tr>
</tbody>
</table>
### Issue Troubleshooting information

**Security setup issues.**
Example: You want to enable security because TLS is not enabled by default for data transport within a DFS deployment. Example: Verify whether security is enabled on both search heads and indexers or is not enabled at all.

For information on securing your DFS deployment, see Secure a DFS deployment. For information on troubleshooting your security set up, see Troubleshoot security issues.

**Incorrect settings in the configuration files.**

For information on configuration settings to set up a big data analysis search, see Edit configuration files for big data analysis. For information on configuration settings to set up a federated search, see Edit configuration files for a federated search. For information on troubleshooting incorrect settings in configuration files, see Troubleshoot settings in configuration files.

**There is a discrepancy between DFS and SPL search results.**

Check the DFS search to verify if it is a non-transforming or a streaming search. Federated or big data analysis searches do not support streaming searches and may return incorrect number of events. For information on the search caveats, see DFS search caveats.

**Mismatch between the SPL search on the search bar and the log file.**

For information on troubleshooting SPL search issues, see Troubleshoot SPL search.

For information on issues that might impact a DFS or federated search, review the list of DFS Known issues.

Use DFS log files to troubleshoot issues with your DFS deployment. For more information, see DFS log files.

### .cfg files

Use the following table to identify the .cfg files, their location, and the properties that you must edit to configure the information levels of the DFS log files:

<table>
<thead>
<tr>
<th>.cfg file</th>
<th>Location</th>
<th>Editable property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>log-dfs-search.cfg</td>
<td>$SPLUNK_HOME/etc/</td>
<td>log4j.logger.com.splunk.df</td>
<td>DFS coordinator (DFC) and DFS worker (DFW) processes</td>
</tr>
<tr>
<td>log-dfs.cfg</td>
<td>$SPLUNK_HOME/etc/</td>
<td>log4j.logger.com.splunk.df</td>
<td>DFS master (DFM) processes</td>
</tr>
<tr>
<td>log.cfg</td>
<td>$SPLUNK_HOME/etc/</td>
<td>category.DFSMaster</td>
<td>splunkd processes</td>
</tr>
<tr>
<td>log-searchprocess.cfg</td>
<td>$SPLUNK_HOME/etc/</td>
<td>rootCategory</td>
<td>Splunk search process logs</td>
</tr>
</tbody>
</table>

### Information levels in the .cfg files

Use the following table to identify the various logging levels that you can select based on the amount of information you want to display in the log files:

<table>
<thead>
<tr>
<th>Information level</th>
<th>Description</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFO</td>
<td>Provides basic information as the search is run.</td>
<td>Default logging level.</td>
</tr>
<tr>
<td>DEBUG</td>
<td>Provides comprehensive information on code execution paths.</td>
<td>Used for troubleshooting.</td>
</tr>
<tr>
<td>WARN</td>
<td>Helps to identify unusual behavior related to applications.</td>
<td>Most issues can be resolved automatically.</td>
</tr>
<tr>
<td>ERROR</td>
<td>Identifies fatal issues as the code is run.</td>
<td>Used to review errors that caused the search to fail.</td>
</tr>
</tbody>
</table>

Set the information level in the log file to what you need. Increasing the information level of log files may impact DFS performance.

67
Troubleshoot network communication issues

Troubleshoot network issues by performing the following verification checks:

On the search.log file:

- Verify that the IP addresses of the DFS workers in the log file match the IP addresses of the corresponding worker machines.
- Verify that the compute cluster nodes have ssh passwordless login enabled. This is not required but might impact communication.
- Verify that the DFS master is running by checking that the compute cluster master UI is running. You can also verify that the DFS master is running by typing the following command on the command line to display all the running Java processes:

  ```bash
  ps -ef | grep java
  ```

  The following screenshot illustrates the compute cluster master UI:

On the .cfg file:

- Verify if the indexers are able to communicate with the DFS workers.
- Verify that Splunk Enterprise is able to communicate with the DFS master (DFM) by verifying that the DFM log file located on the search head at `$SPLUNK_HOME/var/log/splunk/dfs.log` is set up correctly.

  Check the DFM log file for the following response from the DFM to the Search Processor (SP):

  ```plaintext
  ```

- Verify that the compute cluster is running by verifying whether the search request has been dispatched from SP to DFC and the search has been scheduled on the compute cluster using the DFS coordinator (DFC) log file located at `$SPLUNK_HOME/var/run/splunk/dispatch/<remote search id>/dfs.log`.

  Check the DFC log file for the following response from the SP to the DFC:

  ```plaintext
  03-05-2019 21:31:38.361 INFO DistributedSearchResultCollector - Send RunSearch to DFC
  ```

For information on DFS log files, see DFS log files. For information on the .cfg files, see .cfg files.

Troubleshoot settings in the configuration files

A DFS or a federated search might not run if the fields in the configuration files are not set up correctly. Review the following settings in the configuration files to ensure that searches run correctly.

68
Verify that the IP address of the compute cluster master host is set up correctly in the `server.conf` configuration file in the field `spark_master_host`.
- Verify if DFS is enabled by verifying that the `disabled` field in the `server.conf` configuration file is set to `false`.
- Verify that the number of concurrent searches that you run is less than or equal to the value set for the field `dfc_num_slots` in the `limits.conf` file.
- Verify that the number of concurrent searches that you run is less than or equal to the value set for the field `dfw_num_slots` in the `limits.conf` file.
- Ensure that the values displayed in the `.cfg` files are the same as the values set for the settings in the configuration files.
  Example: Check that the values for the following fields in the `limits.conf` file: `dfc_control_port`, `dfw_receiving_data_port`, `dfc_num_slots`, `estimatedEventCount` are the same as displayed by the request from the search process in the DFM log file.

For information on configuration settings to set up a DFS search, see Edit configuration files to set up a DFS search. For information on configuration settings to set up a federated search, see Edit configuration files for a federated search.

**Troubleshoot the SPL search**

Troubleshoot the SPL search using the `.cfg` log files to ensure that the search in the log file matches the search specified in the search bar on Splunk Web.

Example:
The SPL search on the search bar is the same search displayed in the response from the DFC to SP process in the log file:
```
DFC response = {"orderInfo":[],"search":"| dfsjob [| search index=_internal | stats count]","...}
```

**Troubleshoot using Splunk DFS Manager**

If you have configured your compute cluster using the Splunk DFS Manager app, you can use the app to troubleshoot some of the common issues and error messages. For more information on troubleshooting using the app, see Troubleshoot Splunk DFS Manager. For a list of error messages displayed by the app, see Splunk DFS Manager error messages.

**Troubleshoot security issues**

Check log files to identify error messages and troubleshoot security issues when you set up a DFS deployment. For more information on DFS log files, see DFS log files. For more information on securing your DFS deployment, see Secure a DFS deployment.

Transport layer security (TLS) is not enabled by default for data transport within a DFS deployment.

**Partially enabled security on the indexer or the remote search head**

*Problem*

Following error messages in the indexer search log are displayed if DFS security is partially enabled on the indexer or the remote search head:

```
● ERROR ROutProcessor - DfsOutputSerializerSecure:: couldn't connect for hostAndport=10.224.33.54:17501
```

69
Cause

For the big data analysis use case, DFS security is enabled only on the indexers, but not on the search head. For the federated search use case, DFS security is enabled only on the remote search heads, but not on the federated search head.

Solution

For the big data analysis use case, enable DFS security on the search head. For the federated search use case, enable DFS security on the federated search head.

1. Log in to the search head or the federated search head.
2. Edit the server.conf configuration file located at $SPLUNK_HOME/etc/system/local/server.conf to ensure Splunk Enterprise security is enabled. Splunk Enterprise security is enabled by default.
3. Edit the server.conf configuration file located at $SPLUNK_HOME/etc/system/local/server.conf to ensure DFS security is enabled. DFS security is disabled by default.
4. Restart Splunk Enterprise to enable the changes made to the server.conf configuration file.

Partially enabled security on the search head or the federated search head

Problem

The following error messages in the DFS worker (DFW) log are displayed if DFS security is partially enabled on the indexer or the remote search head:

- ERROR NIO:287 - Unrecognized SSL message, plaintext connection?
- ERROR NIO:993 - rdinAccept Failed handshake:: Handshake failed during NEED_UNWRAP
- ERROR NIO:SearchResultsReceiverManager:574 - This is not a good error, the nio base loop should not get any error: Handshake failed during NEED_UNWRAP

Cause

For the big data analysis use case, DFS security is enabled only on the search head, but not on the indexers. For the federated search use case, DFS security is enabled only on the federated search head, but not on the remote search heads.

Solution

For the big data analysis use case, enable DFS security on the indexers. For the federated search use case, enable DFS security on the remote search heads and local indexers.

1. Log in to the indexers or the remote search heads.
2. Edit the server.conf configuration file located at $SPLUNK_HOME/etc/system/local/server.conf to ensure DFS security is enabled. DFS security is disabled by default.
3. Restart Splunk Enterprise to enable the changes made to the server.conf configuration file.
Invalid Keystore and Truststore path in mode 1

**Problem**
The following error messages in the DFS master (DFM) log file are displayed if the Keystore and Truststore path is invalid in mode 1:

- ERROR DFCLauncherEmbedded:113 - DFS keystore needs to be configured properly.
- ERROR DFCLauncherEmbedded:113 - DFS truststore needs to be configured properly.

**Cause**
The Keystore and Truststore path specified in mode 1 is invalid.

**Solution**
1. Configure the correct path for the following parameters in the server.conf configuration file located at `$SPLUNK_HOME/etc/system/local/server.conf`.
   - [dfs_security]
     - dfs_keystore_path
     - dfs_truststore_path

2. Restart Splunk Enterprise to enable the changes made to the server.conf configuration file.

Invalid Truststore password in mode 1

**Problem**
The following error message in the DFS worker (DFW) log file is displayed if the Truststore password is incorrect:

```
ERROR NIOTls:245 - Truststore failed.
```

**Cause**
The Truststore password specified is incorrect.

**Solution**
1. Configure the correct password for the following parameter in the server.conf configuration file located at `$SPLUNK_HOME/etc/system/local/server.conf`. You must delete any previous values for this parameter. The password is encrypted when you restart Splunk Enterprise.
   - [dfs_security]  
     - dfs_truststore_password

2. Restart Splunk Enterprise to enable the changes made to the server.conf configuration file.

Invalid Key password in mode 1

**Problem**
The following error messages in the DFS coordinator (DFC) log file are displayed if the Key password is incorrect:

- SSLContext Failed. Please check the provided keystore, truststore, and corresponding password.
- This error is fatal, this error should not occur under any circumstances other than dfc shutdown:
  - java.security.UnrecoverableKeyException:
- ERROR NIOTls:934 - Closing keyStoreInfoProviderThread server socket 1550095541.4
Cause

The Key password specified is incorrect.

Solution

1. Configure the correct password for the following parameter in the server.conf configuration file located at $SPLUNK_HOME/etc/system/local/server.conf. You must delete any previous values for this parameter. The password is encrypted when you restart Splunk Enterprise.

   [dfs_security]
   dfs_key_password

2. Restart Splunk Enterprise to enable the changes made to the server.conf configuration file.

Invalid DFW Keystore and Truststore path in mode 3

Problem

The following error messages in the DFS worker (DFW) log file are displayed if the Keystore and Truststore path is invalid in mode 3:

- ERROR NIOTls:964 - SSLContext Failed. Please check the provided keystore, truststore, and corresponding password.
- ERROR NIOSearchResultsReceiverManager:574 - This is not a good error, the nio base loop should not get any error:

Cause

The Keystore and Truststore path specified in mode 3 is invalid.

Solution

1. Configure the correct path for the following parameters in the server.conf configuration file located at $SPLUNK_HOME/etc/system/local/server.conf.

   [dfs_security]
   dfw_keystore_path
dfw_truststore_path

2. Restart Splunk Enterprise to enable the changes made to the server.conf configuration file.

Invalid DFW Keystore password in mode 3

Problem

The following error messages in the DFS worker (DFW) log file are displayed if the Keystore password is incorrect in mode 3:

- ERROR NIOTls:227 - Keystore failed.
- ERROR NIOTls:964 - SSLContext Failed. Please check the provided keystore, truststore, and corresponding password.
- ERROR NIOSearchResultsReceiverManager:574 - This is not a good error, the nio base loop should not get any error:
**Cause**

The Keystore password specified is incorrect in mode 3.

**Solution**

Configure the correct password for the field `dfw_keystore_password` in the `dfs_security` stanza of the `server.conf` configuration file. You must delete any previous values for this parameter. The password is encrypted when you restart Splunk Enterprise.

1. Configure the correct password for the following parameter in the `server.conf` configuration file located at `$SPLUNK_HOME/etc/system/local/server.conf`. You must delete any previous values for this parameter. The password is encrypted when you restart Splunk Enterprise.
   ```server.conf```
   ```
   [dfs_security]
   dfw_keystore_password
   ```
2. Restart Splunk Enterprise to enable the changes made to the `server.conf` configuration file.

**Invalid DFW Truststore password in mode 3**

**Problem**

The following error messages in the DFS worker (DFW) log file are displayed if the Truststore password is incorrect in mode 3:

- `ERROR NIOTls:245 - Truststore failed.`
- `ERROR NIOTls:964 - SSLContext Failed. Please check the provided keystore, truststore, and corresponding password.`
- `ERROR NIOSearchResultsReceiverManager:574 - This is not a good error, the nio base loop should not get any error:```

**Cause**

The DFW Truststore password specified is incorrect in mode 3.

**Solution**

1. Configure the correct password for the following parameter in the `server.conf` configuration file located at `$SPLUNK_HOME/etc/system/local/server.conf`. You must delete any previous values for this parameter. The password is encrypted when you restart Splunk Enterprise.
   ```server.conf```
   ```
   [dfs_security]
   dfw_truststore_password
   ```
2. Restart Splunk Enterprise to enable the changes made to the `server.conf` configuration file.

**Invalid DFW Key password in mode 3**

**Problem**

The following error messages in the DFS worker (DFW) log file are displayed if the Key password is incorrect in mode 3:

- `ERROR NIOTls:964 - SSLContext Failed. Please check the provided keystore, truststore, and corresponding password.`

73
ERROR NIOSearchResultsReceiverManager:574 - This is not a good error, the nio base loop should not get any error:

**Cause**

The DFW Key password specified is incorrect in mode 3.

**Solution**

1. Configure the correct password for the following parameter in the server.conf configuration file located at $SPLUNK_HOME/etc/system/local/server.conf. You must delete any previous values for this parameter. The password is encrypted when you restart Splunk Enterprise.

   ```
   [dfs_security]
   dfw_key_password
   ```

2. Restart Splunk Enterprise to enable the changes made to the server.conf configuration file.

**Common and alternative name checks failed**

**Problem**

The following error messages in the DFS worker (DFW) log file are displayed if an unauthorized indexer attempts to connect to the DFS worker (DFW):

```
• ERROR NIOTls:981 - Common and Alternative name check failed
• ERROR NIOTls:993 - rdinAccept Failed handshake:: Rdout to DFW TLS certificate common name and alternative check failed
• ERROR NIOSearchResultsReceiverManager:574 - This is not a good error, the nio base loop should not get any error:
```

**Cause**

An unauthorized indexer attempts to connect to the DFS worker (DFW).

**Solution**

This error is expected if a non-authorized indexer is trying to connect to the DFS worker.

1. Check the following parameters in the server.conf configuration file located at $SPLUNK_HOME/etc/system/local/server.conf.

   ```
   [dfs_security]
   search_peer_to_dfw_common_name_list
   search_peer_to_dfw_alt_name_list
   ```

   The common name and the alternative name for the authorized indexer's certificate should be listed in the search_peer_to_dfw_common_name_list and the search_peer_to_dfw_alt_name_list. Wildcard (*) characters are not supported in these lists.

2. Restart Splunk Enterprise to enable the changes made to the server.conf configuration file.

**Troubleshoot federated searches**

If you have problems running federated searches, it can be due to several reasons. Use the following table to identify the common issues that might cause your federated searches to fail and to locate information on troubleshooting those
issues:

## Troubleshoot federated searches

<table>
<thead>
<tr>
<th>Issue</th>
<th>Troubleshooting information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network connectivity issues with federated providers. Example: You are unable to connect to the remote federated providers.</td>
<td>For information on network connectivity issues, see Troubleshoot network connectivity issues.</td>
</tr>
<tr>
<td>Data accessibility issues. Example: You are unable to access the data from the remote federated providers.</td>
<td>For information on data accessibility issues, see Troubleshoot data accessibility issues.</td>
</tr>
<tr>
<td>Security issues. Example: Security is disabled</td>
<td>Verify that security is enabled. Security is disabled by default. To enable security, set tls_enabled = true in the [dfs_security] stanza of the server.conf configuration file on the search heads, the local indexers, and the indexers of all the remote federated providers.</td>
</tr>
<tr>
<td>Accessibility to remote federated providers. Example: You do not have access rights to the remote federated providers due to limited role capabilities of fsh_search</td>
<td>Check whether you have access to the remote federated providers. Also, check if the service accounts you created are valid.</td>
</tr>
<tr>
<td>Federated search mode is not supported</td>
<td>Check the Splunk Enterprise version of the federated deployment and the remote federated providers. Federated deployments running on a versions lower than 8.0 will failback to sh mode. For more information on setting the mode for federated searches, see Set the mode for federated search.</td>
</tr>
<tr>
<td>Performance issues</td>
<td>Check the federated deployment and the remote federated providers for backward compatibility with older Splunk Enterprise deployments. For optimal performance, upgrade your Splunk Enterprise deployments to version 8.0 so that it can run on fabric mode for optimal search performance. For more information on the modes to run federated searches, see Modes to run a federated search.</td>
</tr>
</tbody>
</table>

### Troubleshoot network connectivity issues

Troubleshoot network connectivity issues by performing the following verification checks:

- Verify that you can connect to the federated providers by checking for network barriers like firewalls, iptables, host file aliases, and so on.
- Verify that you are able to connect to the remote federated providers by running a REST query from the DFS workers to the remote federated providers.
- Verify the federated search defined in the savedsearches.conf file for the federated search head runs on the search head of the remote federated provider when the same service account is used.

In the Job Inspector, you can also navigate to Search Job Properties > Additional Info > dfsFshMetrics to view information on the remote federated provider and the corresponding searches run on each remote federated provider. The information may include the time taken to run a search or the number of events scanned, which may also enable you to identify performance issues.

### Troubleshoot data accessibility issues

Troubleshoot data accessibility issues by performing the following verification checks:

- Verify that the user accounts have permissions to access the datasets included in the federated search.
- Verify that the appropriate capabilities fsh_search or fsh_manage are enabled.
To troubleshoot access issues on the remote deployment, you can identify the user who invoked the search on behalf of the service account. Navigate to Search Job Properties > Additional Info > request > original_user = admin.

For information on issues that might impact a federated search, review the list of Known issues.

For information on search caveats for running federated searches, see Search caveats.

Use DFS log files to troubleshoot issues with your DFS deployment. For more information, see DFS log files.

**DFS Error Messages**

Use the following table to identify DFS specific error messages that are displayed in the \$SPLUNK_HOME/var/run/splunk/dispatch/<sid>/search.log file:

<table>
<thead>
<tr>
<th>Error code</th>
<th>Error message</th>
<th>Workaround</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFC_CONNECTION_ERROR</td>
<td>Connection Error. The Data Fabric Coordinator might have unexpectedly terminated or reset the connection.</td>
<td>Start a new search process by creating a new SPL search. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>DFS:DFC_INVALID_RESPONSE</td>
<td>Invalid Response. The Data Fabric Coordinator might have unexpectedly terminated or reset the connection.</td>
<td>Start a new search process by creating a new SPL search. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>DFMASTER_CONNECTION_ERROR</td>
<td>Connection Error. The Data Fabric Master might have unexpectedly terminated or reset the connection.</td>
<td>Start a new search process by creating a new SPL search. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>DFMASTER_INVALID_RESPONSE</td>
<td>Invalid Response. The Data Fabric Master might have unexpectedly terminated or reset the connection.</td>
<td>Start a new search process by creating a new SPL search. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>DFS_DISABLED</td>
<td>This search cannot run. The Data Fabric Search feature has been disabled.</td>
<td>Check if DFS is enabled by verifying that the disabled parameter in the server.conf configuration file is set to false.</td>
</tr>
<tr>
<td>DFS_INVALID_SPL</td>
<td>This search cannot run. Check the syntax for the dfsjob command.</td>
<td>Check the DFS supported commands on how to run a DFS search. For more information on the dfsjob command, see dfsjob.</td>
</tr>
<tr>
<td>DFS_CURSORED_SEARCH_ERROR</td>
<td>DFS only supports transforming searches currently.</td>
<td>For more information, see Only transforming searches are supported.</td>
</tr>
<tr>
<td>DFS_FEDERATED_LICENSE_MISSING</td>
<td>You do not have the appropriate license to run a federated data fabric search across another Splunk Enterprise deployment.</td>
<td>Contact Splunk Sales to obtain the license. For more information, see Install DFS license.</td>
</tr>
<tr>
<td>DFS_PREMIUM_FEDERATED_LICENSE_MISSING</td>
<td>You do not have the appropriate license to run a federated data fabric search across multiple Splunk Enterprise deployments.</td>
<td>Contact Splunk Sales to obtain the license. For more information, see Install DFS license.</td>
</tr>
<tr>
<td>DFS_INFRA_ERROR</td>
<td>An error occurred while setting up infrastructure for this data fabric search. Review your search logs for more information.</td>
<td>Start a new search process by creating a new SPL search. Otherwise, restart Splunk Enterprise. For more information, see Infrastructure requirements.</td>
</tr>
<tr>
<td>Error code</td>
<td>Error message</td>
<td>Workaround</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DFS_INVALID_DFC_IP</td>
<td>Invalid host IP address of the Data Fabric Coordinator (DFC) process.</td>
<td>Review DFS infrastructure requirements. For more information, see Infrastructure requirements.</td>
</tr>
<tr>
<td>DFS_LICENSE_MISSING</td>
<td>You do not have the appropriate license to run a data fabric search.</td>
<td>Contact Splunk Sales to obtain the license. For more information, see Install license.</td>
</tr>
<tr>
<td>DFS_NOT_SUPPORTED</td>
<td>This search cannot run. Review your search logs for more information.</td>
<td>Start a new search process by creating a new SPL query. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>DFS_SEARCH_EXCEPTION</td>
<td>An error occurred while fetching results for this data fabric search. Review the search logs for more information.</td>
<td>Start a new search process by creating a new SPL query. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>INVALID_CONF_SETTING__S</td>
<td>Invalid configuration setting for '%s' in limits.conf.</td>
<td>For more information on configuration settings for a DFS search, see Edit configuration files for a DFS search.</td>
</tr>
<tr>
<td>FIRST_COMMAND_EXCEPTION__S</td>
<td>The '%s' command can only be used as the first command on a search.</td>
<td>For information on DFS supported search commands, see Run DFS searches.</td>
</tr>
<tr>
<td>DFSJOB_PROCESSOR_EXPECT_ARGUMENT</td>
<td>DFSJob Processor expects subsearch as the only argument.</td>
<td>For information on DFS supported search commands, see Run DFS searches.</td>
</tr>
<tr>
<td>DFS_UNSUPPORTED_COMMAND_FOUND</td>
<td>Unsupported command found in dfsjob.</td>
<td>For information on DFS supported search commands, see Run DFS searches.</td>
</tr>
<tr>
<td>REAL_TIME_SEARCH_NOT_SUPPORTED</td>
<td>Real-time search is not supported by the Data Fabric Search feature.</td>
<td>For information on search caveats, see Search caveats.</td>
</tr>
<tr>
<td>DFS_NOT_SUPPORT_EXTERNAL_COMMANDS</td>
<td>Data Fabric Search does not support external commands currently.</td>
<td>For information on DFS supported search commands, see Run DFS searches.</td>
</tr>
<tr>
<td>INVALID_DFS_UNION_OPTION__S</td>
<td>DFS union search limits are not affected by '%s'. DFS Search will run without this option.</td>
<td>For information on DFS supported search commands, see Run DFS searches.</td>
</tr>
<tr>
<td>DFSJOB_NOT_SUBSEARCH</td>
<td>Unsupported usage: dfsjob cannot be used as a subsearch.</td>
<td>The dfsjob command cannot be used as a subsearch. Workaround: The union command must be included inside the dfsjobs command so that the DFS search can run. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS_UNSUPPORTED_COMMAND__S</td>
<td>Unsupported command '%s' in dfsjob.</td>
<td>For information on DFS supported search commands, see Run DFS searches.</td>
</tr>
<tr>
<td>FEDERATED_SAVEDSEARCH_NOT_SUPPORTED</td>
<td>Cannot run a federated search with 'savedsearch'. Use the from command instead.</td>
<td>For more information on the from command, see from.</td>
</tr>
<tr>
<td>ESTIMATED_EVENT_COUNT_ZERO</td>
<td>This search cannot run. Event count estimation failed.</td>
<td>Start a new search process by creating a new SPL query. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>MISSING_EVENTCOUNT_EXPRESSION</td>
<td>Missing expression to estimate event count.</td>
<td>Start a new search process by creating a new SPL query. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>Error code</td>
<td>Error message</td>
<td>Workaround</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CANNOT_SCHEDULE_FEDERATED_SEARCH</td>
<td>Scheduling a federated search is not allowed at this time.</td>
<td>Start a new search process by creating a new SPL query. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>CANNOT_AUTOSUMMARIZE_FEDERATED_SEARCH</td>
<td>Auto summarize of a federated search is not allowed at this time.</td>
<td>Start a new search process by creating a new SPL query. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>INVALID_FEDERATED_SAVEDSEARCH__S_S</td>
<td>Required field ‘%s’ is missing in stanza ‘%s’.</td>
<td>Start a new search process by creating a new SPL query. Otherwise, restart Splunk Enterprise.</td>
</tr>
<tr>
<td>INVALID_SPARK_HOME_SETTING</td>
<td>Invalid or empty value for the SPARK_HOME configuration setting.</td>
<td>Review the <code>spark_home</code> configuration setting that sets the location of Spark home in the <code>server.conf</code> file.</td>
</tr>
<tr>
<td>INVALID_SPARK_APP_SETTING</td>
<td>Unable to read the [pbs] stanza from the <code>spark_app.conf</code> configuration file.</td>
<td>Review the <code>spark_app.conf</code> configuration file settings.</td>
</tr>
</tbody>
</table>
Third-party software

Splunk DFS Manager app contains some libraries that were written by others and are being redistributed as part of the app under their respective open source licenses. For more information, see Third-party software in the Splunk DFS Manager User Manual.

Additionally, some of the components included in Splunk Data Fabric Search are licensed under free or open source licenses. We wish to thank the contributors to those projects. For complete information, see Third-party software in the Splunk Enterprise Release Notes.
Release notes

What's new in this release

The following table highlights the new features in the Data Fabric Search (DFS) release version 1.1.1:

<table>
<thead>
<tr>
<th>Number</th>
<th>Feature</th>
<th>Description</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Licensing changes</td>
<td>A warning message is displayed to enforce DFS license if the search exceeds the licensed vCPU limit.</td>
<td>For more information on DFS licensing, see Licensing information.</td>
</tr>
</tbody>
</table>

Known issues

Use the following list to review the known issues and workarounds for Splunk Data Fabric Search (DFS) release 1.1.1:

<table>
<thead>
<tr>
<th>Date filed</th>
<th>Issue Number</th>
<th>Description</th>
<th>Workaround</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020-03-23</td>
<td>SPL-185261</td>
<td>Changes may not be applied if the DFS add on license file is the first to be added in a licensing group.</td>
<td>Restart Splunk Enterprise if you added the DFS add-on license file as the first in a licensing group to apply the changes.</td>
</tr>
<tr>
<td>2019-11-08</td>
<td>SPL-179286</td>
<td>Partial search results may not be accompanied by an error message that indicates the results are incomplete and therefore, inaccurate. This may cause confusion about the accuracy of the search results.</td>
<td></td>
</tr>
<tr>
<td>2019-09-23</td>
<td>SPL-176901</td>
<td>The dfsjob search parser fails with the bin command.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thus, the following DFS search will fail:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dfsjob [</td>
<td>search index = &quot;_internal&quot;</td>
</tr>
<tr>
<td>2018-12-12</td>
<td>SPL-157613</td>
<td>A DFS search using the join command does not return the same event count as a standard search because the default value for the &quot;max&quot; join option is set to 1.</td>
<td>Set the &quot;max&quot; join option to 0, when comparing standard search results to DFS search using the join command.</td>
</tr>
<tr>
<td>2018-12-10</td>
<td>SPL-163776</td>
<td>DFS search capacity is only limited by the amount of hardware resources. For example: the RAM specifications or vCPUs for DFS workers or the number of DFS workers. Therefore, DFS search queries ignore any user input values for the &quot;maxout&quot;, &quot;maxtime&quot;, and &quot;timeout&quot; arguments with the union command and run as expected. However, using these arguments with the union command in a DFS search generates a UI warning.</td>
<td></td>
</tr>
<tr>
<td>2018-12-05</td>
<td>SPL-163546</td>
<td>A DFS search using the tail command behaves differently from a standard search and does not display the same search results every time, even though the results are accurate.</td>
<td>Use the tail command with the sort command to order the search results. For example: dfsjob [search index=&quot;index_name&quot; sourcetype=&quot;sourcetype_name&quot; stats count by &lt;field&gt;</td>
</tr>
<tr>
<td>2018-11-18</td>
<td>SPL-162957</td>
<td>The dfsjob command cannot be used as a subsearch.</td>
<td></td>
</tr>
<tr>
<td>Date filed</td>
<td>Issue Number</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>--------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>2018-11-02</td>
<td>SPL-162512</td>
<td>A federated search performed on data from multiple remote deployments will only display data on the federated search head. The remote search results will not display on the remote deployments. Therefore, trying to export or share the results on the individual remote deployments will not work. Workaround: Not required because the final search results on the federated search head are accurately displayed.</td>
<td></td>
</tr>
<tr>
<td>2018-08-30</td>
<td>SPL-159490</td>
<td>A DFS search using the <code>sort</code> command may display the fields with same sort value in random order.</td>
<td></td>
</tr>
<tr>
<td>2018-07-13</td>
<td>SPL-157267</td>
<td>A DFS search using the <code>head</code> command may display results in random order that is different from a similar standard search. Workaround: Use the <code>head</code> command with the <code>sort</code> command to order search results. For example:</td>
<td></td>
</tr>
<tr>
<td>2018-01-25</td>
<td>SPL-158950</td>
<td>Federated searches auto-cancel if the remote deployment is DFS. Thus, the <code>dfsjob</code> command cannot be used in federated search. Workaround: Do not set up DFS as a remote deployment in a federated search.</td>
<td></td>
</tr>
<tr>
<td>2018-01-16</td>
<td>SPL-148212</td>
<td>A DFS search returns an error message unless a source index is explicitly stated in the DFS search. Thus, the following search expects a source index</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>`</td>
<td>stats stdev(kbps)`</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and returns an error message of &quot;index detection failed&quot;. Workaround: Specify the source index name in a DFS search using the <code>dfsjob</code> command.</td>
<td></td>
</tr>
<tr>
<td>2018-07-26</td>
<td>SPL-157955</td>
<td>DFS does not support field extraction. Workaround: Add the <code>fields</code> command so that fields are automatically extracted after the raw search.</td>
<td></td>
</tr>
<tr>
<td>2018-07-09</td>
<td>SPL-156884</td>
<td>A DFS search during a rolling restart in an indexer clustering environment returns an error message without results.</td>
<td></td>
</tr>
<tr>
<td>2019-02-04</td>
<td>SPL-165970</td>
<td>Federated or high cardinality DFS searches do not support streaming searches and may return an incorrect number of events.</td>
<td></td>
</tr>
<tr>
<td>2019-02-12</td>
<td>SPL-163821</td>
<td>DFS may not strictly adhere to the concurrent search limits set by the configuration parameters &quot;dfw_num_slots&quot; and &quot;dfw_num_slots_enabled&quot; in a search head cluster deployment. This can happen when searches are initiated in rapid succession from different search head clusters around the same time frame. However, this is a relatively rare scenario and should not impact DFS functionality in a search head cluster.</td>
<td></td>
</tr>
<tr>
<td>2018-11-05</td>
<td>SPL-162574,</td>
<td>The <code>Share</code> and <code>Export</code> buttons on the <code>Jobs</code> page of the remote deployment are not supported when the search is run from the federated provider as a remote job.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SPL-168125</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A DFS search may change to a splunkd search upon clicking the **Exclude from results** icon.

In the `server.conf` file, if the parameter in the [dfs_security] stanza "tls_enabled = true" for a DFS search, then set the parameter in the [sslConfig] stanza to "enableSplunkdSSL=true", which is the default setting.

DFS does not support using curly brackets when using the value of another field as the name of the destination field.

The `inputlookup` command is not supported by federated searches.

The `stats` command does not support the following functions on DFS: earliest_time(), latest_time(), rate(). Thus, the following DFS search fails with an error message "Time function is not supported":

```
dfsjob[search index=* | stats earliest_time(action)]
```

DFS coordinator (DFC) cannot start and returns an error if the user does not have write permissions to the Spark local directory. The Spark local directory is a comma separated list of directories used for temporary storage and includes map output files and RDDs stored on disks.

In federated searches, a DFS search can be stuck in the parsing stage if the remote job is in a queued status because the disk quota has been reached.

Workaround: Perform the following actions on the remote deployment:

- Increase the disk quota of the search artifacts.
- Use the **Job Manager** to delete some search artifacts.

DFS does not support sub-searches in the main search. Thus, the following search returns an error:

```
|dfsjob [search index = _audit [search index=_internal |head 1 | fields host] | stats count]
```

Search results for the `stats` function `perc95()` are different for DFS and Splunk Enterprise.

Workaround: Use `exactpercX(Y)` function instead of `perc95()` to get more accurate results.

Events viewer in the UI search results table returns unexpected results for DFS searches.

The following known issue is fixed in the Splunk Data Fabric Search (DFS) release 1.1.1:

Eval operations in DFS are rounding results to the nearest integers instead of displaying results as floating point numbers that is expected in Enterprise search.

For example: The following SPL search with eval command returns results in floating point numbers like: 0.04730; 1.101; 0.9965; 1.219;

```
index=_internal | stats count as mycount by splunk_server | eval ratio=(mycount*100.0)/(367888*1.0)
```

However, the following DFS SPL search returns the results rounded to the nearest integer like: 1; 0; 0; 0;
<table>
<thead>
<tr>
<th>Date filed</th>
<th>Issue Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-07-12</td>
<td>SPL-157203, SPL-167457, SPL-167419</td>
<td>In federated searches, the time range button on Splunk Web UI applies only to local searches on the federated search head. For remote searches on the federated providers, the time range is &quot;all time&quot; by default. Workaround: To select a time range for remote searches in a federated search deployment, specify the time range in the search using Unix epoch time. For example, <code>index=_audit earliest=&quot;1551913043&quot; latest=&quot;1551914043&quot;</code></td>
</tr>
</tbody>
</table>
Appendix 1: DFS REST endpoints

**fshpasswords**

/services/storage/fshpasswords

https://<host>:<mPort>/services/services/storage/fshpasswords

Use this endpoint to create, update, delete, and retrieve the user credentials associated with a DFS federated provider.

The password credential is encrypted with a secure key resident on the same server and is the only part of the user credentials that is stored securely. The services/storage/fshpasswords endpoint differs in functionally from the /services/storage/passwords endpoint because it does not allow a listing of all the credentials. Credential information is obtainable only if the user and the corresponding federated provider information are provided.

**Authentication and authorization**

This endpoint is restricted to roles that have the fsh_manage and the fsh_search capabilities. For more information on the roles used for federated searches, see Roles to run federated searches.

**Usage details**

For information about setting up user accounts for access control in a federated search deployment, see Set up user accounts for access control.

**GET**

Disabled.

**Request parameters**

None.

**Returned parameters**

None.

**Example request and response**

**XML request**

```
curl -k -u admin:changeme -X GET https://localhost:8089/services/storage/fshpasswords
```

**XML response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
<messages>
  <msg type="ERROR">Invalid action for this internal handler (handler: fshpasswords, supported: create|edit|remove|_reload|new|doc, wanted: list).</msg>
</messages>
</response>
```
POST

Configures user credentials for the federated providers.

**Request parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>provider</td>
<td>required</td>
<td>String Name of the DFS federated provider.</td>
</tr>
<tr>
<td>name</td>
<td>required</td>
<td>String Name of the account user to access the federated provider.</td>
</tr>
<tr>
<td>password</td>
<td>required</td>
<td>String Password of the account user.</td>
</tr>
</tbody>
</table>

**Returned parameters**

None.

**Example request and response**

**XML request**

```bash
curl -k -u admin:xxxxx -X POST  https://localhost:8089/services/storage/fshpasswords/ -d provider=remote_splunk_deployment_00 -d name=remotefshuser -d password=foobar
```

**XML response**

```xml
...<title>fshpasswords</title>    <id>https://localhost:8089/services/storage/fshpasswords/id>    <updated>2019-02-26T08:50:41+00:00</updated>    <generator build="20f98da975b6" version="7.3.0"/>    <author>        <name>Splunk</name>    </author>    <link href="/services/storage/fshpasswords/_new" rel="create"/>    <link href="/services/storage/fshpasswords/_reload" rel="_reload"/>    <link href="/services/storage/fshpasswords/_acl" rel="_acl"/>    <opensearch:totalResults>1</opensearch:totalResults>    <opensearch:itemsPerPage>30</opensearch:itemsPerPage>    <opensearch:startIndex>0</opensearch:startIndex>    <s:messages/>    <entry>        <title>remote_splunk_deployment_00:remotefshuser:</title>        <id>https://localhost:8089/servicesNS/nobody/system/storage/fshpasswords/remote_splunk_deployment_00%3Aremotefshuser%3A</id>        <updated>1970-01-01T00:00:00+00:00</updated>        <link href="/servicesNS/nobody/system/storage/fshpasswords/remote_splunk_deployment_00%3Aremotefshuser%3A/_reload" rel="_reload"/>        <link href="/servicesNS/nobody/system/storage/fshpasswords/remote_splunk_deployment_00%3Aremotefshuser%3A/_new" rel="create"/>        <link href="/servicesNS/nobody/system/storage/fshpasswords/remote_splunk_deployment_00%3Aremotefshuser%3A/_acl" rel="_acl"/>        <author>            <name>nobody</name>        </author>    </entry>
```

85
DELETE

Removes existing user credentials for the federated providers.

Request parameters
None.

Returned parameters
None.

Example request and response

XML request
curl -k -u admin:changeme -X DELETE
   https://localhost:8089/services/storage/fshpasswords/remote_splunk_deployment_00:remotefshuser

XML response

<title>fshpasswords</title>
<id>https://localhost:8089/services/storage/fshpasswords</id>
<updated>2019-02-26T08:50:52+00:00</updated>
<generator build="20f98da975b6" version="7.3.0"/>
<author>
   <name>Splunk</name>
</author>
<link href="/services/storage/fshpasswords/_new" rel="create"/>
<link href="/services/storage/fshpasswords/_reload" rel="_reload"/>
<link href="/services/storage/fshpasswords/_acl" rel="_acl"/>
<opensearch:totalResults>0</opensearch:totalResults>
<opensearch:itemsPerPage>30</opensearch:itemsPerPage>
<opensearch:startIndex>0</opensearch:startIndex>
<s:messages/>
</feed>

federated

/services/dfs/federated

https://<host>:<mPort>/services/dfs/federated

Use this endpoint to create and maintain the DFS federated providers in the global namespace
/etc/system/local/federated.conf.

Authentication and authorization
This endpoint is restricted to roles that have the fsh_manage capability. Roles with the fsh_manage capability can perform
create, read, edit, or delete operations.

Usage details
For information about setting up service accounts for access control in a federated search deployment, see Set up service
accounts for access control.

GET

Returns information on the remote federated providers in a federated search deployment.

Request parameters
None.

Returned parameters
None.

Example request and response

XML request

curl -k -u admin:xxxxx -X GET https://localhost:8089/services/dfs/federated/remote_splunk_deployment_0

XML response
<entry>
<title>remote_splunk_deployment_0</title>
<id>https://localhost:8089/servicesNS/nobody/system/dfs/federated/remote_splunk_deployment_0</id>
<updated>1970-01-01T00:00:00+00:00</updated>
<author>
<name>nobody</name>
</author>
<content type="text/xml">
<s:dict>
<s:key name="disabled">0</s:key>
<s:key name="eai:acl">
<s:dict>
<s:key name="app">system</s:key>
<s:key name="can_change_perms">1</s:key>
<s:key name="can_list">1</s:key>
<s:key name="can_share_app">1</s:key>
<s:key name="can_share_global">1</s:key>
<s:key name="can_share_user">0</s:key>
<s:key name="can_write">1</s:key>
<s:key name="modifiable">1</s:key>
<s:key name="owner">nobody</s:key>
<s:key name="perms">
<s:dict>
<s:key name="read">
<s:list>
<s:item>*</s:item>
</s:list>
</s:key>
<s:key name="write">
<s:list>
<s:item>*</s:item>
</s:list>
</s:key>
<s:dict>
</s:dict>
<s:key name="removable">1</s:key>
<s:key name="sharing">system</s:key>
</s:dict>
</s:dict>
</s:messages/>
</entry>
POST

Sets up the remote federated providers in a federated search deployment.

Request parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>required</td>
<td>String Name of the DFS federated provider.</td>
</tr>
<tr>
<td>ip</td>
<td>required</td>
<td>String IP address of the DFS federated provider.</td>
</tr>
<tr>
<td>splunk.port</td>
<td>required</td>
<td>String Splunk management port on the Splunk Enterprise deployment that is configured to be a DFS federated provider.</td>
</tr>
<tr>
<td>splunk.serviceAccount</td>
<td>required</td>
<td>String Splunk user on the Splunk Enterprise deployment that is configured to be a DFS federated provider.</td>
</tr>
<tr>
<td>type</td>
<td>optional</td>
<td>String The type of the federated provider. A Splunk deployment is the only supported federated provider in this version. The default value is splunk.</td>
</tr>
<tr>
<td>splunk.app</td>
<td>optional</td>
<td>String The application context used for searches run with this configuration.</td>
</tr>
</tbody>
</table>

Returned parameters

None.
Example request and response

XML request

curl -k -u admin:changeme -X POST https://localhost:8089/services/dfs/federated -d name=remote_splunk_deployment_00 -d ip=10.224.187.234 -d splunk.port=8089 -d splunk.serviceAccount=remotefshuser

XML response

...
DELETE

Removes the remote federated providers from a federated search deployment.

**Request parameters**
None.

**Returned parameters**
None

**Example request and response**

**XML request**

```bash
curl -k -u admin:changeme -X DELETE
https://localhost:8089/services/dfs/federated/remote_splunk_deployment_00
```

**XML response**

```xml
<feed>
  <entry>
    <title>conf-federated</title>
    <id>https://localhost:8089/services/dfs/federated</id>
    <updated>2019-02-26T08:38:15+00:00</updated>
    <generator build="20f98da975b6" version="7.3.0"/>
    <author>
      <name>Splunk</name>
    </author>
    <link href="/services/dfs/federated/_new" rel="create"/>
    <link href="/services/dfs/federated/_reload" rel="_reload"/>
    <link href="/services/dfs/federated/_acl" rel="_acl"/>
    <opensearch:totalResults>0</opensearch:totalResults>
    <opensearch:itemsPerPage>30</opensearch:itemsPerPage>
    <opensearch:startIndex>0</opensearch:startIndex>
    <s:messages/>
  </entry>
</feed>
```
Appendix 2: DFS configuration files

Edit configuration files for big data analysis

Set up big data analysis by manually editing the stanzas in specific configuration files. For more information on DFS-specific configuration files, see DFS-specific configuration files.

Configuration files to set up big data analysis

The following table lists the configuration files that allow you to enable big data analysis:

<table>
<thead>
<tr>
<th>Configuration file</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server.conf</td>
<td>Allows you to enable a DFS search</td>
</tr>
<tr>
<td>limits.conf</td>
<td>Allows you to specify the number of concurrent searches, optimize performance based on the resources, and customize the search based on specific applications</td>
</tr>
</tbody>
</table>

Edit configuration files

Edit the server.conf and limits.conf configuration files to set up big data analysis. For more information on editing configuration files, see How to edit a configuration file in the Splunk Enterprise Admin Manual.

2. Enable big data analysis by changing the disabled field to false in the [dfs] stanza of the server.conf configuration file.
4. Using the following table, change the [dfs] stanza in the limits.conf configuration file to:
   - Specify the number of concurrent searches
   - Identify the resources to use
   - Optimize performance based on the resources
   - Customize the search based on specific applications

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>dfc_control_port</td>
<td>Communication port between the DFS-coordinator (DFC) and the search process</td>
<td>&lt;port&gt;</td>
<td>17000</td>
</tr>
<tr>
<td>dfc_num_slots</td>
<td>Maximum number of concurrent DFS searches that run on each search head The allocation of resources for each search is elastic and dynamic.</td>
<td>&lt;integer&gt;</td>
<td>4</td>
</tr>
<tr>
<td>dfs_max_num_keepalives</td>
<td>Maximum number of keepalive packets to run the search</td>
<td>&lt;integer&gt;</td>
<td>10</td>
</tr>
<tr>
<td>dfs_max_reduce_partition_size</td>
<td>Maximum number of partitions that receive data from the DFS workers</td>
<td>&lt;integer&gt;</td>
<td>500000</td>
</tr>
<tr>
<td>dfw_num_slots</td>
<td>Maximum number of concurrent DFS searches that run on a search head cluster</td>
<td>&lt;integer&gt;</td>
<td>10</td>
</tr>
<tr>
<td>dfw_num_slots_enabled</td>
<td>Enables you to set the value of the field dfw_num_slots.</td>
<td>&lt;boolean&gt;</td>
<td>false</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td>Value</td>
<td>Default value</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>dfw_receiving_data_port</td>
<td>Port that the DFS worker uses to receive pre-processed search results from the indexers. If that port is not available, the DFS worker searches for the next available port up to the value in the field <code>dfw_receiving_data_port_count</code>.</td>
<td>&lt;port&gt;</td>
<td>17500</td>
</tr>
<tr>
<td>dfw_receiving_data_port_count</td>
<td>Maximum number of ports that Splunk software checks for availability, starting from the default port set in the parameter <code>dfw_receiving_data_port</code>. Allows Splunk software checks for any available port without any upper limit.</td>
<td>&lt;integer&gt;</td>
<td>0</td>
</tr>
<tr>
<td>dfs_remote_search_timeout</td>
<td>Amount of time in seconds to wait because the search run on the DFS worker has not received the new results from any of the indexers.</td>
<td>&lt;integer&gt;</td>
<td>600</td>
</tr>
<tr>
<td>dfs_max_remote_pipeline</td>
<td>Controls the number of search pipelines launched at the indexer during a DFS search to stream data to the DFS workers. Optimal value is based on the number of vCPUs available on the indexers. A higher number of pipelines may improve search performance. However, modifying this setting from the default value may lead to slower searches or unread records and generates a UI warning.</td>
<td>&lt;integer&gt;</td>
<td>12</td>
</tr>
</tbody>
</table>

Resource allocation for DFS searches are dynamic and are allocated based on event estimation count, instead of the value provided in the `dfc_num_slot` setting in the `limits.conf` configuration file. Set the number of DFS searches by editing the `max_searches_per_process` field in the `[search]` stanza of the `limits.conf` configuration file. The default value for the maximum number of searches that you can run is 500. For more information, see `limits.conf` file in the *Admin Manual*.

**authorize.conf**

The following are the spec and example files for `authorize.conf`.

**authorize.conf.spec**

Version 8.0.0

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 are used for selecting or eliminating during role inheritance.
* If "true", the search filters are used for selecting. The filters are joined with an OR clause when combined.
* If "false", the search filters are used for eliminating. The filters are joined with an AND clause when combined.
* 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))
* Default: true

[capability::<capability>]

* DO NOT edit, remove, or add capability stanzas. The existing capabilities are the full set of Splunk system capabilities.
* Splunk software 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 capabilities for each role.
* NOTE: 'enabled' is the only accepted value here, as capabilities are disabled by default.
* Roles inherit all capabilities from imported roles, and you cannot disable inherited capabilities.
* Role names cannot have uppercase characters. Usernames, however, are case-insensitive.
* Role names cannot contain spaces, colons, semicolons, or forward slashes.

`importRoles = <semicolon-separated list>`
* A list of other roles and their associated capabilities that Splunk software should import.
* Importing other roles also imports the other aspects of that role, such as allowed indexes to search.
* Default: A role imports no other roles

`grantableRoles = <semicolon-separated list>`
* A list of roles that determines which users, roles, and capabilities that a user with a specific set of permissions can manage.
* This setting lets you limit the scope of user, role, and capability management that these users can perform.
* When you set 'grantableRoles', a user that holds a role with the 'edit_roles_grantable' and 'edit_user' capabilities can do only the following with regards to access control management for the Splunk Enterprise instance:
  * They can edit only the roles that contain capabilities that are a union of the capabilities in the roles that you specify with this setting.
  * Any new roles that they create can contain only the capabilities that are a union of these capabilities.
  * Any new roles that they create can search only the indexes that have been assigned to all roles that have been specified with this setting.
  * They can see only users who have been assigned roles that contain capabilities that are a union of these capabilities.
  * They can assign users only to roles whose assigned capabilities are a union of these capabilities.
* For this setting to work, you must assign a user at least one role that:
  * Has both the 'edit_roles_grantable' and 'edit_user' capabilities assigned to it, and
  * Does NOT have the 'edit_roles' capability assigned to it.
  * Example:
    * Consider a Splunk instance where role1-role4 have the following capabilities:
      role1: cap1, cap2, cap3
      role2: cap4, cap5, cap6
      role3: cap1, cap6
      role4: cap4, cap8
    * And user1-user4 have been assigned the following roles:
      user1: role1
      user2: role2
      user3: role3
      user4: role4
    * If you define the 'grantableRoles' setting as follows for the 'power' role:
      * [role_power]
      * grantableRoles = role1;role2
    * and edit the role so that the 'edit_roles_grantable' capability is selected, and the 'edit_roles' capability is not selected, then a user that has been assigned the 'power' role can make only the following access control changes on the instance:
* View or edit the following users: user1, user2, user3
* Assign the following roles: role1, role2, role3
* Create roles with the following capabilities: cap1, cap2, cap3, cap4, cap5, cap6
* Only the 'admin' role holds the 'edit_roles_grantable' capability on a new Splunk Enterprise installation.
* If you make changes to the 'admin' role, 'grantableRoles' is set to "admin".
* This setting does not work if you use tokens to authenticate into a Splunk Enterprise instance.
* Default (if 'admin' role is edited): admin
* Default (otherwise): No default

srchFilter = <semicolon-delimited list>
* A list of search filters for this role.
* To override any search filters from imported roles, set this to "*", as the 'admin' role does.
* Default: Splunk software does not perform search filtering

srchTimeWin = <integer>
* Maximum time span, in seconds, of a search.
* This time window limit is applied backwards from the latest time specified in a search.
* To override any search time windows from imported roles, set this to "0" (infinite), as the 'admin' role does.
* "-1" is a special value implying that no search window has been set for this role.
* This is equivalent to not setting the 'srchTimeWin' setting at all, which means it can be easily overridden by an imported role.
* Default: Searches are not limited to any specific time window

srchDiskQuota = <integer>
* The maximum amount of disk space, in megabytes, that can be used by search jobs for a specific user with 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. Additionally, the search process checks the quota at intervals that are defined in the 'disk_usage_update_period' setting in limits.conf as long as the search is active.
* A user can occasionally exceed the quota because the search process does not constantly check the quota.
* Exceeding this quota causes the search to be auto-finalized immediately, even if there are results that have not yet been returned.
* Default: 100

srchJobsQuota = <integer>
* The maximum number of concurrently running historical searches that a user with this role can have.
* This setting excludes real-time searches. See the 'rtSrchJobsQuota' setting.
* Default: 3

rtSrchJobsQuota = <integer>
* The maximum number of concurrently running real-time searches that a user with this role can have.
* Default: 6

srchMaxTime = <integer><unit>
* The maximum amount of time that search jobs from specific users with this role are allowed to run.
* After a search runs for this amount of time, it auto-finalizes.
* If the role inherits from other roles, the value of the 'srchMaxTime' setting is
specified in the included roles.
* This maximum value does not apply to real-time searches.
* Examples: 1h, 10m, 2hours, 2h, 2hrs, 100s
* Default: 100days

srchIndexesDefault = <semicolon-separated list>
* A 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 an underscore ("_"). 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.
* No default.

srchIndexesAllowed = <semicolon-separated list>
* A list of indexes that this role is allowed to search.
* Follows the same wildcarding semantics as the 'srchIndexesDefault' setting.
* 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.
* No default.

deleteIndexesAllowed = <semicolon-separated list>
* A list of indexes that 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 the 'srchIndexesDefault' setting.
* No default.

cumulativeSrchJobsQuota = <integer>
* The maximum total number of concurrently running historical searches
  across all members of this role.
* For this setting to take effect, you must set the 'enable_cumulative_quota'
  setting to "true" in limits.conf.
* 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.
* No default.

cumulativeRTSrchJobsQuota = <integer>
* The maximum total number of concurrently running real-time searches
  across all members of this role.
* For this setting to take effect, you must set the 'enable_cumulative_quota'
  setting to "true" in limits.conf.
* 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.
* No default.

federatedProviders = <semicolon-separated list>
* List of federated providers that the role can access.
* Allows a user to run federated searches defined in the savedsearches.conf file. This
  setting must be used in conjunction with fsh_search capability.
* Defaults to none.
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.

**[tokens_auth]**

* Settings for token authorization.

expiration = <relative-time-modifier>|never
* The relative time when an authorization token expires.
* The syntax for using time modifiers is:
  * [+]<time_integer><time_unit>@<time_unit>
  * Where time_integer is an integer value and time_unit is relative
  * time unit in seconds (s), minutes (m), hours (h) or days (d) etc.
* The steps to specify a relative time modifier are:
  * Indicate the time offset from the current time.
  * Define the time amount, which is a number and a unit.
  * Specify a "snap to" time unit. The time unit indicates the nearest
    or latest time to which your time amount rounds down.
* For example, if you configure this setting to "+2h@h", the token expires at
  the top of the hour, two hours from the current time.
* For more information on relative time identifiers, see "Time Modifiers" in
* The default value indicates that a token never expires. To set token
  expiration, you must set this value to a relative time value.
  * Your account must hold the admin role to update this setting.
  * This setting is optional.
  * Default: never

disabled = <boolean>
* Disables and enables Splunk token authorization.
* Default: true

**[capability::accelerate_datamodel]**

* Lets a user enable or disable data model 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.
* Lets users take advantage of the search performance gains
  related to parallel reduce functionality.
* Multi-phased 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 bypass any Access Control List (ACL) restrictions, similar to the way root access in a *nix environment does.
* Splunk software checks this capability when accessing manager pages and objects.

[**capability::edit_tokens_settings**]

* Lets a user access all token auth settings in the system, such as turning the feature on/off and system-wide expiration.
* 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' command.
* NOTE: The 'delete' command does not actually delete the raw data on disk. Instead, it masks the data (via the index) from showing up in search results.

[**capability::delete_messages**]

* Lets a user delete system messages that appear in the UI navigation bar.

[**capability::dispatch_rest_to_indexers**]

* Lets a user dispatch the REST search command to indexers.

[**capability::edit_authentication_extensions**]

* Lets a user change the authentication extensions through the authentication endpoints.

[**capability::edit_bookmarks_mc**]

* Lets a user add bookmark URLs within the Monitoring Console.
[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::list_dist_peer]

* Lets a user list/read peers for distributed search.

[capability::edit_dist_peer]

* Lets a user add and edit peers for distributed search.
* Supercedes list_dist_peer also allows list/read

[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_watchdog]

* Lets a user reconfigure watchdog settings using a REST endpoint.

[capability::edit_forwarders]

* Lets a user edit settings for forwarding data, including settings for SSL, backoff schemes, and so on.
* 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, and so on.
* 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_local_apps]

* Lets a user edit apps on the local Splunk instance through the local apps endpoint.
* For full access to app management, also add the 'install_apps' capability to the role.
* To enable enforcement of the "install_apps" capability, see the "enable_install_apps" setting in limits.conf.

[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 oneshot 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 Active Directory (AD).

**[capability::edit_roles]**

* Lets a user edit roles.
* Lets a user change the mappings from users to roles.
* Used by both user and role endpoints.

**[capability::edit_roles_grantable]**

* Lets a 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 the 'grantableRoles' setting in authorize.conf.
  * For example:
    ```
    grantableRoles = role1;role2;role3
    ```
  * This configuration lets a user create roles using the subset of capabilities that the user has in their 'grantable_roles' setting.

**[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_concurrency_all]**

* Lets a user edit settings related to maximum concurrency of searches.

**[capability::edit_search_concurrency_scheduled]**

* Lets a user edit settings related to concurrency of scheduled searches.

**[capability::edit_search_scheduler]**

* Lets a 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 a user edit general server and introspection settings, such as the server name, log levels, and so on.
* This capability also inherits the ability to read general server and introspection settings.

[capability::edit_server_crl]

* Lets a user reload Certificate Revocation Lists (CRLs) within Splunk.
* A CRL is a list of digital certificates that have been revoked by the issuing certificate authority (CA) before their scheduled expiration date and should no longer be trusted.

[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 splunktcp tokens. 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 for opting in and sending 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, which is a way to send data to Splunk Enterprise and Splunk Cloud.

[capability::edit_tokens_all]

* Lets a user issue tokens to all users.

[capability::edit_tokens_own]

* Lets a user issue tokens to themself.

[capability::edit_udp]

* Lets a user change settings for UDP inputs.

[capability::edit_user]

* Lets a user create, edit, or remove other users.
* Also lets a user manage certificates for distributed search.
* To limit this ability, assign the 'edit_roles_grantable' capability and configure the 'grantableRoles' setting in authorize.conf.
  * Example: grantableRoles = role1;role2;role3

[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 generate a diag on a remote 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 the inputcsv command (except for 
  dispatch=t mode) and the inputlookup command.

[capability::install_apps]

* Lets a user install, uninstall, create, and update apps on the local 
  Splunk platform instance through the apps/local endpoint. 
* For full access to app management, also add the 'edit_local_apps' 
  capability to the role.
* To enable enforcement of the "install_apps" capability, see the 
  "enable_install_apps" setting in limits.conf.

[capability::license_tab]

* DEPRECATED.
* Lets a user access and change the license.
* Replaced with the 'license_edit' capability.

[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_accelerate_search]

* This capability is a subset of the 'accelerate_search' capability.
  * This capability grants access to the summaries that are required to run accelerated reports.
  * Users with this capability, but without the 'accelerate_search' capability, can run, but not create, accelerated reports.

[capability::list_deployment_client]

* Lets a user list the deployment clients.

[capability::list_deployment_server]

* Lets a user list the deployment servers.

[capability::list_pipeline_sets]

* Lets a user list information about pipeline sets.

[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, and so on) 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, and so on.

[capability::list_indexerdiscovery]

* 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, and so on.

[capability::list_introspection]

* Lets a user read introspection settings and statistics for indexers, search, processors, queues, and so on.

[capability::list_search_head_clustering]

* Lets a user list search head clustering objects such as artifacts, delegated jobs, members, captain, and so on.

[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 and log levels.

[capability::list_metrics_catalog]

* Lets a user list metrics catalog information such as the metric names, dimensions, and dimension values.

[capability::edit_metrics_rollup]

* Lets a user create/edit metrics rollup defined on metric indexes.

[capability::list_storage_passwords]

* Lets a user access the /storage/passwords endpoint.
* Lets the user perform GET operations.
* The 'admin_all_objects' capability must be added to the role in order for the user to perform POST operations to the /storage/passwords endpoint.

[capability::list_tokens_all]

* Lets a user view all tokens.
[capability::list_tokens_own]

* Lets a user view their own tokens.

[capability::never_lockout]

* Allows a user's account to never lockout.

[capability::never_expire]

* Allows a user's account to never expire.

[capability::output_file]

* Lets a user create file outputs, including the 'outputcsv' command (except for dispatch-t mode) and the 'outputlookup' command.

[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.spec for more information.

[capability::rest_apps_view]

* Lets a user list various properties in the Python remote apps handler.
* See restmap.conf.spec 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 software through the server control handler.
[capability::rtsearch]

* Lets a user run real-time 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.
  * You must enable the 'scheduled_search' and 'rtsearch' capabilities for the role.

[capability::schedule_search]

* Lets a user schedule saved searches, create and update alerts, and review triggered alert information.

[capability::metric_alerts]

* Lets a user create and update the new metric alerts.

[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' command.
  * The 'file' command is DEPRECATED.
[capability::upload_lookup_files]

* Lets a user upload files which can be used in conjunction with lookup definitions.

[capability::web_debug]

* Lets a user access /_bump and /debug/** web debug endpoints.

[capability::fsh_manage]

* Lets a user in Splunk platform implementations that have enabled Data Fabric Search (DFS) functionality manage the federated search settings.
* With the federated search settings, users with this role can add federated providers to federated.conf and manage user access to those federated providers through the maintenance of authentication settings.
* The 'admin' role has this capability enabled by default.

[capability::fsh_search]

* Lets a user in Splunk platform implementations that have enabled Data Fabric Search (DFS) functionality run federated searches.
* Lets a user create federated searches in the savedsearches.conf.
* The 'admin' role has this capability enabled by default.

[capability::edit_statsd_transforms]

* Lets a user define regular expressions to extract manipulated dimensions out of metric_name fields in statsd metric data using the services/data/transforms/statsdextractions endpoint.
* For example, dimensions can be mashed inside a metric_name field like "dimension1.metric_name1.dimension2" and you can use regular expressions to extract it.

[capability::edit_metric_schema]

* Lets a user define the schema of the log data that must be converted to metric format using the 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 configuration information (except workload rule) through the workloads endpoint.
[capability::select_workload_pools]

* Lets a user select a workload pool for a scheduled or ad-hoc search.

[capability::list_workload_rules]

* Lets a user list and view workload rule information from the workloads/rules endpoint.

[capability::edit_workload_rules]

* Lets a user create and edit workload rules through the workloads/rules endpoint.

[capability::apps_restore]

* Lets a user restore configurations from a backup archive through the apps/restore endpoint.

authorize.conf.example

# Version 8.0.0
#
# 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
srchFilter = host=foo
srchIndexesAllowed = *
srchIndexesDefault = mail;main
srchJobsQuota = 8
rtSrchJobsQuota = 8
srchDiskQuota = 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.

federated.conf

The following are the spec and example files for federated.conf.

federated.conf.spec

Version 8.0.0

This file contains possible setting and value pairs for federated provider entries for use in Data Fabric Search (DFS), when the federated search functionality is enabled.

A federated search allows authorized users to run searches across multiple federated providers. Only Splunk deployments are supported as federated providers. Information on the Splunk deployment (i.e. the federated provider) is added in the federated provider stanza of the federated.conf file. A federated search deployment can have multiple federated search datasets. The settings for federated search dataset stanzas are located in savedsearches.conf.

To learn more about configuration files (including precedence) please see the documentation located at http://docs.splunk.com/Documentation/Splunk/latest/Admin/Aboutconfigurationfiles

Here are the settings for the federated provider stanzas.

[federated-provider-stanza]

* Create a unique stanza name for each federated provider.

type = [splunk]
* Specifies the type of the federated provider.
* Only Splunk deployments are supported as of this revision.
* Default: splunk

ip = <IP address>
* Identifies the IP address of the federated provider.
* Default: No default.

splunk.port = <port>
* Identifies the splunkd REST port on the remote Splunk deployment.
* Default: No default.

splunk.serviceAccount = <user>
* Identifies an authorized user on the remote Splunk deployment.
* The security credentials associated with this account are managed securely in fshpasswords.conf.
* Default: No default.

splunk.app = <string>
* The name of the Splunk application on the remote Splunk deployment in which to perform the search.
* Default: No default.
federated.conf.example

```bash
# Version 8.0.0
#
# This is an example federated.conf.
#
## Federated Providers

[deployment-sf-search]
type = splunk
ip = 192.0.2.0
splunk.port = 8089
splunk.serviceAccount = sf-search
splunk.app = search

[deployment-sf-hr]
type = splunk
ip = 192.0.2.0
splunk.port = 8089
splunk.serviceAccount = sf-hr
splunk.app = search

[deployment-sr-search]
type = splunk
ip = 198.51.100.0
splunk.port = 8089
splunk.serviceAccount = sf-search
splunk.app = search
```

limits.conf

The following are the spec and example files for limits.conf.

limits.conf.spec

```bash
Version 8.0.0

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 = <boolean>
* 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 processor 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 'maxresultrows' setting as the threshold. When the number of results exceeds the 'maxresultrows' setting, the eventstats command processor stops adding the requested fields to the search results.
* Default: 200

\[
\text{min\_batch\_size\_bytes} = \text{<integer>}
\]
* Specifies the size, in bytes, 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: 20971520

\[
\text{regex\_cpu\_profiling} = \text{<boolean>}
\]
* 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

\[
\text{file\_and\_directory\_eliminator\_reaper\_interval} = \text{<integer>}
\]
* Specifies how often, in seconds, to run the FileAndDirectoryEliminator reaping process.
* The FileAndDirectoryEliminator eliminates files and directories by moving them to a location that is reaped periodically. This reduces the chance of encountering issues due to files being in use.
* On Windows, the FileAndDirectoryEliminator is used by the deployment client to delete apps that have been removed or that are being redeployed.
* A value of 0 disables the FileAndDirectoryEliminator.
* NOTE: Do not change this setting unless instructed to do so by Splunk Support.
* Default (on Windows): 60
* Default (otherwise): 0

[searchresults]

* This stanza controls search results for a variety of Splunk search commands.

\[
\text{compression\_level} = \text{<integer>}
\]
* Compression level to use when writing search results to .csv.gz files.
* Default: 1

\[
\text{maxresultrows} = \text{<integer>}
\]
* 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)

subsearch_artifacts_delete_policy = [immediate|ttl]
* How subsearch artifacts are deleted after a sub search completes.
* Set to `immediate` to have subsearch artifacts remove immediately after a
  subsearch completes.
* Set to 'ttl' to have subsearch artifacts delete after the time-to-live of
  the subsearch has been reached.
* For example, you could use '|noop subsearch_artifacts_delete_policy = [immediate|ttl]'
  to overwrite the setting for a particular search.
* Default: ttl

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 = <boolean>
* 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 = <integer>
* 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 = <integer>
* 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.
batch_search_max_results_aggregator_queue_size = <integer>
* 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: 100000000

batch_search_max_serialized_results_queue_size = <integer>
* 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: 100000000

NOTE: The following batch search settings control the periodicity of retries
  to search peers in the event of failure (Connection errors, and others).
  The interval exists between failure and first retry, as well as
  successive retries in the event of further failures.

batch_retry_min_interval = <integer>
* 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 = <integer>
* 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 = <boolean>
* On a search peer, allow remote (search head) bundles to be loaded in splunkd.
* Default: false.

replication_file_ttl = <integer>
* The time to live (ttl), in seconds, of bundle replication tarballs,
  for example: *.bundle files.
* Default: 600 (10 minutes)

replication_period_sec = <integer>
* 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

bundle_status_expiry_time = <interval>
* The amount of time the search head waits before purging the status of a knowledge bundle push request to the indexer.
* The status is purged either when it is not queried for a period greater than this setting or when its associated bundle is deleted by the reaper.
* The interval can be specified as a string for minutes, seconds, hours, days. For example; 60s, 1m, 1h, 1d etc.
* Default: 1h

Concurrency
This section contains settings for search concurrency limits.

base_max_searches = <integer>
* 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 x max_hist_searches
* Default: 1

max_searches_per_cpu = <integer>
* 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 x number_of_cpus + base_max_searches
* NOTE: The maximum number of real-time searches is computed as:
  max_rt_searches = max_rt_search_multiplier x 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|disabled|disabled|savedSearches|disabled]
* When set to "enabled": All remote search logs are downloaded barring the oneshot search.
* When set to "disabled|savedSearches": 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 Boolean values: [true|false] are still supported, but are 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 = <integer>
* The maximum size of the chunk queue.
* default: 10000000

max_combiner_memevents = <integer>
* Maximum size of the in-memory buffer for the search results combiner.
  The <integer> is the number of events.
* Default: 50000

max_tolerable_skew = <positive integer>
* Absolute value of the largest time skew, in seconds, that is tolerated 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 = <integer>
* 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 = <integer>
* 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 mean 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 = <boolean>
* Specifies whether to keep a history of the searches that are run.
* Default: true

max_history_length = <integer>
* 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 = <boolean>
* 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 = <decimal>
* 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 = <boolean>
* 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 "false": A fatal exception occurs at search parsing time.
* Default: false

indexed_as_exact_metasearch = <boolean>
* Specifies if a metasearch can process <field>=<value> the same as <field>::<value>, if <field> is an indexed field.
* When set to "true": Allows a larger set of metasearches when the "allow_inexact_metasearch" setting is "false". However, some of the metasearches might be inconsistent with the results of doing a normal search.
* Default: false

Misc

This section contains miscellaneous search settings.

disk_usage_update_period = <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: 10

dispatch_dir_warning_size = <integer>
* 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: 5000

do_not_use_summaries = <boolean>
* Do not use this setting without working in tandem with Splunk support.
* This setting is a very narrow subset of "summary_mode=none".
* When set to "true": Disables some functionality that is necessary for report acceleration.
* In particular, when set to "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: false

enable_datamodel_meval = <boolean>
* Enable concatenation of successively occurring evals into a single
comma-separated eval during the generation of datamodel searches.
* default: true

enable_conditional_expansion = <boolean>
* Determines whether or not scoped conditional expansion of knowledge
  objects occurs during search string expansion. This only applies on
  the search head.
* NOTE: Do not change unless instructed to do so by Splunk Support.
* Default: true

force_saved_search_dispatch_as_user = <boolean>
* 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 = <integer>
* Specifies how often, in milliseconds, a keepalive is sent while a search
  is running.
* Default: 30000 (30 seconds)

search_keepalive_max = <integer>
* The maximum number of uninterrupted keepalives before the connection is closed.
* This counter is reset if the search returns results.
* Default: 100

search_retry = <boolean>
* Specifies whether the Splunk software retries parts of a search within a
  currently-running search process 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 = <integer>
* 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 = <boolean>`
* Specifies if the system should track the _indextime range of returned search results.
* Default: true

`use_bloomfilter = <boolean>`
* Controls whether to use bloom filters to rule out buckets.
* Default: true

`use_metadata_elimination = <boolean>`
* 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
* NOTE: Do not change this setting unless instructed to do so by Splunk Support.
* Default: srs

`results_compression_algorithm = [gzip|zstd|none]`
* The compression algorithm used for storing serialized results on disk.
* Options:
  * gzip: gzip
  * zstd: zstd
  * none: No compression
* NOTE: Do not change this setting unless instructed to do so by Splunk Support.
* Default: zstd

`record_search_telemetry = <boolean>`
* Controls whether to record search related metrics in search_telemetry.json in the dispatch dir. It also indexes this file to the _introspection index.
* NOTE: Do not change this setting unless instructed to do so by Splunk Support.
* Default: true

`use_dispatchtmp_dir = <boolean>`
* 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 = <boolean>`
* Whether or not search always looks for a field that does not have "INDEXED = true" set in fields.conf using both the indexed and non-indexed forms.
* If set to "true", when searching for `<field>=<value>`, the lexicon is searched for both "<field>::<value>" and "<value>".
* If set to "false", when searching for `<field>=<val>`, the lexicon is searched only for "<value>".
* Set to "true" if you have fields that are sometimes indexed and sometimes not indexed.
* For field names that are always indexed, it is much better for performance to set "INDEXED = true" in fields.conf for that field instead.
* Default: false
max_searchinfo_map_size = <integer>
* Maximum number of entries in each SearchResultsInfo data structure map that
  are used to track information about search behavior
* Default: 50000

track_matching_sourcetypes = <boolean>
* if true, keeps track of the number of events of each sourcetype that match a
  search, and store that information in info.csv
* Default: true

max_audit_sourcetypes = <integer>
* if track_matching_sourcetypes = true, the matching sourcetypes
  for a search will be written to the info=completed audit.log message
  upon completion of the search, up to max_audit_sourcetypes.
* If max_audit_sourcetypes is set to 0, sourcetype information
  will not be added to audit.log.
* If the number of matching sourcetypes exceeds the max_audit_sourcetypes
  setting, the sourcetypes with the greatest number of matching
  events will be included.
* Default: 100

use_search_evaluator_v2 = <boolean>
* If true, search evaluator v2 is used.
* NOTE: Do not change this setting unless instructed to do so by Splunk Support.
* Default: true

eexecute_postprocess_in_search = <boolean>
* If true, try to run postprocess searches ahead of time in the search process
  instead of the main splunkd process.
* Default: true

Parsing

This section contains settings related to parsing searches.

max_macro_depth = <integer>
* 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 = <integer>
* 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 = <boolean>
* 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 = <boolean>
DEPRECATED This setting has been deprecated.

phased_execution_mode = [multithreaded|auto|singlethreaded]
* 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.
* NOTE: Do not change this setting unless instructed to do so by Splunk Support.
* 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

This section contains settings for 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 = <boolean>
* 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.

Reading chunk controls
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** = <integer>
* 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** = <integer>
* 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** = <integer>
* 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** = <integer>
* 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: 50 (.05 seconds)

**bucket_localize_status_check_backoff_start_ms** = <integer>
* When explicitly set, and different from **bucket_localize_status_check_period_ms**, enables exponential backoff between consecutive status checks for bucket localization. Starting from the specified amount of time, in milliseconds, up to
bucket_localize_status_check_period_ms.
* This setting is only relevant when using remote storage.
* Setting this option is beneficial when bucket contents localize quickly (e.g., in less time than the minimal allowed value for bucket_localize_status_check_period_ms), or with high variability.
* The minimum and maximum values are 1 and bucket_localize_status_check_period_ms, respectively. If the specified value falls outside this range, it is effectively set to the nearest value within the range.
* NOTE: Do not change this setting unless instructed to do so by Splunk Support.
* Default: 0 (no backoff)

bucket_localize_max_lookahead = <integer>
* 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 = <integer>
* 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

read_final_results_from_timeliner = <boolean>
* When you run a search of event data where 'status_buckets > 0', this setting controls the contents of the results.csv.gz and results.srs.zstd files in the search artifact.
* When set to "true", the final results saved to disk by the search process on the search head are a sample of events ready from the timeliner.
* When set to "false", the final results saved to disk by the search process on the search head are all events produced by the last SPL command, up to a
The 'read_final_results_from_timeliner' setting affects the output of subsequent 'loadjob' searches.  
* When set to "true" the 'loadjob' search returns the sample of the final results, not the full result set. For example, if the full result set is 10k results, it might return only 1000 results.  
* When set to "false" the 'loadjob' search returns the full set of search results. For example, if the full result set is 10k results, it returns 10k results.  
* Default: true

truncate_report = [1|0]  
* Specifies whether or not to apply the "max_count" setting to report output.  
* Default: 0 (false)

write_multifile_results_out = <boolean>  
* 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 = <integer>  
* 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 = <integer>  
* 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 = <integer>
* 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 = <integer>
* 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 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 = <integer>
* 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 "max_searches_per_process" is set to 0 or 1.
* NOTE: A search can run longer than the time set for "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)

process_min_age_before_user_change = <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
  "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 "search_process_mode" is not "auto" or
  if "max_searches_per_process" is set to 0 or 1.
* Default: 4

search_process_mode = [auto|traditional|debug <debugging-command> <debugging-args>]
* Controls how search processes are started.
* When set to "traditional": Each search process is initialized completely
  from scratch.
* When set to "debug": When set to a string beginning with "debug",
  searches are routed through the <debugging-command>, where the user can
  "plug in" debugging tools.
* The <debugging-command> must reside in one of the following locations:
  * $SPLUNK_HOME/etc/system/bin/
  * $SPLUNK_HOME/etc/apps/$YOUR_APP/bin/
  * $SPLUNK_HOME/bin/scripts/
* The <debugging-args> are passed, followed by the search command it
  would normally run, to <debugging-command>.
* For example, given the following setting:
  search_process_mode = debug $SPLUNK_HOME/bin/scripts/search-debugger.sh 5
  A command similar to the following is run:
  $SPLUNK_HOME/bin/scripts/search-debugger.sh 5 splunkd search \
  --id=... --maxbuckets=... --ttl=... [...]
* Default: auto

search_messages.log

132
log_search_messages = <boolean>
* Specifies whether splunkd promotes user-facing search messages
  from $SPLUNK_HOME/var/run/splunk/dispatch/<sid>/info.csv to
  $SPLUNK_HOME/var/log/splunk/search_messages.log.
* Splunkd does not promote messages with a severity that is ranked
  lower than the value of search_messages_severity.
* Splunkd promotes messages only after search has been audited.
* The search_messages.log file follows this format when it logs messages:
  orig_component="..." sid="..." peer_name="..." message="..."
* Default: false

search_messages_severity = <string>
* When 'log_search_messages = true', this setting specifies the lowest
  severity of message that splunkd logs to search_messages.log.
  The processor ignores all messages with a lower severity.
* Possible values in ascending order: DEBUG, INFO, WARN, ERROR
  * For example, when 'search_messages_severity = WARN', splunkd logs
    only messages with 'WARN' and 'ERROR' severities.
* Default: WARN

Search reuse

This section contains settings for search reuse.

allow_reuse = <boolean>
* 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 = <integer>
* 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

133
unified_search = <boolean>
* Specifies if unified search is turned on for hunk archiving.
* Default: false

Status
This section contains settings for search status.

status_cache_size = <integer>
* 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 = <integer>
* 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 = <integer>
* Size of the pool, in threads, responsible for writing out the full
  remote events.
* Default: 5

remote_event_download_initialize_pool = <integer>
* Size of the pool, in threads, responsible for initiating the remote
  event fetch.
* Default: 5

remote_event_download_local_pool = <integer>
* 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 = <integer>
* 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 = <integer>
* Maximum number of events to be stored per timeline bucket on each search peer.
  * Default: 10000

remote_timeline_max_size_mb = <integer>
* 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 = <integer>
* Minimum number of search peers for enabling remote computation of timelines.
  * Default: 1

remote_timeline_parallel_fetch = <boolean>
* Specifies whether to connect to multiple peers at the same time when fetching remote events.
  * Default: true

remote_timeline_prefetch = <integer>
* Specifies the maximum number of full eventuate that each peer should proactively send at the beginning.
  * Default: 100

remote_timeline_receive_timeout = <integer>
* Receive timeout, in seconds, for fetching events processed by remote peer timeliner.
  * Default: 10

remote_timeline_send_timeout = <integer>
* 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 = <boolean>
* 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 Frequency**

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

**TTL**

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 (24 hours)

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

This section contains settings that are no longer supported.

`enable_status_cache = <boolean>`
* 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 cacheing 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 cacheing is disabled.
* Explicit requests to jobs endpoints, eg /services/search/jobs/<sid> are always satisfied from disk, regardless of this setting.
* Default (when search head pooling is not enabled): true
* Default (when search head pooling is enabled): 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.
* Default: 60000 (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 = <integer>
* REMOVED. This setting is no longer used.

rr_max_sleep_ms = <integer>
* REMOVED. This setting is no longer used.

rr_sleep_factor = <integer>
* 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: The value set for 'maxresultrows' in the [searchresults] stanza, which is 50000 by default.

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 = <integer>
```
* 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: 1000000

**[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, 'maxbins' uses the value set for 'maxresultrows' in the [searchresults] stanza, which is 50000 by default.
* 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 = <decimal>
* 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 = <decimal>
* 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 = <boolean>
* 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 retrieved, the lookup falls 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>
* DEPRECATED: Use this setting in transforms.conf for lookup definitions.

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
* CAUTION: Setting this to a large value results in loading large lookup files in memory. This leads to a bigger process memory footprint.
* Default: 26214400 (25MB)

indexed_csv_ttl = <positive integer>
* Specifies the amount of time, in seconds, that a indexed CSV lookup table can exist without update before it is removed by Splunk software.
* On a period set by 'indexed_csv_keep_alive_timeout', Splunk software checks the CSV lookup table to see if it has been updated. If it has been updated, Splunk software modifies a special token file.
* At the end of the 'indexed_csv_ttl' period Splunk software looks at the token
file. If the token file shows that its CSV lookup table has been updated, Splunk software does not delete that CSV lookup table.
* Default: 300

indexed_csv_keep_alive_timeout = <positive integer>
* Sets the period, in seconds, for an activity check that Splunk software performs on indexed CSV lookup tables.
* When Splunk software performs a CSV lookup table check and finds that the table has been updated, it marks this activity on a token file. The token file update prevents the CSV lookup table from being deleted after 'indexed_csv_ttl' seconds of inactivity have passed.
* Default: 30

indexed_csv_inprogress_max_timeout = <positive integer>
* Sets the maximum time, in seconds, for Splunk software to wait for ongoing indexing of a CSV lookup table to finish before failing any search that is awaiting the lookup table.
* Default: 300

max_reverse_matches = <integer>
* maximum reverse lookup matches (for search expansion)
* Default: 50

shared_provider_cache_size = <integer>
* Sets the cache size in bytes that the Splunk software uses when it shares CSV lookups across multiple lookup commands.
* The <integer> represents the size of the cache in bytes. This is incremented by the size of each in-memory file (in bytes) inserted into the shared cache.
* Set this to 0 to disable lookup sharing, defaults to 200MB (209715200 bytes).
* Do not change this value unless you are advised to do so by Splunk Support or a similar authority.
* Default: 209715200

[metadata]

bucket_localize_max_lookahead = <integer>
* This setting is only relevant when using remote storage.
* 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 this setting unless instructed to do so by Splunk Support.
* Default: 100000
### [metric_alerts]

* This stanza provides global settings for metric alerts.

**condition_evaluation_interval = <integer>**
* This setting provides the alert condition evaluation interval in minutes.
* Must be a number from 1 to 60.
* Default: 1

**search_delay = <time specifier>**
* This setting provides a random delay time in seconds for metric alert searches. It can be passed to the 'allow_skew' setting for the search.
* The search delay allows the search to wait for the latest indexed data.
* Only change this setting if you are experiencing significant data latency issues.
* Default: 15s+

**search_ttl = <positive integer>**
* Specifies the default life span of metric alert search jobs.
* The time to live is defined as "at least until the Nth periodic run of the search, where the period is defined by the 'condition_evaluation_interval' setting".
* Default: 2p

**honor_action = <boolean>**
* Specifies whether the Splunk software should change the 'search_ttl' to the action ttl when an action is triggered.
* If there are multiple actions, the largest action ttl wins.
* Default: false

### [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 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 = <boolean>
* 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 lookup file.
* Default: false

**[rare]**

maxresultrows = <integer>
* Maximum number of result rows to create.
* If not specified, defaults to the value set for 'maxresultrows' in the [searchresults] stanza, which is 50000 by default.
* 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 to respect automatic field extraction when spath is invoked manually.
* If set to "true", all fields are extracted regardless of settings.
* If set to "false", only fields used by later search commands are extracted.
* Default: true
extraction_cutoff = <integer>
* For 'extract-all' spath extraction mode, this setting applies extraction only
to the first <integer> number of bytes. This setting applies both the auto kv
extraction and the spath command, when explicitly extracting fields.
* 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>
* The size of the digest used for approximating distinct count.
* The digest is configured to be $2^{\text{dc_digest_bits}}$ bytes in size.
* Must be $> 8$ (128B) and $\leq 16$ (64KB)
* Default: 10 (equivalent to 1KB)
default_partitions = <integer>
* Number of partitions to split incoming data into for parallel/multithreaded
reduce.
* Default: 1
list_maxsize = <integer>
* Maximum number of list items to emit when using the list() function
stats/sistats
* Default: 100
maxmem_check_freq = <integer>
* How frequently, in number of rows, to check if the in-memory data
structure size limit is exceeded, as specified by the
'max_mem_usage_mb' setting.
* 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 sent to the disk.
* If not specified, uses the value set for 'maxresultrows' in the
[searchresults] stanza, which is 50000 by default.
* 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 sistats 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 = <boolean>
* Whether or not to perform a natural sort on the output of 'stats'
  if the output size is greater than or equal to the 'maxresultrows' setting.
* A natural sort means that numbers are sorted numerically and non-numbers
  are sorted lexicographically.
* Default: true

partitions_limit = <integer>
* Maximum number of partitions to split into that can be specified with 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 \( R1 = \text{floor}(F) \) and \( R2 = \text{ceiling}(F) \).
    Answer = \( (R2 \times (F - R1)) + (R1 \times (1 - (F - R1))) \)
* 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 = <integer>
* 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.
* Example:
  * If you have the following configurations:
    * <sparkline_time_steps> = 1s,5s,10s,30s,1m,5m,10m,30m,1h,1d,1month
    * <sparkline_maxsize> = 100
  * The timespan for 7 days of data is 604,800 seconds.
  * Span = 604,800/<sparkline_maxsize>.
  * If <sparkline_maxsize> = 100, then
    span = (604,800 / 100) = 6,048 sec = 1.68 hours.
  * The "1d" time step is used because it is the lowest value that does not
    exceed the maximum number of bins.
* Default: 1s,5s,10s,30s,1m,5m,10m,30m,1h,1d,1month

146
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

rdigest_maxnodes = <integer>
* Maximum rdigest nodes before automatic compression is triggered.
* When set to "1": Specifies to automatically configure based on k value.
* Default: 1

tdigest_k = <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

tdigest_max_buffer_size = <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

tmpfile_compression = <string>
* temporary file compression format, used for stats tmp files only
* "lz4" indicates use of the lz4 format
* "zstd" indicates use of the zstd format
* "none" indicates use of no compression

tmpfile_compression_level = <int>
* Temporary file compression format level.
* If tmpfile_compression is lz4 or zstd, this will indicate the compression level.
* For zstd higher numbers indicate higher speed, and lower compression ratios.
* For lz4 higher numbers indicate lower speed, and higher compression ratios.

maxresultrows = <integer>
* Maximum number of result rows to create.
* If not specified, uses the value set for 'maxresultrows' in the [searchresults] stanza, which is 50000 by default.
* 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

**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

**tcollect**

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 integer>
* 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>
* Whether or not the 'tstats' command, when run on an accelerated datamodel, confirms that the datamodel search in each bucket's summary metadata is considered to be up to date with the current datamodel search.
* Only bucket summaries that are considered "up to date" are used to deliver results.
* This value is the default value of the 'allow_old_summaries' setting, if that argument is not specified in the command.
* When set to "false", 'tstats' always confirms that the datamodel search in each bucket's summary metadata is considered up to date with the current datamodel search.
* When set to "true", 'tstats' delivers results even from bucket summaries that are considered out of date with the current datamodel.
* Default: false

apply_search_filter = <boolean>
* Whether or not 'tstats' applies role-based search filters when users run the command on normal index data.
* If set to "true", 'tstats' applies role-based search filters.
* NOTE: Regardless of this setting value, 'tstats' never applies search filters to data collected with 'tcollect', or with datamodel acceleration.
* Default: true

bucket_localize_max_lookahead = <integer>
* 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 integer>
* 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>
* Whether or not 'tstats' employs a mixed mode when running against an
  accelerated datamodel.
* This value is the default value for the 'summariesonly' setting, if that
  argument is not specified in the command.
* In mixed mode, 'tstats' falls back to search if it encounters missing
  tsidx data.
* If set to "true", 'tstats' overrides this mixed mode, and only generates
  results from available tsidx data, which might be incomplete.
* If set to "false", 'tstats' uses mixed mode, and falls back to search for
  tsidx data that is missing.
* Default: false

warn_on_missing_summaries = <boolean>
* ADVANCED: Only meant for debugging 'summariesonly=true' searches on
  accelerated datamodels.
* When set to "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

batch_search_max_pipeline = <integer>
* Controls the number of tstats/mstats search pipelines launched at the
  indexer during batch search.
* Increase the number of search pipelines to improve search performance, at
  the cost of a concurrent increase in thread and memory usage.
* This value applies only to searches that run on remote indexers.
* Default: 1

[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 = <integer>
* In eventtyping, pay attention to first <integer> 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 = <boolean>
* 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" seconds.
  * Default: 600 (10 minutes)

detailed_dashboard = <boolean>
* 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 = <integer>
* Maximum number of summarization run statistics to keep track and expose via
  REST.
  * Default: 48

max_verify_buckets = <integer>
* When verifying buckets, stop after verifying this many buckets if no failures
  have been found
  * 0 means never
  * Default: 100

max_verify_bucket_time = <integer>
* 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 = <integer>
* 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 = <boolean>
* 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

shc_accurate_access_counts = <boolean>
* 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 = <boolean>
* Should summaries that fail verification be automatically deleted? * Default: false

[export]

add_offset = <boolean>
* Add an offset/row number to JSON streaming output * Default: true

add_timestamp = <boolean>
* 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

[auth]

* Settings for managing auth features.

enable_install_apps = <boolean>
* Whether or not the "install_apps" capability is enabled for app installation, uninstallation, creation, and update. * If set to "true", you must be assigned a role that holds the 'install_apps' capability to access the 'apps/local' REST endpoint for app installation, uninstallation, creation, and update. * If set to "false", you must be assigned a role that holds either the 'admin_all_objects' or 'edit_local_apps' capabilities for app installation, uninstallation, creation, and update. * Default: false
**[http_input]**

max\_number\_of\_tokens = <unsigned integer>
* 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\_per\_chunk = <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>
* The size, in megabytes, at which point the file tracking
  database, otherwise known as the "fishbucket" or "btree", rolls over
  to a new file.
* The rollover process is as follows:
  * After the fishbucket reaches 'file\_tracking\_db\_threshold\_mb' megabytes
    in size, a new database file is created.
  * From this point forward, the processor writes new entries to the
    new database.
  * Initially, the processor attempts to read entries from the new database,
    but upon failure, falls back to the old database.
  * Successful reads from the old database are written to the new database.
* NOTE: During migration, 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>
* The maximum amount of memory, in megabytes, used by the MonitorNoHandle modular input 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 instance 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 drops some data to stay within the limit.
* This setting is valid only on Windows machines.
* Default: 0

tailing_proc_speed = <integer>
* REMOVED. This setting is no longer used.

monitornohandle_max_driver_mem_mb = <integer>
* The maximum amount of NonPaged memory, in megabytes, used by the kernel driver of the MonitorNoHandle modular input.
* 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 instance can accept it.
* When set to 0, the NonPaged memory size (memory allocated in the kernel driver of the modular input) can grow without limit.
* If this size is limited, and the limit is encountered, the input drops some data to stay within the limit.
* This setting is valid only on Windows machines.
* Default: 0
monitor_nohandle_max_driver_records = <integer>
* The maximum number of in-memory records that the kernel module for
  the MonitorNoHandle modular input stores.
* This setting controls memory growth by limiting the amount of memory
  that the MonitorNoHandle input kernel module uses.
* When 'monitor_nohandle_max_driver_mem_mb' is set to > 0, this
  setting is ignored.
* The 'monitor_nohandle_max_driver_mem_mb' and
  'monitor_nohandle_max_driver_records' settings are mutually exclusive.
* If the limit is encountered, the input drops some data
  to remain within the limit.
* Default: 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 overrides
  the setting for all inputs, for now.

[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.
* 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 data with a large
  number of columns and want to ensure that searches display all fields extracted
  from an automatic key-value field (auto kv) configuration.
* Set this value to 0 if you do not want to limit the number of fields
  that can be extracted at index time and search time.
* Default: 100

indexed_kv_limit = <integer>
* The maximum number of fields that can be extracted at index time from a data source.
* Fields that can be extracted at index time include default fields, custom fields,
  and structured data header fields.
* The summary fields 'host', 'index', 'source', 'sourcetype', 'eventtype', 'linecount',
  'splunk_server', and 'splunk_server_group' do not count against this limit and are
  always 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.
* Set this value to 0 if you do not want to limit the number of fields that can be extracted at index time.
* Default: 200

maxchars = <integer>*
* Truncate _raw to this size and then do auto KV.
* Default: 10240 characters

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 integer>*
* 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 integer>*
* The maximum number of documents that can be saved in a single batch.
* Default: 1000

max_fields_per_acceleration = <unsigned integer>*
* 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 integer>*
* The maximum number of queries that can be run in a single batch.
* Default: 1000

max_rows_in_memory_per_dump = <unsigned integer>*
* The maximum number of rows in memory before flushing it to the CSV projection of KVStore collection.
* Default: 200

max_rows_per_query = <unsigned integer>*
* 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 integer>*
* 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 integer>*
* The maximum size, in megabytes (MB), of a batch save query.
* Default: 50
max_size_per_result_mb = <unsigned integer>
* 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 integer>
* 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 = <boolean>
* Whether or not Splunk Enterprise allows login when it finds multiple
  entries in LDAP with the same value for the 'username' attribute.
* When multiple entries are found, it chooses the first Distinguished Name
  (DN) lexicographically.
* Setting this to false is more secure as it does not allow any ambiguous
  login, but users with duplicate entries will be unable to login.
* Default: true

max_users_to_precache = <unsigned integer>
* The maximum number of users that are pre-cached 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_thruput reports in metrics.log.
* Default: 10

[metrics:tcpin_connections]

aggregate_metrics = <boolean>
* 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 = <boolean>`
  * 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 integer>`
  * The maximum number of rows that will be rendered for a table within integrated PDF rendering.
  * Default: 1000

  `render_endpoint_timeout = <unsigned integer>`
  * 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 = <integer>`
  * The time, in milliseconds, to wait between triggering alerts during a realtime search.
  * This setting limits the frequency at which alerts are triggered during realtime search.
  * A value of 0 means that alerts are triggered for every batch of events that are read. In dense realtime searches with expensive alerts, this can overwhelm the alerting system.
  * Precedence: Searchhead
  * Default: 0

  `blocking = <boolean>`
  * Whether or not the indexer should block if a queue is full.
  * Default: false

  `default_backfill = <boolean>`
  * Whether or not windowed real-time searches should backfill events.
  * Default: true

  `enforce_time_order = <boolean>`
  * Whether or not real-time searches should ensure that events are sorted in ascending time order.
  * Splunk Web automatically reverses the order that it displays events for real-time searches. If set to "true", the latest events will be shown first.
  * Default: true

  `indexfilter = <boolean>`
  * Whether or not the indexer should pre-filter events for efficiency.
  * Default: 1 (true)
indexed_realtime_update_interval = <integer>
* 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 = <integer>
* This setting is deprecated. Use the "indexed_realtime_update_interval" setting instead.
* While running an indexed realtime search on a cluster, the list of allowed primary buckets is updated. This controls the interval at which the list is updated. This value must be less than the 'indexed_realtime_disk_sync_delay' setting. If your buckets transition from Brand New to warm in less than the interval time specified, indexed realtime will lose data in a clustered environment.
* Precedence: Indexers
* Default: 30

indexed_realtime_default_span = <integer>
* 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 = <integer>
* The number of seconds to wait for disk flushes to finish when using indexed/continuous/pseudo realtime search, so that all data can be seen.
* 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 = <integer>
* 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 search drops data 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 = <boolean>
* Whether or not the indexedRealtime mode should be used by default.
* Precedence: SearchHead
* Default: false

local_connect_timeout = <integer>
* Connection timeout, in seconds, for an indexer's search process when connecting to that indexer's splunkd.
  * Default: 5

`local_receive_timeout` = <integer>
* Receive timeout, in seconds, for an indexer's search process when connecting to that indexer's splunkd.
  * Default: 5

`local_send_timeout` = <integer>
* Send timeout, in seconds, for an indexer's search process when connecting to that indexer's splunkd.
  * Default: 5

`max_blocking_secs` = <integer>
* Maximum time, in seconds, to block if the queue is full (meaningless if blocking = false)
  * 0 means no limit
  * Default: 60

`queue_size` = <integer>
* 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
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

can run.

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>.
* 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 'auto_summary_perc'.

concurrency_message_throttle_time = <integer>[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 [<integer>]<unit> to specify a duration;
  a missing <integer> 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_scheduled_search_lookback = <duration-specifier>
* The maximum amount of time to run missed continuous scheduled searches for once Splunk Enterprise comes back up, in the event it was down.
* Use [<integer>]<unit> to specify a duration;
  a missing <integer> 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 = <integer>
* The number of most recent lock files to keep around.
* This setting only applies in search head pooling.

max_lock_file_ttl = <integer>
* Time, in seconds, that must pass before reaping a stale lock file.
* Only applies in search head pooling.

max_per_result_alerts = <integer>
* 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 amount 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] max_searches_per_cpu for
  how to set the system wide maximum number of searches.
* Default: 50

max_searches_perc.<n> = <integer>
* 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 persistence 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

persistance_period = <integer>
* DEPRECATED: Use the 'persistence_period' setting instead.

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 integer>
* 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 = <boolean>
* Whether saved search jobs are disabled by the scheduler.
* Default: false
scheduled_view_timeout = <integer>[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 = <boolean>
* 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 \( n \times \text{number_of_members} \) 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 = <boolean>
* 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 \( n \times \text{number_of_members} \) 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
  \( \text{max_searches_per_cpu} \times \text{number_of_members} \)
  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

shc_local_quota_check = <boolean>
* DEPRECATED. Local (per-member) quota check is enforced by default.
* To disable per-member quota checking, enable one of the cluster-wide quota
  checks (shc_role_quota_enforcement or shc_syswide_quota_enforcement).
  For example, setting '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.

shp_dispatch_to_slave = <boolean>
* By default the scheduler should distribute jobs throughout the pool.
* Default: true

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 [integer]<unit> to specify a duration; a missing integer 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

search_history_max_runtimes = <unsigned integer>
* The number of runtimes kept for each search.
* Used to calculate historical typical runtime during search prioritization.
* Default: 10

[search_metrics]
debug_metrics = <boolean>
* This indicates whether to output more detailed search metrics for debugging.
* This will do things like break out where the time was spent by peer, and might 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 = <boolean>
* Whether or not a distributed search is performed 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 integer>
* The maximum number of events that are requested when performing a 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 software rarely returns anywhere near this number of results, as excess results are pruned.
* 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)
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

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 = <decimal>
* The fraction of peers that must have completed before disconnection begins.
* 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 = <decimal>
* The maximum value of the threshold data rate that is used to determine
  if a peer is slow.
* The actual threshold is computed dynamically at search time but never exceeds
  (100*maximum_threshold_as_fraction_of_mean)% on either side of the mean.
* Default: 0.2

disabled = <boolean>
* Whether or not this feature is enabled.
* Default: true

grace_period_before_disconnect = <decimal>
* How long, in seconds, when multiplied by life_time_of_collector, to wait
  while the heuristic claims that a peer is slow, before disconnecting the
  peer.
* If the heuristic consistently claims that the peer is slow for at least
  <grace_period_before_disconnect>*life_time_of_collector seconds, then the
  peer is disconnected.
* Default: 0.1

packets_per_data_point = <unsigned integer>
* Rate statistics will be sampled once every packets_per_data_point packets.
sensitivity = <decimal>
* 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 integer>
* All peers will be given an initial grace period of at least these many
  seconds before they are considered in the heuristic.
  * Default: 60

threshold_data_volume = <unsigned integer>
* The volume of uncompressed data that must have accumulated, in
  kilobytes (KB), from a peer before it is considered in the heuristic.
  * Default: 1024

[summarize]

bucket_refresh_interval = <integer>
* 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 = <integer>
* 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 integer>
* 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'
    seconds is NOT summarized.
  * NOTE: Do not change this setting unless instructed to do so by Splunk Support.
  * Default: 90

max_hot_bucket_summarization_idle_time = <unsigned integer>
* 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 integer>
* The maximum amount of time, in seconds, that a replicated hot bucket
can remain idle before 'indextime_lag' is no longer applied to it.
* This applies only to idle replicated hot buckets. When new events arrive,
  the default behavior of applying 'indextime_lag' resumes.
  * Default: 150

max_summary_ratio = <decimal>
* 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.
  * Set to 0 to disable.
  * Default: 0

\[\text{max_summary_size} = \text{<integer>}\]

* Size of summary, in bytes, at which point we’ll start applying the \[\text{max_summary_ratio}\].
  * Set to 0 to disable.
  * Default: 0

\[\text{max_time} = \text{<integer>}\]

* The maximum amount of time, seconds, that a summary search process is allowed to run.
  * Set to 0 to disable.
  * Default: 0

\[\text{poll_buckets_until_maxtime} = \text{<boolean>}\]

* 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

\[\text{sleep_seconds} = \text{<integer>}\]

* The amount of time, in seconds, to sleep between polling the summarization complete status.
  * Default: 5

\[\text{stale_lock_seconds} = \text{<integer>}\]

* 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]**

\[\text{insufficient_search_capabilities} = \text{enabled | disabled}\]

* 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

\[\text{installed_files_integrity} = \text{enabled | log_only | disabled}\]

* 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 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 the splunkd.log file.
* 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 thruput 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 = <boolean>
* Enables search optimizations
* Default: true

[search_optimization::search_expansion]

enabled = <boolean>
* 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 = <boolean>
* Enables replace append with union command optimization
* Default: true

[search_optimization::merge_union]

enabled = <boolean>
* Merge consecutive unions
* Default: true

[search_optimization::predicate_merge]

enabled = <boolean>
* Enables predicate merge optimization
* Default: true

inputlookup_merge = <boolean>
* 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 = <boolean>
* 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
* No default.

[search_optimization::predicate_push]

enabled = <boolean>
* Enables predicate push optimization
* Default: true

[search_optimization::predicate_split]

enabled = <boolean>
* Enables predicate split optimization
* Default: true

[search_optimization::dfs_job_extractor]

enabled = <boolean>
* Enables Splunk software to identify portions of searches and send them to
  the DFS cluster for fast processing.
* Can only be used by Splunk platform implementations that have enabled Data
Fabric Search (DFS) functionality.
* Default: true

commands = <Command List>
* A comma-separated list of search commands that are affected by DFS job extraction.

[search_optimization::projection_elimination]

enabled = <boolean>
* 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.
* No default.

[search_optimization::required_field_values]

enabled = <boolean>
* 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 = <boolean>
* 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 = <boolean>
* Enables reversing of calculated fields optimization.
* Default: true
[search_optimization::search_sort_normalization]

enabled = <boolean>
* 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 = <boolean>
* 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 = <boolean>
* 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

[search_optimization::replace_stats_cmds_with_tstats]

enabled = <boolean>
* If you are not using summary indexing, enable this setting to improve performance for searches that perform statistical operations only on indexed fields.
* Do not enable this setting if you are dependent on summary indexes. When it is enabled, searches that perform stats operations on summary indexes and which only reference indexed fields will return incorrect results. This occurs because the 'tstats' command does not respect the fields created by summary indexing commands. If you are using summary indexing but still choose to enable this optimization globally, this optimization can be disabled on a per-search basis by appending '! noop search_optimization.replace_stats_cmds_with_tstats=f' to the search string.
* Default: false

[search_optimization::replace_datamodel_stats_cmds_with_tstats]

enabled = <boolean>
* Enables a search language optimization that replaces stats commands with tstats commands in "| datamodel .. | stats" and "| from datamodel .. | stats" SPL strings.
* 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 integer>
* DEPRECATED. Use the 'maxPrdSearchesPerCpu' setting instead.

maxPrdSearchesPerCpu = <unsigned integer>
* 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

[rollup]

minSpanAllowed = <integer>
* Sets the minimum timespan for the scheduled searches that generate metric rollup summaries.
* Each rollup summary uses a scheduled search to provide its metric data point aggregations. The interval of the search matches the span defined for the rollup summary.
* However, when you run large numbers of scheduled searches with short intervals, you can encounter search concurrency problems, where some searches skip scheduled runs.
* To reduce the risk of search concurrency issues, this setting ensures that the rollup summaries created for your have longer spans.
* Do not set below 60 seconds.
* Default: 300

[mcollect]

always_use_single_value_output = <boolean>
* When set to true, mcollect outputs metric data points that only have one measure per data point.
* When set to false, mcollect outputs metric data points that can have several measures per data point.
* When your Splunk platform instance is fully upgraded to Splunk 8.0.0, change this setting to 'false'.
* Default: true

Data Fabric Search

[dfs]

* The settings in this stanza specify aspects of the Data Fabric Search (DFS) cluster.
dfc_control_port = <port>
* Sets the listening port for data fabric coordinator (DFC) processes. Enables communication between a DFC process and a corresponding search process (SP).
* The port number is internally auto-incremented by Splunk software when the default port is unavailable. If this happens, limits.conf is not updated with
the selected port number.
* The maximum number of DFC control ports that can be used for data fabric
  search at any given time is set by dfc_num_slots.
  * Default: 17000

**dfc_num_slots = <integer>**
* Sets the maximum number of data fabric coordinator (DFC) processes that can run
  concurrently on each search head. Each process uses a search head 'slot'.
  * Default: 4

**dfs_max_num_keepalives = <integer>**
* Sets the maximum number of keepalive packets to run the DFS search.
  * Default: 10

**dfs_max_reduce_partition_size = <integer>**
* Sets the maximum number of partition size to receive data to run the DFS search.
  * Recommended setting for executor node with 5 Cores and 12GB memory: 150000.
  * Default: 500000

**dfs_max_search_result_size = <integer>**
* Sets the maximum number of results which a DFS search returns.
  * When this value is zero (0), a DFS search returns all the results.
  * Default: 1000000

**dfw_num_slots = <integer>**
* This setting applies only when 'dfw_num_slots_enabled' is set to "true" or
  when search head clustering is enabled in your Splunk implementation.
  * If you have enabled search head clustering, this setting sets the maximum
    number of data fabric coordinator (DFC) processes that can run concurrently
    across the search head cluster.
  * If you have disabled search head clustering, the value of 'dfw_num_slots'
    is equal to 'dfc_num_slots'.
  * When multiple deployments are utilizing the same DFS cluster, this setting
    can help resolve concurrent search issues.
  * Default : 10

**dfw_num_slots_enabled = <boolean>**
* Set this to "true" to enable the use of 'dfw_num_slots'.
  * Default: false

**dfw_receiving_data_port = <port>**
* Sets the listening port for data fabric worker (DFW) nodes. Receives
  redistributed data from Splunk indexers.
  * The port number is internally auto-incremented by Splunk software when the
    default port is unavailable. If this happens, limits.conf is not updated with
    the selected port number.
  * Default: 17500

**dfw_receiving_data_port_count = <integer>**
* Maximum number of ports that Splunk software checks for availability, starting from
  the default port set in the parameter 'dfw_receiving_data_port'.
  * If the 'dfw_receiving_data_port_count' is set to 0, Splunk software checks for any
    available port without any upper limit.
  * Default: 0

**dfs_remote_search_timeout = <integer>**
* The amount of time (in seconds) to wait because the search run on the
  DFS worker has not received the new results from any of the indexers.
  * Default: 600

**dfs_max_remote_pipeline = <integer>**
* Controls the number of search pipelines launched at the indexer during a DFS search.
Increasing the number of search pipelines typically helps improve search performance, but requires additional thread and memory usage. Depending on data volume and cardinality, modifying this setting may lead to slower searches or unread records.

Default: 12

dfs_meta_phase_exec_timeout = <integer>
Sets the time (in seconds) to wait for various meta phase processes to complete during a federated search.
Default: 300

limits.conf.example

# Version 8.0.0
# 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 = 10000000

[kv]
# when non-zero, the point at which kv should stop creating new columns
maxcols = 512

[rare]
maxresultrows = 50000
# maximum distinct value vectors to keep track of
maxvalues = 100000
maxvaluesize = 1000

[restapi]
# maximum result rows to be returned by /events or /results getters from REST API
maxresultrows = 50000

[search]
# 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

[scheduler]
# 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
savedsearches.conf

The following are the spec and example files for savedsearches.conf.

savedsearches.conf.spec

Version 8.0.0

This file contains possible setting/value pairs for saved search entries in the savedsearches.conf file. You can configure saved searches by creating your own savedsearches.conf file.

There is a default savedsearches.conf file in $SPLUNK_HOME/etc/system/default. To set custom configurations, place a savedsearches.conf file in $SPLUNK_HOME/etc/system/local/. For examples, see the savedsearches.conf.example file. 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, settings are combined. In the case of multiple definitions of the same settings, 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.
The possible settings for the savedsearches.conf file are:

```plaintext
[<stanza name>]
* Create a unique stanza name for each saved search.
* Follow the stanza name with any number of the following settings.
* If you do not specify a setting, Splunk software uses the default.

disabled = [0|1]
* Disable your search by setting 'disabled=1'.
* A disabled search cannot run until it is enabled.
* This setting is typically used to prevent a scheduled search from running on its schedule, without deleting the stanza for the search in the savedsearches.conf file.
* Default: 0

search = <string>
* The actual search string for the saved search.
  For example, 'search = index::sampledata http NOT 500'.
* Your search can include macro searches for substitution.
* To learn more about creating a macro search, search the documentation for "macro search."
* Multi-line search strings currently have some limitations. For example use with the search command '|savedsearch' does not currently work with multi-line search strings.
* No default.

dispatchAs = [user|owner]
* When the saved search is dispatched using the "saved/searches/{name}/dispatch" endpoint, this setting controls what user that search is dispatched as.
* This setting is only meaningful for shared saved searches.
* When dispatched as "user", the search is run as if the requesting user owned the search.
* When dispatched as "owner", the search is run as if the owner of the search dispatched the search, no matter which user requested it.
* If the 'force_saved_search_dispatch_as_user' setting, in the limits.conf file, is set to "true", then the 'dispatchAs' setting is reset to "user" while the saved search is dispatching.
* Default: owner

Scheduling options

enableSched = [0|1]
* Set this to 1 (true) to run your search on a schedule.
* Default: 0

cron_schedule = <cron string>
* The cron schedule that is used to run this search.
* For example: */5 * * * * causes the search to run every 5 minutes.
* You can use standard cron notation to define your scheduled search interval. In particular, cron can accept this type of notation: 00,20,40 * * * *, which runs the search every hour at hh:00, hh:20, hh:40.
* A cron of 03,23,43 * * * * runs the search every hour at hh:03, hh:23, hh:43.
* You should schedule your searches so that they are staggered over time.
  This reduces system load. Running all of the saved searches every 20 minutes (*/20) means all of the searches would launch at hh:00 (20, 40) and might
slow your system every 20 minutes.
* The Splunk cron implementation does not currently support names of
  months or days.
* No default.

schedule = <cron-style string>
* This field is DEPRECATED as of version 4.0.
* For more information, see the pre-4.0 spec file.
* Use 'cron_schedule' to define your scheduled search interval.

allow_skew = <percentage>|<duration-specifier>
* Lets the search scheduler randomly distribute scheduled searches more
  evenly over the scheduled time 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.
* To specify a percentage: Use an integer value followed by the percent '%'
  symbol. This specifies the maximum amount of time to skew, as a percentage of
  the scheduled search period.
* To specify a duration: Use <integer><timescale> to specify a maximum duration.
  Supported units are:
  m, min, minute, mins, minutes
  h, hr, hour, hrs, hours
  d, day, days
  The <timescale> is required and can be omitted only when <integer> is 0.
* Skew examples:
  100% (For an every-5-minute search = 5 minutes maximum)
  50%  (For an every-1-minute search = 30 seconds maximum)
  5m = 5 minutes maximum
  1h = 1 hour maximum
* A value of 0 does not allow a skew to occur.
* Default: 0

max_concurrent = <unsigned integer>
* The maximum number of concurrent instances of this search that the scheduler
  is allowed to run.
* Default: 1

realtime_schedule = [0|1]
* Controls the way the scheduler computes the next run time of a
  scheduled search.
* If set to 1, the scheduler determines the next scheduled search run time
  based on the current time.
  * NOTE: When set to 1, the scheduler might skip some execution periods
    to make sure that the scheduler is executing the searches that are running
    over the most recent time range.
* If set to 0, the scheduler determines the next scheduled search run time
  based on the last run time for the search. This is called continuous
  scheduling.
  * NOTE: When set to 0, the scheduler never skips scheduled execution periods.
    However, the execution of the saved search might fall behind
    depending on the scheduler's load.
* Use continuous scheduling whenever you enable the 'summary index' option.
* The scheduler tries to run searches that have 'realtime_schedule' set to 1 before it runs searches that have continuous scheduling (realtime_schedule = 0).
* Default: 1

**schedule_priority** = [default | higher | highest]
* Raises the scheduling priority of a search:
  * When set to "default", specifies that there is no increase to the scheduling priority.
  * When set to "higher", specifies that the scheduling priority is higher than other searches of the same scheduling tier. While there are four tiers of priority for scheduled searches, only the following are affected by this setting:
    1. Real-Time-Scheduled (realtime_schedule=1).
    2. Continuous-Scheduled (realtime_schedule=0).
  * When set to "highest", specifies that the scheduling priority is higher than other searches regardless of scheduling tier. However, real-time-scheduled searches with 'schedule_priority = highest' always have priority over continuous scheduled searches with 'schedule_priority = highest'.
* The high-to-low order is:
  RTSS(H) > CSS(H) > RTSS(h) > RTSS(d) > CSS(h) > CSS(d)
Where:
  RTSS = real-time-scheduled search
  CSS = continuous-scheduled search
  d = default
  h = higher
  H = highest
* The scheduler honors a non-default priority only when the search owner has the 'edit_search_schedule_priority' capability.
* A non-default priority is mutually exclusive with a non-zero 'schedule_window' (see below). If a user specifies both for a scheduled search, the scheduler honors the priority only.
* However, if a user specifies both settings for a search, but the search owner does not have the 'edit_search_schedule_priority' capability, then the scheduler ignores the priority setting and honors the 'schedule_window'.
* CAUTION: 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.
* Default: default

**schedule_window** = <unsigned integer> | auto
* When schedule_window is non-zero, it indicates to the scheduler that the search does not require a precise start time. This gives the scheduler greater flexibility when it prioritizes searches.
* When 'schedule_window' is set to an integer greater than 0, it specifies the "window" of time (in minutes) that a search may start within.
  * The 'schedule_window' must be shorter than the period of the search.
  * Schedule windows are not recommended for searches that run every minute.
* When set to 0, there is no schedule window. The scheduler starts the search as close to its scheduled time as possible.
* When set to "auto," the scheduler calculates the 'schedule_window' value automatically.
  * For more information about this calculation, see the search scheduler documentation.
* A non-zero 'schedule_window' is mutually exclusive with a non-default 'schedule_priority'. See 'schedule_priority' for details.
* Default: 0 for searches that are owned by users with the 'edit_search_schedule_window' capability. For these searches, this value can be changed.
* Default: auto for searches that are owned by users that do not have the 'edit_search_schedule_window' capability.
For these searches, this setting cannot be changed.

**Workload management options**

workload_pool = <name of workload pool>
* Sets the name of the workload pool to be used by this search.
* There are multiple workload pools defined in the workload_pools.conf file. Each workload pool has different resource limits associated with it, for example, CPU, Memory, etc.
* The search process of this search is launched into the 'workload_pool' specified above.
* The 'workload_pool' used should be defined in the workload_pools.conf file.
* If workload management is enabled and an explicit 'workload_pool' is not specified, the 'default_pool' defined in the workload_pools.conf file is used.

**Notification options**

counttype = number of events | number of hosts | number of sources | custom | always
* Set the type of count for alerting.
* Used with the 'relation' and 'quantity' settings.
* NOTE: If you specify "always," do not set 'relation' or 'quantity'.
* Default: always

relation = greater than | less than | equal to | not equal to | drops by | rises by
* Specifies how to compare against 'counttype'.
* Default: empty string

quantity = <integer>
* Specifies a value for the 'counttype' and 'relation', to determine the condition under which an alert is triggered by a saved search.
* Think of it as a sentence constructed like this: <counttype> <relation> <quantity>. For example, "number of events [is] greater than 10" sends an alert when the count of events is larger than by 10.
* For example, "number of events drops by 10%" sends an alert when the count of events drops by 10%.
* Default: empty string

alert_condition = <search string>
* Contains a conditional search that is evaluated against the results of the saved search. Alerts are triggered if the specified search yields a non-empty search result list.
* Default: empty string

*******

generic action settings.
For a comprehensive list of actions and their arguments, refer to the alert_actions.conf file.
*******

action.<action_name> = 0 | 1
* Indicates whether the action is enabled or disabled for a particular saved search.
* The 'action_name' can be: email | populate_lookup | script | summary_index
* For more about your defined alert actions see the alert_actions.conf file.
* Default: empty string

**action.<action_name>.<parameter> = <value>**
* Overrides an action's parameter, defined in the alert_actions.conf file, with a new <value> for this saved search only.
* Default: empty string

### Settings for email action

**action.email = 0 | 1**
* Enables or disables the email action.
* Default: 0 (false)

**action.email.to = <email list>*
* REQUIRED. This setting is not defined in the alert_actions.conf file.
* Set a comma-delimited list of recipient email addresses.
* Default: empty string

* NOTE: When configured in Splunk Web, the following email settings are written to this conf file only if their values differ from the settings in the alert_actions.conf file.

**action.email.from = <email address>**
* Set an email address to use as the sender's address.
* Default: splunk@<LOCALHOST>
  (or the 'from' setting in the alert_actions.conf file)

**action.email.subject = <string>*
* Set the subject of the email delivered to recipients.
* Default: SplunkAlert-<savedsearchname>
  (or the 'subject' setting in the alert_actions.conf file)

**action.email.mailserver = <string>*
* Set the address of the MTA server to be used to send the emails.
* Defaults to <LOCALHOST>
  (or the 'mailserver' setting in alert_actions.conf file)

**action.email.maxresults = <integer>*
* Set the maximum number of results to email.
* Any alert-level results threshold greater than this number is capped at this level.
* This value affects all methods of result inclusion by email alert: inline, CSV, and PDF.
* NOTE: This setting is affected globally by the 'maxresults' setting in the [email] stanza of the alert_actions.conf file.
* Default: 10000

**action.email.include.results_link = [1|0]*
* Specify whether to include a link to search results in the alert notification email.
* Default: 1 (true)
  (or the 'include.result.link' setting in the alert_actions.conf file)

**action.email.include.search = [1|0]*
* Specify whether to include the query whose results triggered the email.
* Default: 0 (false)
  (or the 'include.search' setting in the alert_actions.conf file)
action.email.include.trigger = [1|0]
* Specify whether to include the alert trigger condition.
  * Default: 0 (false)
    (or the 'include.trigger' setting in the alert_actions.conf file)

action.email.include.trigger_time = [1|0]
* Specify whether to include the alert trigger time.
  * Default: 0 (false) or whatever is set in the alert_actions.conf file

action.email.include.view_link = [1|0]
* Specify whether to include the saved search title and a link for editing the saved search.
  * Default: 1 (true)
    (or the 'include.view_link' setting in the alert_actions.conf file)

action.email.inline = [1|0]
* Specify whether to include search results in the body of the alert notification email.
  * Default: 0 (false)
    (or the 'inline' setting in the alert_actions.conf file)

action.email.sendcsv = [1|0]
* Specify whether to send results as a CSV file.
  * Default: 0
    (or the 'sendcsv' setting in the alert_actions.conf file)

action.email.sendpdf = [1|0]
* Specify whether to send results as a PDF file.
  * Default: 0 (false)
    (or the 'sendpdf' setting in the alert_actions.conf file)

action.email.sendresults = [1|0]
* Specify whether to include search results in the alert notification email.
  * Default: 0 (false)
    (or the 'sendresults' setting in the alert_actions.conf file)

**Settings for script action**

action.script = 0 | 1
* Enables or disables the script action.
  * Specify 1 to enable, 0 to disable.
  * Default: 0 (false)

action.script.filename = <script filename>
* The filename, with no path, of the shell script to run.
* The script should be located in: $SPLUNK_HOME/bin-scripts/
* For system shell scripts on UNIX, or .bat or .cmd file on Windows, there are no further requirements.
* For other types of scripts, the first line should begin with a ! marker, followed by a path to the interpreter that will run the script.
  * Example: !C:\Python27\python.exe
  * Default: empty string
**Settings for lookup action**

action.lookup = 0 | 1
* Enables or disables the lookup action.
* Specify 1 to enable, 0 to disable.
* Default: 0 (false)

action.lookup.filename = <lookup filename>
* Provide the name of the CSV lookup file to write search results to.
  Do not provide a file path.
* Lookup actions can only be applied to CSV lookups.

action.lookup.append = 0 | 1
* Specify whether to append results to the lookup file defined for the
  'action.lookup.filename' setting.
* Default: 0 (false)

**Settings for summary index action**

action.summary_index = 0 | 1
* Enables or disables the summary index action.
* Specify 1 to enable, 0 to disable.
* Default: 0

action.summary_index._name = <index>
* Specifies the name of the summary index where the results of the scheduled
  search are saved.
* Default: summary

action.summary_index.inline = <boolean>
* Specify whether to run the summary indexing action as part of the
  scheduled search.
* NOTE: This option is considered only if the summary index action is enabled
  and is always run (in other words, if 'counttype = always').
* Default: 1 (true)

action.summary_index.<field> = <string>
* Specifies a field/value pair to add to every event that gets summary indexed
  by this search.
* You can define multiple field/value pairs for a single summary index search.

**Settings for lookup table population parameters**

action.populate_lookup = 0 | 1
* Enables or disables the lookup population action.
* Specify 1 to enable, 0 to disable.
* Default: 0

action.populate_lookup.dest = <string>
* Can be one of the following two options:
  * A lookup name from transforms.conf. The lookup name cannot be associated
    with KV store.
* A path to a lookup .csv file that the search results should be copied to, relative to $SPLUNK_HOME.
  * NOTE: This path must point to a .csv file in either of the following directories:
    * etc/system/lookups/
    * etc/apps/<app-name>/lookups
  * NOTE: the destination directories of the above files must already exist.
  * Default: empty string

run_on_startup = <boolean>
* Specifies whether this search runs when the Splunk platform starts or any edit that changes search related arguments happen. This includes search and dispatch.* arguments.
* If set to "true", the search is run as soon as possible during startup or after edit. Otherwise the search is run at the next scheduled time.
* Set 'run_on_startup' to "true" for scheduled searches that populate lookup tables or generate artifacts used by dashboards.
* Default: false

run_n_times = <unsigned integer>
* Runs this search exactly the specified number of times. The search is not run again until the Splunk platform is restarted.
* Default: 0 (infinite).

**dispatch search options**

dispatch.ttl = <integer>[p]
* Indicates the time to live (ttl), in seconds, for the artifacts of the scheduled search, if no actions are triggered.
* If the integer is followed by the letter 'p', the ttl is calculated as a multiple of the execution period for the scheduled search. For example, if the search is scheduled to run hourly and ttl is set to 2p, the ttl of the artifacts is set to 2 hours.
* If an action is triggered, the ttl is changed to the ttl for the action. If multiple actions are triggered, the action with the largest ttl is applied to the artifacts. To set the ttl for an action, refer to the alert_actions.conf.spec file.
* For more information on the ttl for a search, see the limits.conf.spec file [search] stanza ttl setting.
* Default: 2p, which is 2 times the period of the scheduled search

dispatch.buckets = <integer>
* The maximum number of timeline buckets.
* Default: 0

dispatch.max_count = <integer>
* The maximum number of results before finalizing the search.
* Default: 500000

dispatch.max_time = <integer>
* The maximum amount of time, in seconds, before finalizing the search.
* Default: 0

dispatch.lookups = 1| 0
* Enables or disables lookups for this search.
* Specify 1 to enable, 0 to disable.
* Default: 1
dispatch.earliest_time = <time-str>
* Specifies the earliest time for this search. Can be a relative or absolute time.
* If this value is an absolute time, use the 'dispatch.time_format' setting to format the value.
* Default: empty string

dispatch.latest_time = <time-str>
* Specifies the latest time for this saved search. Can be a relative or absolute time.
* If this value is an absolute time, use the 'dispatch.time_format' setting to format the value.
* Default: empty string

dispatch.index_earliest= <time-str>
* Specifies the earliest index time for this search. Can be a relative or absolute time.
* If this value is an absolute time, use the 'dispatch.time_format' setting to format the value.
* Default: empty string

dispatch.index_latest= <time-str>
* Specifies the latest index time for this saved search. Can be a relative or absolute time.
* If this value is an absolute time, use the 'dispatch.time_format' setting to format the value.
* Default: empty string

dispatch.time_format = <time format str>
* Defines the time format that is used to specify the earliest and latest time.
* Default: %FT%T.%Q%:z

dispatch.spawn_process = 1 | 0
* Specifies whether a new search process is started when this saved search is run.
* Default: 1 (true)

dispatch.auto_cancel = <integer>
* Specifies the amount of inactive time, in seconds, after which the job is automatically canceled.
* 0 means to never auto-cancel the job.
* Default: 0

dispatch.auto_pause = <integer>
* Specifies the amount of inactive time, in seconds, after which the search job is automatically paused.
* 0 means to never auto-pause the job.
* To restart a paused search job, specify 'unpause' as an action to POST search/jobs/{search_id}/control.
* auto_pause only goes into effect once. Unpausing after auto_pause does not put auto_pause into effect again.
* Default: 0

dispatch.reduce_freq = <integer>
* Specifies the frequency, in number of intermediary results chunks, that the MapReduce reduce phase should run on the accumulated map values.
* Default: 10

dispatch.rt_backfill = <boolean>
* Specifies whether to do real-time window backfilling for scheduled real-time searches.
* Default: false

dispatch.indexedRealtime = <boolean>
* Specifies whether to use 'indexed-realtime' mode when doing real-time searches.
* Overrides the setting in the limits.conf file for the
  'indexed_realtime_use_by_default' setting in the [realtime] stanza.
* This setting applies to each job.
* See the [realtime] stanza in the limits.conf.spec file for more information.
* Default: The value for 'indexed_realtime_use_by_default' in the limits.conf file

dispatch.indexedRealtimeOffset = <integer>
* Controls the number of seconds to wait for disk flushes to finish.
* Overrides the setting in the limits.conf file for the
  'indexed_realtime_disk_sync_delay' setting in the [realtime] stanza.
* This setting applies to each job.
* See the [realtime] stanza in the limits.conf.spec file for more information.
* Default: The value for 'indexed_realtime_disk_sync_delay' in the limits.conf file

dispatch.indexedRealtimeMinSpan = <integer>
* Minimum seconds to wait between component index searches.
* Overrides the setting in the limits.conf file for the
  'indexed_realtime_default_span' setting in the [realtime] stanza.
* This setting applies to each job.
* See the [realtime] stanza in the limits.conf.spec file for more information.
* Default: The value for 'indexed_realtime_default_span' in the limits.conf file

dispatch.rt_maximum_span = <integer>
* The max seconds allowed to search data which falls behind realtime.
* Use this setting to set a limit, after which events are not longer considered
  for the result set. The search catches back up to the specified delay from
  realtime and uses the default span.
* Overrides the setting in the limits.conf file for the
  'indexed_realtime_maximum_span' setting in the [realtime] stanza.
* This setting applies to each job.
* See the [realtime] stanza in the limits.conf.spec file for more information.
* Default: the value for 'indexed_realtime_maximum_span' in the limits.conf file

dispatch.sample_ratio = <integer>
* The integer value used to calculate the sample ratio. The formula is 1 / <integer>.
* The sample ratio specifies the likelihood of any event being included in the sample.
* For example, if sample_ratio = 500, each event has a 1/500 chance of being
  included in the sample result set.
* Default: 1

restart_on_searchpeer_add = 1 | 0
* Specifies whether to restart a real-time search managed by the scheduler when
  a search peer becomes available for this saved search.
* NOTE: The peer can be a newly added peer or a peer that has been down and has
  become available.
* Default: 1 (true)

**auto summarization options**

auto_summarize = <boolean>
* Specifies if the scheduler should ensure that the data for this search is
  automatically summarized.
* Default: false

auto_summarize.command = <string>
* A search template to use to construct the auto summarization for this search.
* DO NOT change unless you know what you're doing

**auto_summarize.timespan** = <time-specifier> (, <time-specifier>)*
* Comma delimited list of time ranges that each summarized chunk should span. This comprises the list of available granularity levels for which summaries would be available. For example, a timechart over the last month whose granularity is at the day level should set this to "1d". If you need the same data summarized at the hour level because you need to have weekly charts then use: "1h;1d".

**auto_summarize.cron_schedule** = <cron-string>
* Cron schedule to use to probe or generate the summaries for this search.

**auto_summarize.dispatch.<arg-name>** = <string>
* Any dispatch.* options that need to be overridden when running the summary search.

**auto_summarize.suspend_period** = <time-specifier>
* The amount of time to suspend summarization of this search if the summarization is deemed unhelpful.
  * Default: 24h

**auto_summarize.max_summary_size** = <unsigned integer>
* The minimum summary size when to start testing it's helpfulness.
  * Default: 52428800 (5MB)

**auto_summarize.max_summary_ratio** = <positive decimal>
* The maximum ratio of summary_size/bucket_size when to stop summarization and deem it unhelpful for a bucket.
  * Default: 0.1

**auto_summarize.max_disabled_buckets** = <unsigned integer>
* The maximum number of buckets with the suspended summarization before the summarization search is completely stopped and the summarization of the search is suspended for the value specified in the 'auto_summarize.suspend_period' setting.
  * Default: 2

**auto_summarize.max_time** = <unsigned integer>
* The maximum amount of time that the summary search is allowed to run.
  * NOTE: This is an approximate time and the summarize search will be stopped at clean bucket boundaries.
  * Default: 3600

**auto_summarize.hash** = <string>
* An auto generated setting.

**auto_summarize.normalized_hash** = <string>
* An auto generated setting.

**auto_summarize.max_concurrent** = <unsigned integer>
* The maximum number of concurrent instances of this auto summarizing search, that the scheduler is allowed to run.
  * Default: 1

**auto_summarize.workload_pool** = <name of workload pool>
* Sets the name of the workload pool to be used by this auto summarization.
  * There are multiple workload pools defined in workload_pools.conf.
Each workload pool has different resource limits associated with it, for example, CPU, Memory, etc.
* The search process of this auto summarization will be launched into the workload_pool specified above.
* The workload_pool used should be defined in workload_pools.conf.
* If workload management is enabled and an explicit workload_pool is not specified, the workload rules defined in workload_rules.conf will try to put the search into a proper pool as specified in some rule. If there is no rule defined for this search, the default_pool defined in workload_pools.conf will be used.

**alert suppression/severity/expiration/tracking/viewing settings**

alert.suppress = 0 | 1
* Specifies whether alert suppression is enabled for this scheduled search.
* Default: 0 (false)

alert.suppress.period = <time-specifier>
* Sets the suppression period. Use [number][time-unit] to specify a time.
* For example: 60 = 60 seconds, 1m = 1 minute, 1h = 60 minutes.
* Honored if and only if 'alert.suppress = 1'.
* Default: empty string

alert.suppress.fields = <comma-delimited-field-list>
* List of fields to use when suppressing per-result alerts. This field *must* be specified if the digest mode is disabled and suppression is enabled.
* Default: empty string.

alert.severity = <integer>
* Sets the alert severity level.
* Valid values are: 1-debug, 2-info, 3-warn, 4-error, 5-severe, 6-fatal
* Default: 3

alert.expires = <time-specifier>
* Sets the period of time to show the alert on the Triggered Alerts page.
* Use [number][time-unit] to specify a time.
* For example: 60s = 60 seconds, 1m = 1 minute, 1h = 60 minutes = 1 hour etc
* This setting is only honored when 'alert.track = true' (when the "Add to Triggered Alerts" action is selected for the alert in Splunk Web).
* This property is valid until splunkd restarts. Restart clears the listing of triggered alerts.
* Default: 24h

alert.digest_mode = <boolean>
* Specifies whether Splunk applies the alert actions to the entire result set or to each individual result.
* Default: true

alert.track = <boolean> | auto
* Specifies whether to track the actions triggered by this scheduled search.
* auto - determine whether to track or not based on the tracking setting of each action, do not track scheduled searches that always trigger actions.
* true - force alert tracking.
* false - disable alert tracking for this search.
* Default: auto

alert.display_view = <string>
* Name of the UI view where the emailed link for each result alerts should point to.
* If not specified, the value of the 'request.ui_dispatch_app' setting is used.
* If the 'request.ui_dispatch_app' setting is missing then "search" is used.
alert.managedBy = <string>
* Specifies the feature or component that created the alert.
* Default: empty string

**UI-specific settings**

displayview = <string>
* Defines the default UI view name (not label) in which to load the results.
* Accessibility is subject to the user having sufficient permissions.
* Default: empty string

vfid = <string>
* Defines the view state ID associated with the UI view listed in the
  'displayview' setting.
* Must match up to a stanza in the viewstates.conf file.
* Default: empty string

is_visible = <boolean>
* Specifies whether this saved search should be listed in the visible saved
  search list within apps.
* Saved searches are still visible when accessing the "Searches, reports,
  and alerts" page in Splunk Web.
* Default: true

description = <string>
* Human-readable description of this saved search.
* Default: empty string

request.ui.dispatch_app = <string>
* Specifies a field used by Splunk UI to denote the app that this search
  should be dispatched in.
* Default: empty string

request.ui.dispatch_view = <string>
* Specifies a field used by Splunk UI to denote the view this search should be
  displayed in.
* Default: empty string

**Display Formatting Options**

**General options**
display.general.enablePreview = 0 | 1
display.general.type = [events|statistics|visualizations]
display.general.timeRangePicker.show = 0 | 1
display.general.migratedFromViewState = 0 | 1
display.general.locale = <string>

**Event options**
display.events.fields = [<string>, <string>]*
display.events.type = [raw|list|table]
display.events.rowNumbers = 0 | 1
display.events.maxLines = <integer>
display.events.raw.drilldown = [inner|outer|full|none]
display.events.list.drilldown = [inner|outer|full|none]
display.events.list.wrap = 0 | 1
display.events.table.drilldown = 0 | 1
display.events.table.wrap = 0 | 1

Statistics options
display.statistics.rowNumbers = 0 | 1
display.statistics.wrap = 0 | 1
display.statistics.overlay = [none|heatmap|highlow]
display.statistics.drilldown = [row|cell|none]
display.statistics.totalsRow = 0 | 1
display.statistics.percentagesRow = 0 | 1
display.statistics.show = 0 | 1

Visualization options
display.visualizations.trellis.enabled = 0 | 1
display.visualizations.trellis.scales.shared = 0 | 1
display.visualizations.trellis.size = [small|medium|large]
display.visualizations.trellis.splitBy = <string>
display.visualizations.show = 0 | 1

display.visualizations.type = [charting|singlevalue|mapping|custom]
display.visualizations.chartHeight = <integer>

display.visualizations.charting.chart = [line|area|column|bar|pie|scatter|bubble|radialGauge|fillerGauge|markerGauge]
display.visualizations.charting.chart.stackMode = [default|stacked|stacked100]
display.visualizations.charting.chart.nullValueMode = [gaps|zero|connect]
display.visualizations.charting.chart.overlayFields = <string>
display.visualizations.charting.drilldown = [all|none]
display.visualizations.charting.chart.style = [minimal|shiny]
display.visualizations.charting.layout.splitSeries = 0 | 1
display.visualizations.charting.layout.splitSeries.allowIndependentYRanges = 0 | 1
display.visualizations.charting.legend.mode = [standard|seriesCompare]
display.visualizations.charting.legend.placement = [right|bottom|top|left|none]
display.visualizations.charting.legend.labelStyle.overflowMode = [ellipsisEnd|ellipsisMiddle|ellipsisStart]

display.visualizations.charting.axisTitleX.text = <string>
display.visualizations.charting.axisTitleY.text = <string>
display.visualizations.charting.axisTitleY2.text = <string>
display.visualizations.charting.axisTitleX.visibility = [visible|collapsed]
display.visualizations.charting.axisTitleY.visibility = [visible|collapsed]
display.visualizations.charting.axisTitleY2.visibility = [visible|collapsed]

display.visualizations.charting.axisX.scale = linear|log

display.visualizations.charting.axisY.scale = linear|log

display.visualizations.charting.axisY2.scale = linear|log|inherit

display.visualizations.charting.axisX.abbreviation = none|auto

display.visualizations.charting.axisY.abbreviation = none|auto

display.visualizations.charting.axisY2.abbreviation = none|auto

display.visualizations.charting.axisLabelsX.majorLabelStyle.overflowMode = [ellipsisMiddle|ellipsisNone]

display.visualizations.charting.axisLabelsX.majorLabelStyle.rotation = [-90|-45|0|45|90]

display.visualizations.charting.axisLabelsX.majorUnit = <decimal> | auto

display.visualizations.charting.axisLabelsY.majorUnit = <decimal> | auto


display.visualizations.charting.chart.rangeValues = [<string>, <string>]*/
display.visualizations.charting.chart.bubbleMaximumSize = <integer>
display.visualizations.charting.chart.bubbleMinimumSize = <integer>
display.visualizations.charting.chart.bubbleSizeBy = [area|diameter]
display.visualizations.charting.fieldColors = <string>
display.visualizations.charting.fieldDashStyles = <string>
display.visualizations.charting.lineWidth = <decimal>
display.visualizations.custom.drilldown = [all|none]
display.visualizations.custom.height = <integer>
display.visualizations.custom.type = <string>
display.visualizations.singlevalueHeight = <integer>
display.visualizations.singlevalue.beforeLabel = <string>
display.visualizations.singlevalue.afterLabel = <string>
display.visualizations.singlevalue.underLabel = <string>
display.visualizations.singlevalue.unit = <string>
display.visualizations.singlevalue.unitPosition = [before|after]
display.visualizations.singlevalue.drilldown = [all|none]
display.visualizations.singlevalue.colorMode = [block|none]
display.visualizations.singlevalue.rangeValues = [<string>, <string>]*/
display.visualizations.singlevalue.rangeColors = [<string>, <string>]*/
display.visualizations.singlevalue.trendColorInterpretation = [standard|inverse]
display.visualizations.singlevalue.showTrendIndicator = 0 | 1
display.visualizations.singlevalue.showSparkline = 0 | 1
display.visualizations.singlevalue.trendDisplayMode = [percent|absolute]
display.visualizations.singlevalue.colorBy = [value|trend]
display.visualizations.singlevalue.useColors = 0 | 1
display.visualizations.singlevalue.numberPrecision = [0|0.0|0.00|0.000|0.0000]
display.visualizations.singlevalue.useThousandSeparators = 0 | 1
display.visualizations.mapHeight = <integer>
display.visualizations.mapping.type = [marker|choropleth]
display.visualizations.mapping.drilldown = [all|none]
display.visualizations.mapping.map.center = (<decimal>,<decimal>)
display.visualizations.mapping.map.zoom = <integer>
display.visualizations.mapping.map.scrollZoom = 0 | 1
display.visualizations.mapping.map.panning = 0 | 1
display.visualizations.mapping.choroplethLayer.colorMode = [auto|sequential|divergent|categorical]
display.visualizations.mapping.choroplethLayer.maximumColor = <string>
display.visualizations.mapping.choroplethLayer.minimumColor = <string>
display.visualizations.mapping.choroplethLayer.colorBins = <integer>
display.visualizations.mapping.choroplethLayer.neutralPoint = <decimal>
display.visualizations.mapping.choroplethLayer.shapeOpacity = <decimal>
display.visualizations.mapping.choroplethLayer.showBorder = 0 | 1
display.visualizations.mapping.markerLayer.markerOpacity = <decimal>
display.visualizations.mapping.markerLayer.minSize = <integer>
display.visualizations.mapping.markerLayer.maxSize = <integer>
display.visualizations.mapping.legend.placement = [bottomright|none]
display.visualizations.mapping.data.maxClusters = <integer>
display.visualizations.mapping.showTiles = 0 | 1
display.visualizations.mapping.tileLayer.tileOpacity = <decimal>
display.visualizations.mapping.tileLayer.url = <string>
display.visualizations.mapping.tileLayer.minZoom = <integer>
display.visualizations.mapping.tileLayer.maxZoom = <integer>

Patterns options
display.page.search.patterns.sensitivity = <decimal>

Page options
display.page.search.mode = [fast|smart|verbose]
* This setting has no effect on saved search execution when dispatched by the scheduler. It only comes into effect when the search is opened in the UI and run manually.
Table format settings

Format options
display.statistics.format.<index> = [color|number]
display.statistics.format.<index>.field = <string>
display.statistics.format.<index>.fields = [<string>{, <string>}*]

Color format options
display.statistics.format.<index>.scale = [category|linear|log|minMidMax|sharedCategory|threshold]
display.statistics.format.<index>.colorPalette = [expression|list|map|minMidMax|sharedList]

Number format options
display.statistics.format.<index>.precision = <integer>
display.statistics.format.<index>.useThousandSeparators = <boolean>
display.statistics.format.<index>.unit = <string>
display.statistics.format.<index>.unitPosition = [before|after]

Scale options for 'category'
display.statistics.format.<index>.scale.categories = [<string>{, <string>}*]

Scale options for 'log'
display.statistics.format.<index>.scale.base = <integer>

Scale options for 'minMidMax'
display.statistics.format.<index>.scale.minType = [number|percent|percentile]
display.statistics.format.<index>.scale.minValue = <decimal>
display.statistics.format.<index>.scale.midType = [number|percent|percentile]
display.statistics.format.<index>.scale.midValue = <decimal>
display.statistics.format.<index>.scale.maxType = [number|percent|percentile]
display.statistics.format.<index>.scale.maxValue = <decimal>

Scale options for 'threshold'
display.statistics.format.<index>.scale.thresholds = [<decimal>{, <decimal>}*]

Color palette options for 'expression'
display.statistics.format.<index>.colorPalette.rule = <string>

Color palette options for 'list'
display.statistics.format.<index>.colorPalette.colors = [<hex>{, <hex>}*]
display.statistics.format.<index>.colorPalette.interpolate = <boolean>

Color palette options for 'map'
display.statistics.format.<index>.colorPalette.colors = [<string>:<hex>{, <string>:<hex>}*]

Color palette options for 'minMidMax'
display.statistics.format.<index>.colorPalette.minColor = <hex>
display.statistics.format.<index>.colorPalette.midColor = <hex>
display.statistics.format.<index>.colorPalette.maxColor = <hex>
Other settings

`embed.enabled = 0 | 1`
* Specifies whether a saved search is shared for access with a guestpass.
* Search artifacts of a search can be viewed using a guestpass only if:
  * A token has been generated that is associated with this saved search.
  * The token is associated with a particular user and app context.
  * The user to whom the token belongs has permissions to view that search.
  * The saved search has been scheduled and there are artifacts available.
  * Only artifacts are available using guestpass. A search is never dispatched.
  * The saved search is not disabled, it is scheduled.
  * The saved search is not real-time.
  * The saved search is not an alert.

`defer_scheduled_searchable_idxc = <boolean>`
* Specifies whether to defer a continuous saved search during a searchable rolling restart or searchable rolling upgrade of an indexer cluster.
* Note: When disabled, a continuous saved search might return partial results.
* Default: true (enabled)

DFS options

`federated.provider = <federated-provider-stanza>`
* Identifies the federated provider where this search has to run.
* Select a federated provider stanza defined in your federated.conf file.
* No default.

Deprecated settings

`sendresults = <boolean>`
* Use the 'action.email.sendresult' setting.

`action_rss = <boolean>`
* Use the 'action.rss' setting.

`action_email = <string>`
* Use the 'action.email' and 'action.email.to' settings.

`role = <string>`
* See saved search permissions.

`userid = <string>`
* See saved search permissions.

`query = <string>`
* use the 'search' setting.

`nextrun = <integer>`
* Not used anymore, the scheduler maintains this info internally.

`qualifiedSearch = <string>`
* Not used anymore, Splunk software computes this value during runtime.
savedsearches.conf.example

#   Version 8.0.0
#
# This file contains example saved searches and alerts.
#
# To use one or more of these configurations, copy the configuration block into
# savedsearches.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
#
# The following searches are example searches. To create your own search,
# modify the values by following the spec outlined in savedsearches.conf.spec.

[Daily indexing volume by server]
save = index=_internal todaysBytesIndexed LicenseManager-Audit NOT source=web_service.log NOT source=web_access.log | eval Daily_Indexing_Volume_in_MB = todaysBytesIndexed/1024/1024 | timechart avg(Daily_Indexing_Volume_in_MB) by host
dispatch.earliest_time = -7d

[Errors in the last 24 hours]
save = error OR failed OR severe OR ( sourcetype=access_* ( 404 OR 500 OR 503 ) )
dispatch.earliest_time = -1d

[Errors in the last hour]
save = error OR failed OR severe OR ( sourcetype=access_* ( 404 OR 500 OR 503 ) )
dispatch.earliest_time = -1h

[KB indexed per hour last 24 hours]
save = index=_internal metrics group=per_index_thruput NOT debug NOT sourcetype=splunk_web_access | timechart fixedrange=t span=1h sum(kb) | rename sum(kb) as totalKB
dispatch.earliest_time = -1d

[Messsages by minute last 3 hours]
save = index=_internal eps "group=per_source_thruput" NOT filetracker | eval events=eps*kb/kbps | timechart fixedrange=t span=1m s sum(events) by series
dispatch.earliest_time = -3h

[Splunk errors last 24 hours]
save = index=_internal " error " NOT debug source=/splunkd.log*
dispatch.earliest_time = -24h

## Federated Datasets
[search-sf-usage]
save = search index=sf_index | stats count by user
defederated.provider = deployment-sf-search

[search-sf-hr]
save = | union [search index=sf_index] [search index=hr_index] | stats count by user
defederated.provider = deployment-sf-hr

[search-sr]
search = search index=sr_index | join left=L right=R L.user=R.group [ search index=hr_index]
federated.provider = deployment-sr-search

server.conf

The following are the spec and example files for server.conf.

server.conf.spec

Version 8.0.0

OVERVIEW

This file contains settings and values to configure server options in server.conf.

Each stanza controls different search commands settings.

There is a server.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 server.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 server.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

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 configuration file should have at most one default stanza. If you have multiple default stanzas, settings are combined. If you have multiple definitions of the same settings, 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.

General Server Configuration

[general]
serverName - <ASCII string>
* The name that identifies this Splunk software instance for features such as
distributed search.
* Cannot be an empty string.
* Can contain environment variables.
* After any environment variables are expanded, the server name
(if not an IPv6 address) can only contain letters, numbers, underscores,
dots, and dashes. The server name must start with a letter, number, or an
underscore.
* Default: <hostname>-<user_running_splunk>

hostnameOption = <ASCII string>
* This option lets you specify the details in the server name that
identifies this Splunk instance.
* Applies to Windows only.
* Can be one of the following: "fullyqualifiedname", "clustername", "shortname".
* Cannot be an empty string.

sessionTimeout = <nonnegative integer>[s|m|h|d]
* The amount of time before a user session times out, expressed as a
search-like time range.
* Examples include "24h" (24 hours), "3d" (3 days),
"7200s" (7200 seconds, or two hours)
* Default: "1" (1 hour)

trustedIP = <IP address>
* All logins from specified IP addresses are trusted. This means a
password is no longer required.
* Only set this if you are using Single Sign-On (SSO).

allowRemoteLogin = always|never|requireSetPassword
* Controls remote management by restricting general login. Note that this
does not apply to trusted SSO logins from a trustedIP.
* When set to "always", all remote login attempts are allowed.
* When set to "never", only local logins to splunkd are allowed. Note that this
still allows remote management through Splunk Web if Splunk Web is on
the same server.
* If set to "requireSetPassword":
  * In the free license, remote login is disabled.
  * In the pro license, remote login is disabled for the "admin" user if
    the default password of "admin" has not been changed.
* NOTE: As of version 7.1, Splunk software does not support the use of default
passwords. The "requireSetPassword" value is deprecated and might be removed
in the future.
* Default: requireSetPassword

tar_format = gnutar|ustar
* Sets the default TAR format.
* Default: gnutar

access_logging_for_phonehome = <boolean>
* Enables/disables logging to the splunkd_access.log file for client phonehomes.
* Default: true (logging enabled)

hangup_after_phonehome = <boolean>
* Controls whether or not the deployment server hangs up the connection
after the phonehome is done.
* By default, persistent HTTP 1.1 connections are used with the server to
handle phonehomes. This might show higher memory usage if you have a large
number of clients.
* If you have more than the maximum recommended concurrent TCP connection
deployment clients, persistent connections can not help with the reuse of
connections. Setting this attribute to false helps bring down memory usage.  
* Default: false (persistent connections for phonehome)

pass4SymmKey = <password>
* Authenticates traffic between:  
  * License master and its license slaves.  
  * Members of a cluster.  
  * Deployment server (DS) and its deployment clients (DCs).
* When authenticating members of a cluster, clustering might override the 
  passphrase specified in the clustering stanza. A clustering searchhead 
  connecting to multiple masters might further override in the 
  [clustermaster:stanza1] stanza.
* When authenticating deployment servers and clients, by default, DS-DCs 
  passphrase authentication is disabled. To enable DS-DCs passphrase 
  authentication, you must also add the following line to the [broker:broker] 
  stanza in the restmap.conf file: requireAuthentication = true
* In all scenarios, every node involved must set the same passphrase in 
  the same stanzas. For example in the [general] stanza and/or 
  [clustering] stanza. Otherwise, the respective communication does not proceed:
  - licensing and deployment in the case of the [general] stanza  
  - clustering in case of the [clustering] stanza
* Unencrypted passwords must not begin with "$1$". This is used by 
  Splunk software to determine if the password is already encrypted.

listenOnIPv6 = no|yes|only
* By default, splunkd listens for incoming connections (both REST and 
  TCP inputs) using IPv4 only.  
* When you set this value to "yes", splunkd simultaneously listens for 
  connections on both IPv4 and IPv6.  
* To disable IPv4 entirely, set listenOnIPv6 to "only". This causes splunkd 
  to exclusively accept connections over IPv6. You might need to change 
  the mgmtHostPort setting in the web.conf file. Use '[:1]' instead of 
  '127.0.0.1'.
* Any setting of SPLUNK_BINDIP in your environment or the 
  splunk-launch.conf file overrides the listenOnIPv6 value. 
  In this case splunkd listens on the exact address specified.

connectUsingIpVersion = auto|4-first|6-first|4-only|6-only
* When making outbound TCP connections for forwarding event data, making 
  distributed search requests, etc., this setting controls whether the 
  connections are made using IPv4 or IPv6.  
* Connections to literal addresses are unaffected by this setting. For 
  example, if a forwarder is configured to connect to "10.1.2.3" the 
  connection is made over IPv4 regardless of this setting.  
* "auto:"  
  * If listenOnIPv6 is set to "no", the Splunk server follows the 
    "4-only" behavior.  
  * If listenOnIPv6 is set to "yes", the Splunk server follows "6-first"  
  * If listenOnIPv6 is set to "only", the Splunk server follow 
    "6-only" behavior.  
* "4-first:" If a host is available over both IPv4 and IPv6, then 
  the Splunk server connects over IPv4 first and falls back to IPv6 if the 
  connection fails.  
* "6-first": splunkd tries IPv6 first and fallback to IPv4 on failure.  
* "4-only": splunkd only attempts to make connections over IPv4.  
* "6-only": splunkd only attempts to connect to the IPv6 address.  
* Default: auto. This means that the Splunk server selects a reasonable 
  value based on the listenOnIPv6 setting.

guid = <globally unique identifier for this instance>
* This setting (as of version 5.0) belongs in the [general] stanza of 
  SPLUNK_HOME/etc/instance.cfg file. See the .spec file of instance.cfg for
useHTTPServerCompression = <boolean>
* Specifies whether the splunkd HTTP server should support gzip content encoding. For more info on how content encoding works, see Section 14.3 of Request for Comments: 2616 (RFC2616) on the World Wide Web Consortium (W3C) website.
* Default: true

defaultHTTPServerCompressionLevel = <integer>
* If the useHTTPServerCompression setting is enabled (it is enabled by default), this setting controls the compression level that the Splunk server attempts to use.
* This number must be between 1 and 9.
* Higher numbers produce smaller compressed results but require more CPU usage.
* Default: 6 (This is appropriate for most environments)

skipHTTPCompressionAcl = <network_acl>
* Lists a set of networks or addresses to skip data compression. These are addresses that are considered so close that network speed is never an issue, so any CPU time spent compressing a response is wasteful.
* Note that the server might still respond with compressed data if it already has a compressed version of the data available.
* These rules are separated by commas or spaces.
* Each rule can be in the following forms:
  1. A single IPv4 or IPv6 address, for example: "10.1.2.3", "fe80::4a3"
  2. A CIDR block of addresses, for example: "10/8", "fe80:1234/32"
  3. A DNS name, possibly with a '*' used as a wildcard, for example: "myhost.example.com", "*.splunk.com"
  4. A single '*' which matches anything
* Entries can also be prefixed with '!' to negate their meaning.
* Default: localhost addresses

legacyCiphers = decryptOnly|disabled
* This setting controls how Splunk software handles support for legacy encryption ciphers.
* If set to "decryptOnly", Splunk software supports decryption of configurations that have been encrypted with legacy ciphers. It encrypts all new configurations with newer and stronger ciphers.
* If set to "disabled", Splunk software neither encrypts nor decrypts configurations that have been encrypted with legacy ciphers.
* Default: decryptOnly

site = <site-id>
* Specifies the site that this Splunk instance belongs to when multisite is enabled.
* Valid values for site-id include site0 to site63
* The special value "site0" can be set only on search heads or on forwarders that are participating in indexer discovery.
* For a search head, "site0" disables search affinity.
* For a forwarder participating in indexer discovery, "site0" causes the forwarder to send data to all peer nodes across all sites.

useHTTPClientCompression = true|false|on-http|on-https
* Specifies whether gzip compression should be supported when splunkd acts as a client (including distributed searches). Note: For the content to be compressed, the HTTP server that the client is connecting to should also support compression.
* If the connection is being made over https and "useClientSSLCompression=true", then setting "useHTTPClientCompression=true" results in double compression work without much compression gain. To
mitigate this, set this value to "on-http" (or to "true", and useClientSSLCompressaion to "false").
* Default: false

embedSecret = <string>
* When using report embedding, normally the generated URLs can only be used on the search head that they were generated on.
* If "embedSecret" is set, then the token in the URL is encrypted with this key. Then other search heads with the exact same setting can also use the same URL.
* This is needed if you want to use report embedding across multiple nodes on a search head pool.

parallelIngestionPipelines = <integer>
* The number of discrete data ingestion pipeline sets to create for this instance.
* A pipeline set handles the processing of data, from receiving streams of events through event processing and writing the events to disk.
* An indexer that operates multiple pipeline sets can achieve improved performance with data parsing and disk writing, at the cost of additional CPU cores.
* For most installations, the default setting of "1" is optimal.
* Use caution when changing this setting. Increasing the CPU usage for data ingestion reduces available CPU cores for other tasks like searching.
* NOTE: Enabling multiple ingestion pipelines can change the behavior of some settings in other configuration files. Each ingestion pipeline enforces the limits of the following settings independently:
  1. maxKBps (in the limits.conf file)
  2. max_fd (in the limits.conf file)
  3. maxHotBuckets (in the indexes.conf file)
  4. maxHotSpanSecs (in the indexes.conf file)
* Default: 1

pipelineSetSelectionPolicy = <round_robin | weighted_random>
* Specifies the pipeline set selection policy to use while selecting pipeline sets for new inputs.
* If set to round_robin, the incoming inputs are assigned to pipeline sets in a round robin fashion.
* If set to weighted_random, the incoming inputs are assigned to pipeline sets using a weighted random scheme designed to even out the CPU usage of each pipeline set.
* NOTE: This setting only takes effect when parallelIngestionPipelines is greater than 1.
* Default: round_robin

pipelineSetWeightsUpdatePeriod = <number>
* The interval, in seconds, when pipeline set weights are recalculated for the weighted_random pipeline set selection policy.
* Reducing this interval causes pipeline set weights to be re-evaluated more frequently, thereby enabling the system to react more quickly to changes in dutycycle estimation.
* Increasing this interval causes pipeline set weights to be re-evaluated less frequently, thereby reducing the likelihood of the system responding to bursty events.
* Default: 30

pipelineSetNumTrackingPeriods = <number>
* The number of look-back periods, of interval pipelineSetWeightsUpdatePeriod, that are used to keep track of incoming ingestion requests for pipeline sets.
* This information is used as a heuristic to calculate the pipeline set weights at every expiry of pipelineSetWeightsUpdatePeriod.
* Default: 5
instanceType = <string>
* Should not be modified by users.
* Informs components (such as the Splunk Web Manager section) which environment the Splunk server is running in, to allow for more customized behaviors.
* Default: "download"

requireBootPassphrase = <boolean>
* Prompt the user for a boot passphrase when starting splunkd.
* Splunkd uses this passphrase to grant itself access to platform-provided secret storage facilities, like the GNOME keyring.
* For more information about secret storage, see the [secrets] stanza in $SPLUNK_HOME/etc/system/README/authentication.conf.spec.
* Default (if Common Criteria mode is enabled): true
* Default (if Common Criteria mode is disabled): false

remoteStorageRecreateIndexesInStandalone = <boolean>
* Controls re-creation of remote storage enabled indexes in standalone mode.
* Default: true

cleanRemoteStorageByDefault = <boolean>
* Allows 'splunk clean eventdata' to clean the remote indexes when set to true.
* Default: false

recreate_index_fetch_bucket_batch_size = <positive_integer>
* Controls the maximum number of bucket IDs to fetch from remote storage as part of a single transaction for a remote storage enabled index.
* Only valid for standalone mode.
* Default: 500

recreate_bucket_fetch_manifest_batch_size = <positive_integer>
* Controls the maximum number of bucket manifests to fetch in parallel from remote storage.
* Only valid for standalone mode.
* Default: 100

splunkd_stop_timeout = <positive_integer>
* The maximum time, in seconds, that splunkd waits for a graceful shutdown to complete before splunkd forces a stop.
* Default: 360 (6 minutes)

python.version = {python2|python3|force_python3}
* For Python scripts only, sets the default Python version to use.
* Can be overridden by other 'python.version' values elsewhere, with the following exception:
* If you set to "force_python3", the system always uses Python 3, and ignores 'python.version' values that you set elsewhere.
* Default: python2

**Deployment Configuration details**

deployment

pass4SymmKey = <passphrase string>
* Authenticates traffic between the deployment server (DS) and its deployment clients (DCs).
* By default, DS-DCs passphrase authentication key is disabled. To enable DS-DCs passphrase authentication, you must *also* add the following line to the [broker:broker] stanza in the restmap.conf file:
requireAuthentication = true
  * If the key is not set in the [deployment] stanza, the key is looked for in the [general] stanza.
  * NOTE: Unencrypted passwords must not begin with "$1$", because this is used by Splunk software to determine if the password is already encrypted.

**SSL Configuration details**

[sslConfig]
* Set SSL for communications on Splunk back-end under this stanza name.
  * NOTE: To set SSL (for example HTTPS) for Splunk Web and the browser, use the web.conf file.
  * Follow this stanza name with any number of the following attribute/value pairs.
  * If you do not specify an entry for each attribute, the default value is used.

enableSplunkdSSL = <boolean>
  * Enables/disables SSL on the splunkd management port (8089) and KV store port (8191).
  * NOTE: Running splunkd without SSL is not recommended.
  * Distributed search often performs better with SSL enabled.
  * Default: true

useClientSSLCompression = <boolean>
  * Turns on HTTP client compression.
  * Server-side compression is turned on by default. Setting this on the client-side enables compression between server and client.
  * Enabling this potentially gives you much faster distributed searches across multiple Splunk instances.
  * Default: true

useSplunkdClientSSLCompression = <boolean>
  * Controls whether SSL compression is used when splunkd is acting as an HTTP client, usually during certificate exchange, bundle replication, remote calls, etc.
  * This setting is effective if, and only if, useClientSSLCompression is set to "true".
  * NOTE: splunkd is not involved in data transfer in distributed search, the search in a separate process is.
  * Default: true

sslVersions = <versions_list>
  * Comma-separated list of SSL versions to support for incoming connections.
  * 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 a version is prefixed with "-" it is removed from the list.
  * SSLv2 is always disabled; "-ssl2" is accepted in the version list but does nothing.
  * When configured in FIPS mode, "ssl3" is always disabled regardless of this configuration.
  * Default: The default can vary (see the 'sslVersions' setting in the $SPLUNK_HOME/etc/system/default/server.conf file for the current default)

sslVersionsForClient = <versions_list>
  * Comma-separated list of SSL versions to support for outgoing HTTP connections
from splunkd. This includes distributed search, deployment client, etc.
* This is usually less critical, since SSL/TLS always picks the highest
  version both sides support. However, you can use this setting to prohibit
  making connections to remote servers that only support older protocols.
* The syntax is the same as the 'sslVersions' setting above.
* NOTE: For forwarder connections, there is a separate 'sslVersions'
  setting in the outputs.conf file. For connections to SAML servers, there
  is a separate 'sslVersions' setting in the authentication.conf file.
* Default: The default can vary (see the 'sslVersionForClient' setting in
  the $SPLUNK_HOME/etc/system/default/server.conf file for the
  current default)

supportSSLV3Only = <boolean>
* DEPRECATED. SSLv2 is disabled. The exact set of SSL versions
  allowed is configurable using the 'sslVersions' setting above.

sslVerifyServerCert = <boolean>
* This setting is used by distributed search and distributed
  deployment clients.
  * For distributed search: Used when making a search request
    to another server in the search cluster.
  * For distributed deployment clients: Used when polling a
    deployment server.
  * If set to true, make sure that the connected server is
    authenticated. Both the common name and the alternate name
    of the server are checked for a match if they are specified
    in this configuration file. A certificate is considered
    verified if either is matched.
* Default: false

sslCommonNameToCheck = <commonName1>, <commonName2>, ...
* If set, and 'sslVerifyServerCert' is set to "true",
  splunkd limits most outbound HTTPS connections to hosts which
  use a certificate with one of the listed common names.
* The most important scenario is distributed search.
* This feature does not work with the deployment server and client
  communication over SSL.
* Optional.
* No default (no common name checking.)

sslCommonNameList = <commonName1>, <commonName2>, ...
* DEPRECATED. Use the 'sslCommonNameToCheck' setting instead.

sslAltNameToCheck = <alternateName1>, <alternateName2>, ...
* If this value is set, and 'sslVerifyServerCert' is set to true,
  splunkd also verifies 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 certificates beyond the commonName. A common
    practice for HTTPS certificates is to use these values to
    store additional valid hostnames or domains where the
    certificate should be considered valid.
  * Accepts a comma-separated list of Subject Alternate Names to
    consider as valid.
  * Items in this list are never validated against the SSL Common Name.
* This feature does not work with the deployment server and client
  communication over SSL.
* Optional.
* No default (no alternate name checking.)

requireClientCert = <boolean>
* Requires that any HTTPS client that connects to a splunkd internal HTTPS server has a certificate that was signed by a CA (Certificate Authority) specified by the 'sslRootCAPath' setting.
  * Used by distributed search: Splunk indexing instances must be authenticated to connect to another splunk indexing instance.
  * Used by distributed deployment: The deployment server requires that deployment clients are authenticated before allowing them to poll for new configurations/applications.
* If set to "true", a client can connect ONLY if a certificate created by our certificate authority was used on that client.
* Default: false

cipherSuite = <cipher suite string>
* If set, Splunk uses the specified cipher string for the HTTP server.
* If not set, Splunk 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.
* Default: The default can vary (See the 'cipherSuite' setting in the $SPLUNK_HOME/etc/system/default/server.conf file 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.

ecdhCurves = <comma-separated list>
* A list of ECDH curves to use for ECDH key negotiation.
* The curves should be specified in the order of preference.
* The client sends these curves as a part of an SSL Client Hello.
* The server supports only the curves specified in the list.
* Splunk software 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
* Example setting: "ecdhCurves = prime256v1,secp384r1,secp521r1"
* Default: The default can vary (See the 'ecdhCurves' setting in the $SPLUNK_HOME/etc/system/default/server.conf file for the current default)

serverCert = <path>
* The full path to the PEM (Privacy-Enhanced Mail) format server certificate file.
* Certificates are auto-generated by splunkd on starting Splunk Enterprise.
* You can replace the default certificate with your own PEM format file.
* Default: $SPLUNK_HOME/etc/auth/server.pem

sslKeysfile = <filename>
* DEPRECATED. Use the 'serverCert' setting instead.
* This file is in the directory specified by the 'caPath' setting (see below).
* Default: server.pem
sslPassword = <password>
* Server certificate password.
* Default: "password"

sslKeysfilePassword = <password>
* DEPRECATED. Use the 'sslPassword' setting instead.

sslRootCAPath = <path>
* Full path to the root CA (Certificate Authority) certificate store on the operating system.
* The <path> must refer to a PEM (Privacy-Enhanced Mail) format file containing one or more root CA certificates concatenated together.
* Required for Common Criteria.
* This setting is valid on Windows machines only if you have not set 'sslRootCAPathHonoredOnWindows' to "false".
* No default.

sslRootCAPathHonoredOnWindows = <boolean>
* DEPRECATED.
* Whether or not the Splunk instance respects the 'sslRootCAPath' setting on Windows machines.
* If you set this setting to "false", then the instance does not respect the 'sslRootCAPath' setting on Windows machines.
* This setting is valid only on Windows, and only if you have set 'sslRootCAPath'.
* When the 'sslRootCAPath' setting is respected, the instance expects to find a valid PEM file with valid root certificates that are referenced by that path. If a valid file is not present, SSL communication fails.
* Default: true

caCertFile = <filename>
* DEPRECATED. Use the 'sslRootCAPath' setting instead.
* Used only if 'sslRootCAPath' is not set.
* File name (relative to 'caPath') of the CA (Certificate Authority) certificate PEM format file containing one or more certificates concatenated together.
* Default: cacert.pem

dhFile = <path>
* PEM (Privacy-Enhanced Mail) format Diffie-Hellman(DH) parameter file name.
* DH group size should be no less than 2048bits.
* This file is required in order to enable any Diffie-Hellman ciphers.
* No default.

caPath = <path>
* DEPRECATED. Use absolute paths for all certificate files.
* If certificate files given by other settings in this stanza are not absolute paths, then they are relative to this path.
* Default: $SPLUNK_HOME/etc/auth

certCreateScript = <script name>
* Creation script for generating certificates on startup of Splunk Enterprise.

sendStrictTransportSecurityHeader = <boolean>
* 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.
* However, this requires a commitment that no non-SSL web hosts ever run on this hostname on any port. For
example, if Splunk Web is in default non-SSL mode this can break the
ability of a browser to connect to it.
* NOTE: Enable with caution.
* Default: false

allowSslCompression = <boolean>
* If set to "true", the server allows clients to negotiate
  SSL-layer data compression.
* KV Store also observes this setting.
* If set to "false", KV Store disables TLS compression.
* Default: true

allowSslRenegotiation = <boolean>
* In the SSL protocol, a client may request renegotiation of the
  connection settings from time to time.
* If set 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

sslClientSessionPath = <path>
* Path where all client sessions are stored for session re-use.
* Used if 'useSslClientSessionCache' is set to "true".
* No default.

useSslClientSessionCache = <boolean>
* Specifies whether to re-use client session.
* When set to "true", client sessions are stored in memory for
  session re-use. This reduces handshake time, latency and
  computation time to improve SSL performance.
* When set to "false", each SSL connection performs a full
  SSL handshake.
* Default: false

sslServerSessionTimeout = <integer>
* Timeout, in seconds, for newly created session.
* If set to "0", disables Server side session cache.
* The openssl default is 300 seconds.
* Default: 300 (5 minutes)

sslServerHandshakeTimeout = <integer>
* The timeout, in seconds, for an SSL handshake to complete between an
  SSL client and the Splunk SSL server.
* If the SSL server does not receive a "Client Hello" from the SSL client within
  'sslServerHandshakeTimeout' seconds, the server terminates
  the connection.
* Default: 60

Data fabric search security configuration

[dfs_security]
* This stanza is for data fabric search (DFS) security configuration.
* Use a combination of the following settings to configure security for your
  DFS implementation. Most of the settings are optional, but some settings
  require that dependent settings also be set.
* All the settings in this stanza require 'enableSplunkdSSL' to be set to
  "true".

tls_enabled = <boolean>
* This setting enables your DFS implementation to use security certificates to secure data flows between compute nodes.
* The type of certificates that this setting enables DFS to use depends on whether additional settings are also set to "true".
* DFS uses default security certificates when 'tls_enabled' is set to "true" but 'use_spark_security_configs' and 'use_node_specific_certificates' are set to "false".
* Set 'use_spark_security_configs' and 'use_node_specific_certificates' to "true" to gain additional degrees of security hardening. These settings require additional Apache Spark configuration. See the setting descriptions for more information.
* See the Data Fabric Search manual for more information about DFS security configuration.
* Default: false

```
tls_protocol = [TLSv1.2]
```
* This setting is required when 'tls_enabled' is set to "true".
* DFS currently supports only TLSv1.2.
* Default: TLSv1.2

```
override_default_certificate = <boolean>
```
* This is an optional setting when:
  * tls_enabled = true
  * use_spark_security_configs = false
  * use_node_specific_certificates = false
* Use this setting if you do not want DFS to use the default security certificates (dfsks.jks and dfsts.jks in SPLUNK_HOME/etc/auth). It causes DFS to push a certificate that you have configured to the data fabric coordinator (DFC) and the data fabric worker (DFW) nodes. This secures communication between the DFC and the DFW nodes as well as the communication between DFW nodes and search peers.
* When you set 'override_default_certificate' to "true", you must define the following DFS security settings.
  * dfs_keystore_path
  * dfs_truststore_path
  * dfs_keystore_password
  * dfs_key_password
  * dfs_truststore_password
* Default: false

```
use_spark_security_configs = <boolean>
```
* This is an optional security setting for DFS. It requires 'tls_enabled' to be set to "true".
* When this setting is set to "true", DFS applies the Apache Spark security configurations instead of its default security settings.
* When this setting is set to "true", you must configure the following settings for each worker node in the Spark cluster:
  * spark.ssl.keyStore
  * spark.ssl.keyPassword
  * spark.ssl.keyStorePassword
  * spark.ssl.trustStore
  * spark.ssl.trustStorePassword
* Default: false

```
use_node_specific_certificates = <boolean>
```
* This is an optional security setting for DFS. It requires 'tls_enabled' to be set to "true".
* When this setting is set to "true", DFS uses certificates that you have configured on the Splunk search head and on each data fabric worker (DFW) node to secure DFS communication.
* When this setting is set to "true", you must configure the following settings for the data fabric coordinator (DFC) in the Splunk search head:
* dfc_keystore_path
* dfc_truststore_path
* dfc_keystore_password
* dfc_key_password
* dfc_truststore_password
* In addition, you must configure the following settings for each of the DFW nodes:
* dfw_keystore_path
* dfw_truststore_path
* dfw_keystore_password
* dfw_key_password
* dfw_truststore_password
* Default: false

verify_search_peer_to_dfw_client_certificate = <boolean>
* This is an optional security setting. It requires 'tls_enabled' to be set to "true".
* When this setting is set to "true", the data fabric worker (DFW) nodes verify the common name and alternative names of the certificates on the search peer they communicate with. DFW nodes can only have outbound TLS connections with search peers whose certificates pass this validation step.
* The common name used in the search peer certificate has to be a value of the 'search_peer_to_dfw_common_name_list' setting.
* The alternative names used in the search peer certificate has to be a value of the 'search_peer_to_dfw_alt_name_list' setting.
* Default: false

search_peer_to_dfw_common_name_list = <commonName1>, <commonName2>, ...
* This setting is required when 'verify_search_peer_to_dfw_client_certificate' is set to "true".
* Provides a comma-separated list of valid common names for the search peer certificate.
* If this list is empty, the DFS workers skip the common name check to the search peers' certificate.
* Default: No default (No checks for search peer certificate's common name)

search_peer_to_dfw_alt_name_list = <alternateName1>, <alternateName2>, ...
* This setting is required when 'verify_search_peer_to_dfw_client_certificate' is set to "true".
* Provides a comma-separated list of valid alternative names for the search peer certificate.
* If this list is empty, the DFS workers skip the alternative name check to the search peers' certificate.
* Default: No default (No checks for search peer certificate's alternative names)

legacy_ca_certificate_folder = <path>
* This is an optional security setting. It helps you provide DFS security for legacy Splunk platform deployments. Set it only when:
* tls_enabled = true
* use_spark_security_configs = false
* use_node_specific_certificates = false
* override_default_certificate = false
* Provides the path to a folder that contains Certificate Authority (CA) certificates for legacy Splunk platform.
* When the data fabric master (DFM) generates default certificates, it uses this setting to add the additional CA certificates in this folder to the trust store.
* Default: SPLUNK_HOME/etc/auth/dfsCACerts

defs_keystore_path = <path>
* This setting is required when:
* tls_enabled = true
* override_default_certificate = true
* use_spark_security_configs = false
* use_node_specific_certificates = false
* Provides the path to the key store file on the search head. Splunk software pushes the key store to the data fabric coordinator (DFC) and the data fabric worker (DFW) nodes.
* The DFC key store file must include the following:
  * The public/private key pair.
  * A certificate export from that key pair and signed by a trusted Certificate Authority (CA).
* This key store is used to secure communication between:
  * the DFC and the DFW nodes.
  * the DFW nodes and the search peers.
* Default: empty string

dfs_truststore_path = <path>
* This setting is required when:
  * tls_enabled = true
  * override_default_certificate = true
  * use_spark_security_configs = false
  * use_node_specific_certificates = false
* Provides the path to the trust store file on the search head. Splunk software pushes the trust store to the data fabric coordinator (DFC) and the data fabric worker (DFW) nodes.
* This trust store file must include the following:
  * All trusted Certificate Authority (CA) certificates.
  * The certificate in dfs_keystore.
* The DFC and the DFW will trust the certificate signed by the CA in this trust store.
* Default: empty string

dfs_keystore_password  = <password>
* This setting is required when:
  * tls_enabled = true
  * override_default_certificate = true
  * use_spark_security_configs = false
  * use_node_specific_certificates = false
* Provides the storage password for dfs_keystore.
* Default: empty string

dfs_key_password  = <password>
* This setting is required when:
  * tls_enabled = true
  * override_default_certificate = true
  * use_spark_security_configs = false
  * use_node_specific_certificates = false
* Provides the password for the public/private key pair entry in dfs_keystore.
* Default: empty string

dfs_truststore_password  = <password>
* This setting is required when:
  * tls_enabled = true
  * override_default_certificate = true
  * use_spark_security_configs = false
  * use_node_specific_certificates = false
* Provides storage password for dfs_truststore.
* Default: empty string

dfc_keystore_path = <path>
* This setting is required when:
  * tls_enabled = true
* use_node_specific_certificates = true
  * Provides the path to the key store file configured on the data fabric coordinator (DFC).
  * The DFC key store file must include the following:
    * The public/private key pair
    * A certificate export from that key pair and signed by a trusted Certificate Authority (CA)
  * Splunk software uses this key store to secure communication between the DFC and the DFW nodes.
  * Default: empty string

  dfc_truststore_path = <path>
  * This setting is required when:
    * tls_enabled = true
    * use_node_specific_certificates = true
  * Provides the path to the trust store file configured on the data fabric coordinator (DFC).
  * The DFC trust store file must include the following:
    * All trusted CA certificates
    * The certificate in dfc_keystore
  * The DFC trusts all certificates signed by the CA in this trust store.
  * Default: empty string

  dfc_keystore_password = <password>
  * This setting is required when:
    * tls_enabled = true
    * use_node_specific_certificates = true
  * Provides the storage password for dfc_keystore.
  * Default: empty string

  dfc_key_password = <password>
  * This setting is required when:
    * tls_enabled = true
    * use_node_specific_certificates = true
  * Provides the password for the public/private key pair entry in dfc_keystore.
  * Default: empty string

  dfc_truststore_password = <password>
  * This setting is required when:
    * tls_enabled = true
    * use_node_specific_certificates = true
  * Provides the storage password for dfc_truststore.
  * Default: empty string

  dfw_keystore_path = <path>
  * This setting is required when:
    * tls_enabled = true
    * use_node_specific_certificates = true
  * Provides the path to the key store file configured on each data fabric worker (DFW) node.
  * The DFW key store file must include the following:
    * The public/private key pair
    * A certificate export from that key pair and signed by a trusted Certificate Authority (CA)
  * Splunk software uses this key store to secure communication between the data fabric coordinator (DFC) and the DFW nodes as well as communication between the DFW nodes and the search peers.
  * Default: empty string

  dfw_truststore_path = <path>
  * This setting is required when:
    * tls_enabled = true
* use_node_specific_certificates = true
* Provides the path to the trust store file configured on the data fabric worker (DFW) nodes.
* The DFW trust store file must include the following:
  * All trusted CA certificates
  * The certificate in dfw_keystore
* Default: empty string

dfw_keystore_password = <password>
* This setting is required when:
  * tls_enabled=true
  * use_node_specific_certificates = true
* Provides the storage password for dfw_keystore.
* Default: empty string

dfw_key_password = <password>
* This setting is required when:
  * tls_enabled=true
  * use_node_specific_certificates = true
* Provides the password for the public/private key pair entry in dfw_keystore.
* Default: empty string

dfw_truststore_password = <password>
* This setting is required when:
  * tls_enabled=true
  * use_node_specific_certificates = true
* Provides the storage password for dfw_truststore.
* Default: empty string

**Splunkd http proxy configuration**

[proxyConfig]
http_proxy = <string>
* If set, splunkd sends all HTTP requests through the proxy server that you specify.
* No default.

https_proxy = <string>
* If set, splunkd sends all HTTPS requests through the proxy server that you specify.
* If not set, splunkd uses the 'http_proxy' setting instead.
* No default.

proxy_rules = <string>
* One or more host names or IP addresses for which splunkd should route HTTPS requests only through the proxy server.
* If set, splunkd uses the proxy server only for endpoints that match the hosts or IP addresses in this value.
* Splunkd does not route requests to either the localhost or loopback addresses through the proxy server.
* Separate multiple entries with commas.
* This setting accepts the following values:
  * '*' (asterisk): Proxy all requests. This is the only wildcard, and it can be used only by itself.
  * <IPv4 or IPv6 address>: Route the request through the proxy if the request is intended for that address.
  * <hostname>/<domain name>: Route the request through the proxy if the request is intended for that host name or domain name.
* Examples:
  * proxy_rules = "wimpy": This matches the host name "wimpy".
**proxy_rules** = "splunk.com": Matches all host names in the splunk.com domain (such as apps.splunk.com, www.splunk.com, etc.)
* Default: *

no_proxy = <string>
* One or more host names or IP addresses for which splunkd should explicitly bypass the proxy server for HTTPS requests.
* If set, splunkd does not route requests to matching host names and IP addresses through the proxy server.
* This setting overrides the 'proxy_rules' setting. If a host name or IP address is in both settings, splunkd does not route requests for that host name or IP address through the proxy server.
* Splunkd does not route requests to either the localhost or loopback addresses.
* Separate multiple entries with commas.
* This setting accepts the following values:
  * '*' (asterisk): Proxy all requests. This is the only wildcard, and it can be used only by itself.
  * <IPv4 or IPv6 address>: Route the request through the proxy if the request is intended for that address.
  * <hostname>/<domain name>: Route the request through the proxy if the request is intended for that host name or domain name.
* Examples:
  * no_proxy = "wimpy": This matches the host name "wimpy".
  * no_proxy = "splunk.com": Matches all host names in the splunk.com domain (such as apps.splunk.com, www.splunk.com, etc.)
* Default: localhost, 127.0.0.1, ::1

**Splunkd HTTP server configuration**

[httpServer]
* Set stand-alone HTTP settings for splunkd 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, splunkd uses the default value.

atomFeedStylesheet = <string>
* Defines the stylesheet relative URL to apply to default Atom feeds.
* Set to 'none' to stop writing out xsl-stylesheet directive.
* Default: /static/atom.xsl

max-age = <nonnegative integer>
* Set the maximum time, in seconds, to cache a static asset served off of the '/static' directory.
* This value is passed along in the 'Cache-Control' HTTP header.
* Default: 3600 (60 minutes)

follow-symlinks = <boolean>
* Specifies whether the static file handler (serving the '/static' directory) follows filesystem symlinks when serving files.
* Default: false

disableDefaultPort = <boolean>
* If set to "true", turns off listening on the splunkd management port, which is 8089 by default.
* NOTE: Changing this setting is not recommended.
* This is the general communication path to splunkd. If it is disabled,
there is no way to communicate with a running splunk instance.
* This means many command line splunk invocations cannot function,
  Splunk Web cannot function, the REST interface cannot function, etc.
* If you choose to disable the port anyway, understand that you are
  selecting reduced Splunk functionality.
* Default: false

acceptFrom = <network_acl> ...
* Lists a set of networks or addresses from which to accept connections.
* 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)

streamInWriteTimeout = <positive number>
* The timeout, in seconds, for uploading data to the http server.
* When uploading data to http server, if the http server is unable
  to write data to the receiver for the specified value, the operation
  aborts.
* Default: 5

max_content_length = <integer>
* Maximum content length, in bytes.
* HTTP requests over the size specified are rejected.
* This setting exists to avoid allocating an unreasonable amount
  of memory from web requests.
* In environments where indexers have enormous amounts of RAM, this
  number can be reasonably increased to handle large quantities of
  bundle data.
* Default: 2147483648 (2GB)

maxSockets = <integer>
* The number of simultaneous HTTP connections that Splunk Enterprise accepts
  simultaneously. You can limit this number to constrain resource usage.
* If set to 0, Splunk Enterprise automatically sets maxSockets to
  one third of the maximum allowable open files on the host.
* If this number is less than 50, it is set to 50.
* If this number is greater than 400000, it is set to 400000.
* If set to a negative number, no limit is enforced.
* Default: 0

maxThreads = <integer>
* The number of threads that can be used by active HTTP transactions.
  You can limit this number to constrain resource usage.
* If set to 0, Splunk Enterprise automatically sets the limit to
  one third of the maximum allowable threads on the host.
* If this number is less than 20, it is set to 20. If this number is
  greater than 150000, it is set to 150000.
* If maxSockets is not negative and maxThreads is greater than maxSockets, then
  Splunk Enterprise sets maxThreads to be equal to maxSockets.
* If set to a negative number, no limit is enforced.
* Default: 0
keepAliveIdleTimeout = <integer>
* How long, in seconds, that the Splunkd HTTP server allows a keep-alive
  connection to remain idle before forcibly disconnecting it.
* If this number is less than 7200, it is set to 7200.
* Default: 7200 (12 minutes)

busyKeepAliveIdleTimeout = <integer>
* How long, in seconds, that the Splunkd HTTP server allows a keep-alive
  connection to remain idle while in a busy state before forcibly
  disconnecting it.
* Use caution when configuring this setting as a value that is too large
  can result in file descriptor exhaustion due to idling connections.
* If this number is less than 12, it is set to 12.
* Default: 12

forceHttp10 = auto|never|always
* 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 does this 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 may be buggy.
* Default: "auto"

crossOriginSharingPolicy = <origin_acl> ...
* List of the HTTP Origins for which to return Access-Control-Allow-* (CORS)
  headers.
* These headers tell browsers that web applications are trusted 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
* A single "*" can also be used to match all origins
* No default.

x_frame_options_sameorigin = <boolean>
* Adds a X-Frame-Options header set to "SAMEORIGIN" to every response
  served by splunkd.
* Default: true

allowEmbedTokenAuth = <boolean>
* If set to false, splunkd does not allow any access to artifacts
  that previously had been explicitly shared to anonymous users.
* This effectively disables all use of the "embed" feature.
* Default: true

cliLoginBanner = <string>
* Sets a message which is added to the HTTP reply headers
  of requests for authentication, and to the "server/info" endpoint
* This is printed by the Splunk CLI before it prompts
for authentication credentials. This can be used to print access policy information.
* If this string starts with a '"' character, it is treated as a CSV-style list with each line comprising a line of the message.
For example: "Line 1","Line 2","Line 3"
* No default.

allowBasicAuth = <boolean>
* Allows clients to make authenticated requests to the Splunk server using "HTTP Basic" authentication in addition to the normal "authtoken" system
* This is useful for programmatic access to REST endpoints and for accessing the REST API from a web browser. It is not required for the UI or CLI.
* Default: true

basicAuthRealm = <string>
* When using "HTTP Basic" authentication, the 'realm' is a human-readable string describing the server. Typically, a web browser presents this string as part of its dialog box when asking for the username and password.
* This can be used to display a short message describing the server and/or its access policy.
* Default: "/splunk"

allowCookieAuth = <boolean>
* Allows clients to request an HTTP cookie from the /services/auth/login endpoint which can then be used to authenticate future requests
* Default: true

cookieAuthHttpOnly = <boolean>
* When using cookie based authentication, mark returned cookies with the "HttpOnly" flag to tell the client not to allow javascript code to access its value
* NOTE: has no effect if allowCookieAuth=false
* Default: true

cookieAuthSecure = <boolean>
* When using cookie based authentication, mark returned cookies with the "secure" flag to tell the client never to send it over an unencrypted HTTP channel
* NOTE: has no effect if allowCookieAuth=false OR the splunkd REST interface has SSL disabled
* Default: true

dedicatedIoThreads = <integer>
* If set to zero, HTTP I/O is performed in the same thread that accepted the TCP connection.
* If set to a non-zero value, separate threads are run to handle the HTTP I/O, including SSL encryption.
* Typically this setting does not need to be changed. For most usage scenarios using the same thread offers the best performance.
* Default: 0

replyHeader.<name> = <string>
* Add a static header to all HTTP responses this server generates
* For example, "replyHeader.My-Header = value" causes the response header "My-Header: value" to be included in the reply to every HTTP request to the REST server
Splunkd HTTPServer listener configuration

[[httpServerListener:<ip:><port>]]
* Enable the splunkd REST HTTP server to listen on an additional port number specified by <port>. If a non-empty <ip> is included (for example: "[httpServerListener:127.0.0.1:8090]") the listening port is bound only to a specific interface.
* Multiple "httpServerListener" stanzas can be specified to listen on more ports.
* Normally, splunkd listens only on the single REST port specified in the web.conf "mgmtHostPort" setting, and none of these stanzas need to be present. Add these stanzas only if you want the REST HTTP server to listen to more than one port.

ssl = <boolean>
* Toggle whether this listening ip:port uses SSL or not.
* If the main REST port is SSL (the "enableSplunkdSSL" setting in this file's [sslConfig] stanza) and this stanza is set to "ssl=false" then clients on the local machine such as the CLI may connect to this port.
* Default: true

listenOnIPv6 = no|yes|only
* Toggle whether this listening ip:port listens on IPv4, IPv6, or both.
* If not present, the setting in the [general] stanza is used

acceptFrom = <network_acl> ...
* Lists a set of networks or addresses from which to accept connections.
* 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.
* Default: The setting in the [httpServer] stanza

Static file handler MIME-type map

[mimetype-extension-map]
* Map filename extensions to MIME type for files served from the static file handler under this stanza name.

<file-extension> = <MIME-type>
* Instructs the HTTP static file server to mark any files ending in 'file-extension' with a header of 'Content-Type: <MIME-type>'
* Default:

  [mimetype-extension-map]
gif = image/gif
htm = text/html
jpg = image/jpeg
Log rotation of splunkd_stderr.log & splunkd_stdout.log

These stanzas apply only on UNIX. splunkd on Windows has no stdout.log or stderr.log files.

[stderr_log_rotation]
* Controls the data retention of the file containing all messages written to
  splunkd's stderr file descriptor (fd 2).
* Typically this is extremely small, or mostly errors and warnings from
  linked libraries.
maxFileSize = <bytes>
  * When splunkd_stderr.log grows larger than this value, it is rotated.
  * maxFileSize is expressed in bytes.
  * You might want to increase this if you are working on a problem
    that involves large amounts of output to the splunkd_stderr.log file.
  * You might want to reduce this to allocate less storage to this log category.
  * Default: 10000000 (10 si-megabytes)
BackupIndex = <non-negative integer>
  * How many rolled copies to keep.
  * For example, if this setting is 2, the splunkd_stderr.log.1 and
    splunkd_stderr.log.2 file might exist. Further rolls delete the
    current splunkd_stderr.log.2 file.
  * You might want to increase this value if you are working on a problem
    that involves large amounts of output to the splunkd_stderr.log files.
  * You might want to reduce this to allocate less storage to this log category.
  * Default: 2
checkFrequency = <seconds>
  * How often, in seconds, to check the size of splunkd_stderr.log
  * Larger values may result in larger rolled file sizes but take less resources.
  * Smaller values may take more resources but more accurately constrain the
    file size.
  * Default: 10

[stdout_log_rotation]
* Controls the data retention of the file containing all messages written to
  splunkd's stdout file descriptor (fd 1).
* Almost always, there is nothing in this file.
* This stanza can have the same settings as the [stderr_log_rotation]
  stanza with the same defaults. See above for definitions.
maxFileSize = <bytes>
BackupIndex = <non-negative integer>
checkFrequency = <seconds>
Remote applications configuration (e.g. SplunkBase)

[applicationsManagement]
* Set remote applications settings for Splunk 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 uses the default value.

allowInternetAccess = <boolean>
* Allow Splunk to access the remote applications repository.

url = <URL>
* Applications repository.
* Default: https://apps.splunk.com/api/apps

loginUrl = <URL>
* Applications repository login.
* Default: https://apps.splunk.com/api/account:login/

detailsUrl = <URL>
* Base URL for application information, keyed off of app ID.
* Default: https://apps.splunk.com/apps/id

useragent = <splunk-version>-<splunk-build-num>-<platform>
* User-agent string to use when contacting applications repository.
* <platform> includes information like operating system and CPU architecture.

updateHost = <URL>
* Host section of URL to check for app updates, e.g. https://apps.splunk.com

updatePath = <URL>
* Path section of URL to check for app updates
  For example: /api/apps:resolve/checkforupgrade

updateTimeout = <time range string>
* The minimum amount of time Splunk software waits between checks for app updates.
* Examples include '24h' (24 hours), '3d' (3 days), '7200s' (7200 seconds, or two hours)
* Default: 24h

sslVersions = <versions_list>
* Comma-separated list of SSL versions to connect to 'url' (https://apps.splunk.com).
* The versions available are "sslv2", "tlsv1.0", "tlsv1.1", and "tlsv1.2".
* The special version "*" selects all supported versions. The version "tlsv1.0" selects all versions tlsv1.0 or newer.
* If a version is prefixed with "-" it is removed from the list.
* SSLv2 is always disabled; "-sslv2" is accepted in the version list but does nothing.
* When configured in FIPS mode, sslv3 is always disabled regardless of this configuration.
* Default: The default can vary (See the 'sslVersions' setting in the $SPLUNK_HOME/etc/system/default/server.conf file for the current default)

sslVerifyServerCert = <boolean>
* If this is set to true, Splunk verifies that the remote server (specified in 'url') being connected to is a valid one (authenticated).
Both the common name and the alternate name of the server are then checked for a match if they are specified in 'sslCommonNameToCheck' and 'sslAltNameToCheck'. A certificate is considered verified if either is matched.
* Default: true

caCertFile = <path>
* Full path to a CA (Certificate Authority) certificate(s) PEM format file.
* The <path> must refer to a PEM format file containing one or more root CA certificates concatenated together.
* Used only if 'sslRootCAPath' is not set.
* Used for validating SSL certificate from https://apps.splunk.com/

sslCommonNameToCheck = <commonName1>, <commonName2>, ...
* If this value is set, and 'sslVerifyServerCert' is set to true, splunkd checks the common name(s) of the certificate presented by the remote server (specified in 'url') against this list of common names.
* Default: apps.splunk.com

sslCommonNameList = <commonName1>, <commonName2>, ...
* DEPRECATED. Use the 'sslCommonNameToCheck' setting instead.

sslAltNameToCheck = <alternateName1>, <alternateName2>, ...
* If this value is set, and 'sslVerifyServerCert' is set to true, splunkd checks the alternate name(s) of the certificate presented by the remote server (specified in 'url') against this list of subject alternate names.
* Default: splunkbase.splunk.com, apps.splunk.com

cipherSuite = <cipher suite string>
* Uses the specified cipher string for making outbound HTTPS connection.
* The default can vary. See the 'cipherSuite' setting in the $SPLUNK_HOME/etc/system/default/server.conf file for the current default.

ecdhCurves = <comma separated list of ec curves>
* ECDH curves to use for ECDH key negotiation.
* The curves should be specified in the order of preference.
* The client sends these curves as a part of Client Hello.
* Splunk software only supports named curves specified by their SHORT names.
* The list of valid named curves by their short/long names can be obtained by executing this command:
  $SPLUNK_HOME/bin/splunk cmd openssl ecparam -list_curves
* e.g. ecdhCurves = prime256v1,secp384r1,secp521r1
* Default: The default can vary (See the 'ecdhCurves' setting in the $SPLUNK_HOME/etc/system/default/server.conf file for the current default)

Misc. configuration

[scripts]

initialNumberOfScriptProcesses = <num>
* The number of pre-forked script processes that are launched when the system comes up. These scripts are reused when script REST endpoints and search scripts are executed.
* The idea is to eliminate the performance overhead of launching the script interpreter every time it is invoked. These processes are put in a pool.
* If the pool is completely busy when a script gets invoked, a new processes
is fired up to handle the new invocation - but it disappears when that
invocation is finished.

**Disk usage settings (for the indexer, not for Splunk log files)**

```plaintext
[diskUsage]
minFreeSpace = <num>|<percentage>
* Minimum free space for a partition.
* Specified as an integer that represents a size in binary
  megabytes (ie MiB) or as a percentage, written as a decimal
  between 0 and 100 followed by a '%' sign, for example "10%" or "10.5%"
* If specified as a percentage, this is taken to be a percentage of
  the size of the partition. Therefore, the absolute free space required
  varies for each partition depending on the size of that partition.
* Specifies a safe amount of space that must exist for splunkd to continue
  operating.
* Note that this affects search and indexing
* For search:
  * Before attempting to launch a search, Splunk software requires this
    amount of free space on the filesystem where the dispatch directory
    is stored, $SPLUNK_HOME/var/run/splunk/dispatch
  * Applied similarly to the search quota values in authorize.conf and
    limits.conf.
* For indexing:
  * Periodically, the indexer checks space on all partitions
    that contain splunk indexes as specified by indexes.conf. Indexing
    is paused and a UI banner + splunkd warning posted to indicate
    need to clear more disk space.
* Default: 5000 (approx 5GB)

pollingFrequency = <num>
* Specifies that after every 'pollingFrequency' events are indexed,
  the disk usage is checked.
* Default: 100000

pollingTimerFrequency = <num>
* Minimum time, in seconds, between two disk usage checks.
* Default: 10

**Queue settings**

```plaintext
[queue]
maxSize = [<integer>|<integer>[KB|MB|GB]]
* Specifies default capacity of a queue.
* If specified as a lone integer (for example, maxSize=1000), maxSize
  indicates the maximum number of events allowed in the queue.
* If specified as an integer followed by KB, MB, or GB (for example,
  maxSize=100MB), it indicates the maximum RAM allocated for queue.
* Default: 500KB

cntr_1_lookback_time = [<integer>[s|m]]
* The lookback counters are used to track the size and count (number of
elements in the queue) variation of the queues using an exponentially moving weighted average technique. Both size and count variation has 3 sets of counters each. The set of 3 counters is provided to be able to track short, medium and long term history of size/count variation. The user can customize the value of these counters or lookback time.
* Specifies how far into history should the size/count variation be tracked for counter 1.
* It must be an integer followed by [s|m] which stands for seconds and minutes respectively.
* Default: 60s

$cntr_2\_lookback\_time = [<integer>[s|m]]$
* See above for explanation and usage of the lookback counter.
* Specifies how far into history should the size/count variation be tracked for counter 2.
* Default: 600s (10 minutes)

$cntr_3\_lookback\_time = [<integer>[s|m]]$
* See above for explanation and usage of the lookback counter.
* Specifies how far into history should the size/count variation be tracked for counter 3.
* Default: 900s (15 minutes)

$sampling\_interval = [<integer>[s|m]]$
* The lookback counters described above collects the size and count measurements for the queues. This specifies at what interval the measurement collection happens. Note that for a particular queue all the counters sampling interval is same.
* It needs to be specified via an integer followed by [s|m] which stands for seconds and minutes respectively.
* Default: 1s

[queue=<queueName>]

$maxSize = [<integer]|[integer][KB|MB|GB]]$
* Specifies the capacity of a queue. It overrides the default capacity specified in the [queue] stanza.
* If specified as a lone integer (for example, $maxSize=1000$), $maxSize$ indicates the maximum number of events allowed in the queue.
* If specified as an integer followed by KB, MB, or GB (for example, $maxSize=100MB$), it indicates the maximum RAM allocated for queue.
* Default: The default is inherited from the 'maxSize' value specified in the [queue] stanza.

$cntr_1\_lookback\_time = [<integer>[s|m]]$
* Same explanation as mentioned in the [queue] stanza.
* Specifies the lookback time for the specific queue for counter 1.
* Default: The default value is inherited from the 'cntr_1\_lookback\_time' value that is specified in the [queue] stanza.

$cntr_2\_lookback\_time = [<integer>[s|m]]$
* Specifies the lookback time for the specific queue for counter 2.
* Default: The default value is inherited from the 'cntr_2\_lookback\_time' value that is specified in the [queue] stanza.

$cntr_3\_lookback\_time = [<integer>[s|m]]$
* Specifies the lookback time for the specific queue for counter 3.
* Default: The default value is inherited from the 'cntr_3\_lookback\_time' value that is specified in the [queue] stanza.

$sampling\_interval = [<integer>[s|m]]$
* Specifies the sampling interval for the specific queue.
* Default: The default value is inherited from the 'sampling_interval' value specified in the [queue] stanza.

**PubSub server settings for the http endpoint.**

[pubsubsvr-http]
disabled = <boolean>
* If disabled, then http endpoint is not registered. Set this value to 'false' to expose PubSub server on http.
* Default: true

stateIntervalInSecs = <seconds>
* The number of seconds before a connection is flushed due to inactivity. The connection is not closed, only messages for that connection are flushed.
* Default: 300 (5 minutes)

**General file input settings. **NOT SUPPORTED**

[fileInput]
outputQueue = <queue name>
* REMOVED. Historically this allowed the user to set the target queue for the file-input (tailing) processor, but there was no valid reason to modify this.
* This setting is now removed, and has no effect.
* Tailing always uses the parsingQueue.

**Settings controlling the behavior of 'splunk diag', the diagnostic tool**

[diag]
These settings provide defaults for invocations of the splunk diag command. Generally these can be further modified by command line flags to the diag command.

EXCLUDE-<class> = <glob expression>
* Specifies a glob / shell pattern to be excluded from diags generated on this Splunk instance.
* Example: */etc/secret_app/local/*.conf
* Further excludes can be added at the splunk diag command line, but there is no facility to disable configuration-based excludes at the command line.
* There is one exclude by default, for the splunk.secret file.

the following commands can be overridden entirely by their command-line equivalents.

components = <comma separated list>
* Specifies which components of the diag should be gathered.
* This allows the disabling and enabling, categorically, of entire portions of diag functionality.
* All of these components are further subject to the exclude feature (see above), and component-specific filters (see below).
* Currently, with no configuration, all components except "rest" are enabled by default.
* Available components are:
  * index_files : Files from the index that indicate their health (Hosts|Sources|Sourcetypes.data and bucketManifests). User data is not collected.
  * index_listing : Directory listings of the index contents are gathered, in order to see filenames, directory names, sizes, timestamps and the like.
  * etc : The entire contents of the $SPLUNK_HOME/etc directory. In other words, the configuration files.
  * log : The contents of $SPLUNK_HOME/var/log/...
  * pool : If search head pooling is enabled, the contents of the pool dir.
  * dispatch : Search artifacts, without the actual results, In other words var/run/splunk/dispatch, but not the results or events files
  * searchpeers : Directory listings of knowledge bundles replicated for distributed search In other words: $SPLUNK_HOME/var/run/searchpeers
  * consensus : Consensus protocol files produced by search head clustering In other words: $SPLUNK_HOME/var/run/splunk/_raft
  * conf_replication_summary : Directory listing of configuration replication summaries produced by search head clustering In other words: $SPLUNK_HOME/var/run/splunk/snapshot
  * rest : The contents of a variety of splunkd endpoints
    * Includes server status messages (system banners), licenser banners, configured monitor inputs & tailing file status (progress reading input files).
    * On cluster masters, also gathers master info, fixups, current peer list, clustered index info, current generation, & buckets in bad stats
    * On cluster slaves, also gathers local buckets & local slave info, and the master information remotely from the configured master.
  * kvstore : Directory listings of the KV Store data directory contents are gathered, in order to see filenames, directory names, sizes, and timestamps.
  * file_validate : Produce list of files that were in the install media which have been changed. Generally this should be an empty list.
* The special value "all" is also supported, enabling everything explicitly.
* Further controlling the components from the command line:
  * The switch --collect replaces this list entirely.
    * Example: --collect log,etc
      This would set the components to log and etc only, regardless of config
  * The switch --enable adds a specific component to this list.
    * Example: --enable pool
      This would ensure that pool data is collected, regardless of config
  * The switch --disable removes a specific component from this list.
    * Example: --disable pool
      This would ensure that pool data is "NOT" collected, regardless of config
* Default: To collect all components, except "rest".

Data filters; these further refine what is collected most of the existing ones are designed to limit the size and collection
NOTE: Most values here use underscores '_' while the command line uses hyphens '-'

all_dumps = <boolean>
* This setting currently is irrelevant on UNIX platforms.
* Affects the 'log' component of diag. (dumps are written to the log dir on Windows)
* Can be overridden with the --all-dumps command line flag.
* Normally, Splunk diag gathers only three .DMP (crash dump) files on Windows to limit diag size.
* If this is set to true, splunk diag collects *all* .DMP files from the log directory.
* No default. (false equivalent).

index_files = [full|manifests]
* Selects a detail level for the 'index_files' component.
* Can be overridden with the --index-files command line flag.
* If set to 'manifests', limits the index file-content collection to just .bucketManifest files which give some information about the general state of buckets in an index.
* If set to 'full', adds the collection of Hosts.data, Sources.data, and Sourcetypes.data which indicate the breakdown of count of items by those categories per-bucket, and the timespans of those category entries
  * 'full' can take quite some time on very large index sizes, especially when slower remote storage is involved.
* Default: manifests

index_listing = [full|light]
* Selects a detail level for the 'index_listing' component.
* Can be overridden with the --index-listing command line flag.
* 'light' gets directory listings (ls, or dir) of the hot/warm and cold container directory locations of the indexes, as well as listings of each hot bucket.
* 'full' gets a recursive directory listing of all the contents of every index location, which should mean all contents of all buckets.
* 'full' may take significant time as well with very large bucket counts, especially on slower storage.
* Default: light

etc_filesize_limit = <non-negative integer in kilobytes>
* This filters the 'etc' component
* Can be overridden with the --etc-filesize-limit command line flag
* This value is specified in kilobytes.
  * Example: 2000 - this would be approximately 2MB.
* Files in the $SPLUNK_HOME/etc directory which are larger than this limit is not collected in the diag.
* Diag produces a message stating that a file has been skipped for size to the console. (In practice, large files have been found to oftentimes be a surprise to the administrator, and indicate problems).
* If desired, this filter may be entirely disabled by setting the value to 0.
* Currently, as a special exception, the file $SPLUNK_HOME/etc/system/replication/ops.json is permitted to be 10x the size of this limit.
* Default: 10000 (10MB)

log_age = <non-negative integer in days>
* This filters the 'log' component
* Can be overridden with the --log-age command line flag
* This value is specified in days
* Example: 75 - this would be 75 days, or about 2.5 months.
* If desired, this filter may be entirely disabled by setting the value to 0.
* The idea of this default filter is that data older than this is rarely helpful in troubleshooting cases in any event.
* Default: 60 (or approximately 2 months)

upload_proto_host_port = <protocol://host:port>|disabled
* URI base to use for uploading files/diags to Splunk support.
* If set to disabled (override in a local/server.conf file), effectively disables diag upload functionality for this Splunk install.
* Modification may theoretically permit operations with some forms of proxies, but diag is not specifically designed for such, and support of proxy configurations that do not currently work is considered an Enhancement Request.
* The communication path with api.splunk.com is over a simple but not documented protocol. If for some reason you wish to accept diag uploads into your own systems, it probably is simpler to run diag and then upload via your own means independently. However if you have business reasons that you want this built-in, get in touch.
* Uploading to unencrypted http definitely not recommended.
* Default: https://api.splunk.com

SEARCHFILTERSIMPLE-<class> = regex
SEARCHFILTERLUHN-<class> = regex
* Redacts strings from ad-hoc searches logged in the audit.log and remote_searches.log files.
* Substrings which match these regexes *inside* a search string in one of those two files is replaced by sequences of the character X, as in XXXXXXXX.
* Substrings which match a SEARCHFILTERLUHN regex has the contained numbers further tested against the luhn algorithm, used for data integrity in mostly financial circles, such as credit card numbers. This permits more accurate identification of that type of data, relying less heavily on regex precision. See the Wikipedia article on the "Luhn algorithm" for additional information.
* Search string filtering is entirely disabled if --no-filter-searchstrings is used on the command line.
* NOTE: That matching regexes must take care to match only the bytes of the term. Each match "consumes" a portion of the search string, so matches that extend beyond the term (for example, to adjacent whitespace) could prevent subsequent matches, and/or redact data needed for troubleshooting.
* Please use a name hinting at the purpose of the filter in the <class> component of the setting name, and consider an additional explicative comment, even for custom local settings. This might skip inquiries from support.

Application License manager settings for configuring app license checking

[applicense]
appLicenseHostPort = <IP:port>
* Specifies the location of the IP address or DNS name and port of the app license server.

appLicenseServerPath = <path>
* Specifies the path portion of the URI of the app license server.

cacertFile = <path>
* Full path to a CA (Certificate Authority) certificate(s) PEM format file.
* NOTE: Splunk plans to submit Splunk Enterprise for Common Criteria evaluation. Splunk does not support using the product in Common...
Criteria mode until it has been certified by NIAP. See the "Securing Splunk Enterprise" manual for information on the status of Common Criteria certification.

* Default: $SPLUNK_HOME/etc/auth/applicenseCA.pem

sslVersions = <versions_list>
* Comma-separated list of SSL versions to support.
* The special version "*" selects all supported versions. The version "tls" selects all versions tls1.0 or newer.
* If a version is prefixed with "-" it is removed from the list.
* SSLv2 is always disabled; "-ssl2" is accepted in the version list but does nothing.
* When configured in FIPS mode, ssl3 is always disabled regardless of this configuration.
* Default: The default can vary (See the 'sslVersions' setting in the $SPLUNK_HOME/etc/system/default/server.conf file for the current default)

cipherSuite = <cipher suite string>
* If set, uses the specified cipher string for the SSL connection.
* Default: The default can vary (See the 'cipherSuite' setting in the $SPLUNK_HOME/etc/system/default/server.conf file for the current default)

sslVerifyServerCert = <boolean>
* If this is set to true, Splunk Enterprise verifies that the remote server (specified in 'url') being connected to is a valid one (authenticated).
* Both the common name and the alternate name of the server are then checked for a match if they are specified in 'sslCommonNameToCheck' and 'sslAltNameToCheck'.
* A certificate is considered verified if either is matched.
* Default: true

sslCommonNameToCheck = <commonName1>, <commonName2>, ...
* If this value is set, and 'sslVerifyServerCert' is set to true, splunkd limits most outbound HTTPS connections to hosts which use a cert with one of the listed common names.
* Default: Some common name checking

sslAltNameToCheck = <alternateName1>, <alternateName2>, ...
* If this value is set, and 'sslVerifyServerCert' is set to true, splunkd is also willing to 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.
* Items in this list are never validated against the SSL Common Name.
* Default: Some alternate name checking

disabled = <boolean>
* Select true to disable this feature or false to enable this feature. App licensing is experimental, so it is disabled by default.
* Default: true
License manager settings for configuring the license pool(s)

[license]
master_uri = [self]<uri>
* An example of <uri>: <scheme>://<hostname>:<port>

active_group = Enterprise|Trial|Forwarder|Free
* These timeouts only matter if you have a master_uri set to remote master connection_timeout = 30
* Maximum time, in seconds, to wait before connection to master times out.

send_timeout = <integer>
* Maximum time, in seconds, to wait before sending data to master times out
* Default: 30

receive_timeout = <integer>
* Maximum time, in seconds, to wait before receiving data from master times out
* Default: 30

squash_threshold = <positive integer>
* Advanced setting. Periodically the indexer must report to license manager the data indexed broken down by source, sourcetype, host, and index. If the number of distinct (source, sourcetype, host, index) tuples grows over the 'squash_threshold', the (host, source) values are squashed and only a breakdown by (sourcetype, index) is reported. This is to prevent explosions in memory + license_usage.log lines.
* Set this only after consulting a Splunk Support engineer.
* This needs to be set on license slaves as well as license master.
* Default: 2000

report_interval = <nonnegative integer>[s|m|h]
* Selects a time period for reporting in license usage to the license master.
* This value is intended for very large deployments (hundreds of indexers) where a large number of indexers may overwhelm the license server.
* The maximum permitted interval is 1 hour.
* The minimum permitted interval is 1 minute.
* Can be expressed as a positive number of seconds, minutes or hours.
* If no time unit is provided, seconds is assumed.
* Default: 1m

license_warnings_update_interval = <nonnegative integer>
* Specifies a time period, in seconds, for license master to update license warnings in Splunk Web bulletin messages.
* License master checks at every second the last time it updated the warnings, and updates if this time period has elapsed.
* Increase this value for very large deployments that contain very large number of source types.
* The minimum permitted interval is 10.
* The maximum permitted interval is 3600, equivalent to 1 hour.
* If set to special value of 0, the license master automatically tunes this setting to accommodate the size of the deployment.
* Default: 0

strict_pool_quota = <boolean>
* Toggles strict pool quota enforcement
* If set to true, members of pools receive warnings for a given day if usage exceeds pool size regardless of whether overall stack quota was
exceeded
* If set to false, members of pool only receive warnings if both pool usage exceeds pool size AND overall stack usage exceeds stack size
* Default: true

pool_suggestion = <string>
* Suggest a pool to the master for this slave.
* The master uses this suggestion if the master doesn't have an explicit rule mapping the slave to a given pool (i.e., no slave list for the relevant license stack contains this slave explicitly)
* If the pool name doesn't match any existing pool, it is ignored, no error is generated
* This setting is intended to give an alternative management option for pool/slave mappings. When onboarding an indexer, it may be easier to manage the mapping on the indexer itself via this setting rather than having to update server.conf on master for every addition of new indexer
* NOTE: If you have multiple stacks and a slave maps to multiple pools, this feature is limited in only allowing a suggestion of a single pool; This is not a common scenario however.
* No default. (which means this feature is disabled)

[lmpool:auto_generated_pool_forwarder]
* This is the auto generated pool for the forwarder stack
description = <textual description of this license pool>
quota = MAX|<maximum amount allowed by this license>
* MAX indicates the total capacity of the license. You may have only 1 pool with MAX size in a stack
* The quota can also be specified as a specific size eg. 20MB, 1GB etc

slaves = *|<slave list>
* An asterisk(*) indicates that any slave can connect to this pool
* You can also specify a comma separated slave guid list

stack_id = forwarder
* The stack to which this pool belongs.

[lmpool:auto_generated_pool_free]
* This is the auto generated pool for the free stack
* Field descriptions are the same as that for the 'lmpool:auto_generated_pool_forwarder' setting.

[lmpool:auto_generated_pool_enterprise]
* This is the auto generated pool for the enterprise stack
* Field descriptions are the same as that for the 'lmpool:auto_generated_pool_forwarder' setting.

[lmpool:auto_generated_pool_fixed-sourcetype_<sha256 hash of srctypes>]
* This is the auto generated pool for the enterprise fixed source type stack
* Field descriptions are the same as that for the 'lmpool:auto_generated_pool_forwarder' setting.

[lmpool:auto_generated_pool_download_trial]
* This is the auto generated pool for the download trial stack
* Field descriptions are the same as that for the "lmpool:auto_generated_pool_forwarder"

[pooling]
state = [enabled|disabled]
* UNSUPPORTED: This setting is no longer supported
storage = <path to shared storage>
* UNSUPPORTED: This setting is no longer supported

app_update_triggers = true|false|silent
* UNSUPPORTED: This setting is no longer supported

lock.timeout = <time range string>
* UNSUPPORTED: This setting is no longer supported

lock.logging = <boolean>
* UNSUPPORTED: This setting is no longer supported

poll.interval.rebuild = <time range string>
* UNSUPPORTED: This setting is no longer supported

poll.interval.check = <time range string>
* UNSUPPORTED: This setting is no longer supported

poll.blacklist.<name> = <regex>
* UNSUPPORTED: This setting is no longer supported

High availability clustering configuration

[clustering]

mode = [master|slave|searchhead|disabled]
* Sets operational mode for this cluster node.
* Only one master may exist per cluster.
* Default: disabled

master_uri = [<uri> | clustermaster:stanzaName1, clustermaster:stanzaName2]
* Only valid for 'mode=slave' or 'mode=searchhead'.
* The URI of the cluster master that this slave or search head
  should connect to.
* An example of <uri>: <scheme>://<hostname>:<port>
* Only for 'mode=searchhead' - If the search head is a part of multiple
  clusters, the master URIs can be specified by a comma separated list.

advertised_disk_capacity = <integer>
* Percentage to use when advertising disk capacity to the cluster master.
  This is useful for modifying weighted load balancing in indexer discovery.
* For example, if you set this attribute to 50 for an indexer with a
  500GB disk, the indexer advertises its disk size as 250GB, not 500GB.
* Acceptable value range is 10 to 100.
* Default: 100

pass4SymmKey = <password>
* Secret shared among the nodes in the cluster to prevent any
  arbitrary node from connecting to the cluster. If a slave or
  search head is not configured with the same secret as the master,
  it is not able to communicate with the master.
* If it is not set in the [clustering] stanza, the key
  is looked in the [general] stanza
* Unencrypted passwords must not begin with "$1$", as this is used by
  Splunk software to determine if the password is already encrypted.
* No default.
service_interval = <zero or positive integer>
* Only valid when "mode=master".
* Specifies, in seconds, how often the master runs its service loop. In its service loop, the master checks the state of the peers and the buckets in the cluster and also schedules corrective action, if possible, for buckets that are not in compliance with replication policies.
* A special default value of 0 indicates an auto mode where the service interval for the next service call is determined by the time taken by previous call.
* Service interval is bounded by the values 1 and the 'max_auto_service_interval' setting.
  If previous service call takes more than 'max_auto_service_interval' seconds, next service interval is set to 'max_auto_service_interval' seconds.
* Default: 0

defered_cluster_status_update = <boolean>
* Only valid when "mode=master"
* If set to true (default), SF/RF met (complete cluster state) checks are performed lazily for optimal performance, only when CM is busy with cluster maintenance operations (e.g peer addition, fix ups, data rebalance).
* If set to false, SF/RF met checks are performed relatively more aggressively to improve accuracy, increasing CM overhead and slowing down cluster maintenance operations (e.g peer addition, fix ups, data rebalance).
* Note: It is recommended to be set as true with high number of indexers and buckets.
* Default: true

max_fixup_time_ms = <zero or positive integer>
* Only valid for 'mode=master'.
* Specifies, in milliseconds, how long each fixup level runs before short circuiting to continue to the next fixup level. This introduces an upper bound on each service level, and likewise introduces an upper bound on the full service() call.
* This setting is useful for larger clusters that have lots of buckets, where service() calls can consume a significant amount of time blocking other operations.
* 0 denotes that there is no max fixup timer.
* Default: 5000

cxn_timeout = <integer>
* Lowlevel timeout, in seconds, for establishing connection between cluster nodes.
* Default: 60

send_timeout = <integer>
* Lowlevel timeout, in seconds, for sending data between cluster nodes.
* Default: 60

rcv_timeout = <integer>
* Lowlevel timeout, in seconds, for receiving data between cluster nodes.
* Default: 60

rep_cxn_timeout = <integer>
* Lowlevel timeout, in seconds, for establishing connection for replicating data.
* Default: 5

rep_send_timeout = <integer>
* Lowlevel timeout, in seconds, for sending replication slice data between cluster nodes.
* This is a soft timeout. When this timeout is triggered on source peer, it tries to determine if target is still alive. If it is still alive, it reset the timeout for another 'rep_send_timeout interval' and continues. If target has failed or cumulative timeout has exceeded the 'rep_max_send_timeout', replication fails.
* Default: 5

rep_rcv_timeout = <integer>
* Lowlevel timeout, in seconds, for receiving acknowledgment data from peers.
* This is a soft timeout. When this timeout is triggered on source peer, it tries to determine if target is still alive. If it is still alive, it reset the timeout for another 'rep_send_timeout interval and continues.
* If target has failed or cumulative timeout has exceeded 'rep_max_rcv_timeout', replication fails.
* Default: 10

search_files_retry_timeout = <integer>
* Timeout, in seconds, after which request for search files from a peer is aborted.
* To make a bucket searchable, search specific files are copied from another source peer with search files. If search files on source peers are undergoing changes, it asks requesting peer to retry after some time. If cumulative retry period exceeds the specified timeout, the requesting peer aborts the request and requests search files from another peer in the cluster that may have search files.
* Default: 600 (10 minutes)

re_add_on_bucket_request_error = <boolean>
* Valid only for 'mode=slave'.
* If set to true, slave re-adds itself to the cluster master if cluster master returns an error on any bucket request. On re-add, slave updates the master with the latest state of all its buckets.
* If set to false, slave doesn't re-add itself to the cluster master. Instead, it updates the master with those buckets that master returned an error.
* Default: false

decommission_search_jobs_waitsecs = <integer>
* Valid only for mode=slave
* Determines maximum time, in seconds, that a peer node waits for search jobs to finish before it transitions to the down (or) GracefulShutdown state, in response to the 'splunk offline' (or) 'splunk offline --enforce-counts' command.
* Default: 180 (3 minutes)

decommission_node_force_timeout = <seconds>
* Valid only for mode=slave and during node offline operation
* The maximum time, in seconds, that a peer node waits for searchable copy reallocation jobs to finish before it transitions to the down (or) GracefulShutdown state.
* This period begins after the peer node receives a 'splunk offline' command or its '/cluster/slave/control/control/decommission' REST endpoint is accessed.
* This attribute is not applicable to the "--enforce-counts" version of the "splunk offline" command
* Default: 300 seconds

decommission_force_finish_idle_time = <zero or positive integer>
* Valid only for mode=master.
* Time in minutes the master waits before forcibly finishing the decommissioning of a peer when there is no progress in the associated fixup activity.
* A value of zero (0) means that the master does not forcibly finish decommissioning.
* Default: 0

rolling_restart = restart|shutdown|searchable|searchable_force
* Only valid for 'mode=master'.
* Determines whether indexer peers restart or shutdown during a rolling restart.
* If set to restart, each peer automatically restarts during a rolling restart.
* If set to shutdown, each peer is stopped during a rolling restart, and the customer must manually restart each peer.
* If set to searchable, the cluster attempts a best-effort to maintain a searchable state during the rolling restart by reassigning primaries from peers that are about to restart to other searchable peers, and performing a health check to ensure that a searchable rolling restart is possible.
* If set to searchable_force, the cluster performs a searchable rolling restart, but overrides the health check and enforces 'decommission_force_timeout' and 'restart_inactivity_timeout'.
* If set to searchable or searchable_force, scheduled searches are deferred or run during the rolling restart based on the 'defer_scheduled_searchable_idx' setting in savedsearches.conf.
* Default: restart

site_by_site = <boolean>
* Only valid for mode=master and multisite=true.
* If set to true, the master restarts peers from one site at a time, waiting for all peers from a site to restart before moving on to another site, during a rolling restart.
* If set to false, the master randomly selects peers to restart, from across all sites, during a rolling restart.
* Default: true

decommission_force_timeout = <zero or positive integer>
* Only valid for rolling_restart=searchable_force
* The amount of time, in seconds, the cluster master waits for a peer in primary decommission status to finish primary reassignment and restart, during a searchable rolling restart with timeouts.
* Differs from decommission_force_finish_idle_time in its default value and its presence only during a searchable rolling restart with timeouts.
* If you set this parameter to 0, it is automatically reset to default value.
* Maximum accepted value is 1800 (30 minutes).
* Default: 180 (3 minutes)

restart_inactivity_timeout = <zero or positive integer>
* Only valid for rolling_restart=searchable_force
* The amount of time, in seconds, that the master waits for a peer to restart and rejoin the cluster before it considers the restart a failure and proceeds to restart other peers.
* More specifically, the amount of time that the master waits for a peer in the 'Down' status to transition to 'BatchAdding' or 'Up' status.
* A value of zero (0) means that the master waits indefinitely for a peer to restart.
* Default: 600 (10 minutes)

rebalance_pipeline_batch_size = <integer>
* Valid only for 'mode=master'.
* Valid only for 'searchable_rebalance=true'.
* The maximum number of buckets for a batch entering the excess bucket removal phase of the rebalance pipeline.
rebalance_primary_failover_timeout = <zero or positive integer>
* Valid only for 'mode=master'.
* Valid only for 'searchable_rebalance=true'.
* The maximum length of time, in seconds, that the master waits for primacy to be reassigned from the batch of excess buckets to other buckets.
* Default: 75

rebalance_newgen_propagation_timeout = <zero or positive integer>
* Valid only for 'mode=master'.
* Valid only for 'searchable_rebalance=true'.
* The amount of time, in seconds, that the master waits for the search heads to get the newly committed generation after the discarded buckets' primacy has been reassigned.
* Default: 60 (1 minute)

rebalance_search_completion_timeout = <integer>
* Valid only for 'mode=master'.
* Valid only for 'searchable_rebalance=true'.
* The amount of time, in seconds, that the master waits for older generation searches on indexers to complete before removing any excess buckets.
* Default: 180 (3 minute)

searchable_rebalance = <boolean>
* Valid only for 'mode=master'.
* Controls whether searches can continue uninterrupted during data rebalancing.
* Default: false

rep_max_send_timeout = <integer>
* Maximum send timeout, in seconds, for sending replication slice data between cluster nodes.
* On rep_send_timeout source peer determines if total send timeout has exceeded 'rep_max_send_timeout'. If so, replication fails.
* If cumulative 'rep_send_timeout' exceeds 'rep_max_send_timeout', replication fails.
* Default: 180 (3 minutes)

rep_max_rcv_timeout = <integer>
* Maximum cumulative receive timeout, in seconds, for receiving acknowledgment data from peers.
* On 'rep_rcv_timeout' source peer determines if total receive timeout has exceeded 'rep_max_rcv_timeout'. If so, replication fails.
* Default: 180 (3 minutes)

multisite = <boolean>
* Turns on the multisite feature for this master.
* Make sure you set site parameters on the peers when you turn this to true.
* Default: false

replication_factor = <positive integer>
* Only valid for mode=master.
* Determines how many copies of rawdata are created in the cluster.
* Use site_replication_factor instead of this in case 'multisite' is turned on.
* Must be greater than 0.
* Default: 3

site_replication_factor = <comma-separated string>
* Only valid for 'mode=master' and is only used if 'multisite=true'.

Default: 60
* This specifies the per-site replication policy for any given bucket represented as a comma-separated list of per-site entries.
* Currently specified globally and applies to buckets in all indexes.
* Each entry is of the form <site-id>:<positive integer> which represents the number of copies to make in the specified site.
* Valid site-ids include two mandatory keywords and optionally specific site-ids from site1 to site63.
* The mandatory keywords are:
  - origin: Every bucket has a origin site which is the site of the peer that originally created this bucket. The notion of 'origin' makes it possible to specify a policy that spans across multiple sites without having to enumerate it per-site.
  - total: The total number of copies needed for each bucket.
* When a site is the origin, it could potentially match both the origin and a specific site term. In that case, the max of the two is used as the count for that site.
* The total must be greater than or equal to sum of all the other counts (including origin).
* The difference between total and the sum of all the other counts is distributed across the remaining sites.
* Example 1: site_replication_factor = origin:2, total:3
  Given a cluster of 3 sites, all indexing data, every site has 2 copies of every bucket ingested in that site and one rawdata copy is put in one of the other 2 sites.
* Example 2: site_replication_factor = origin:2, site3:1, total:3
  Given a cluster of 3 sites, 2 of them indexing data, every bucket has 2 copies in the origin site and one copy in site3. So site3 has one rawdata copy of buckets ingested in both site1 and site2 and those two sites have 2 copies of their own buckets.
* Default: origin:2, total:3

search_factor = <positive integer>
* Only valid for 'mode=master'.
* Determines how many buckets have index structures pre-built.
* Must be less than or equal to the 'replication_factor' setting and greater than 0.
* Default: 2

site_search_factor = <comma-separated string>
* Only valid for 'mode=master' and is only used if 'multisite=true'.
* This specifies the per-site policy for searchable copies for any given bucket represented as a comma-separated list of per-site entries.
* This is similar to the 'site_replication_factor' setting. Please see that entry for more information on the syntax.
* Default: origin:1, total:2

available_sites = <comma-separated string>
* Only valid for 'mode=master' and is only used if 'multisite=true'.
* This is a comma-separated list of all the sites in the cluster.
* If 'multisite=true' then 'available_sites' must be explicitly set.
* Default: an empty string

forwarder_site_failover = <comma-separated string>
* Only valid for mode=master and is only used if 'multisite=true'.
* This is a comma-separated list of pair of sites, "site1:site2", in the cluster.
* If 'multisite' is turned on 'forwarder_site_failover' must be explicitly set.
* Default: an empty string
site_mappings = <comma-separated string>
* Only valid for mode=master
* When you decommission a site, you must update this attribute so that the origin bucket copies on the decommissioned site are mapped to a remaining active site. This attribute maps decommissioned sites to active sites. The bucket copies for which a decommissioned site is the origin site are then replicated to the active site specified by the mapping.
* Used only if multisite is true and sites have been decommissioned.
* Each comma-separated entry is of the form <decommissioned_site_id>:<active_site_id> or default_mapping:<default_site_id>.
<decommissioned_site_id> is a decommissioned site and <active_site_id> is an existing site, specified in the 'available_sites' setting. For example, if available_sites=site1,site2,site3,site4 and you decommission site2, you can map site2 to a remaining site such as site4, like this: site2:site4.
* If a site used in a mapping is later decommissioned, its previous mappings must be remapped to an available site. For instance, if you have the mapping site1:site2 but site2 is later decommissioned, you can remap both site1 and site2 to an active site3 using the following replacement mappings = site1:site3,site2:site3.
* Optional entry with syntax default_mapping:<default_site_id> represents the default mapping, for cases where an explicit mapping site is not specified. For example: default_mapping:site3 maps any decommissioned site to site3, if they are not otherwise explicitly mapped to a site. There can only be one such entry.
* Example 1: site_mappings = site1:site3,default_mapping:site4. The cluster must include site3 and site4 in available_sites, and site1 must be decommissioned. The origin bucket copies for decommissioned site1 is mapped to site3. Bucket copies for any other decommissioned sites is mapped to site4.
* Example 2: site_mappings = site2:site3 The cluster must include site3 in available_sites, and site2 must be decommissioned. The origin bucket copies for decommissioned site2 is mapped to site3. This cluster has no default.
* Example 3: site_mappings = default_mapping:site5 The above cluster must include site5 in available_sites. The origin bucket copies for any decommissioned sites is mapped onto site5.
* Default: an empty string

constrain_singlesite_buckets = <boolean>
* Only valid for mode=master and is only used if multisite is true.
* Specifies whether the cluster keeps single-site buckets within one site in multisite clustering.
* When this setting is "true", buckets in a single site cluster do not replicate outside of their site. The buckets follow 'replication_factor' 'search_factor' policies rather than 'site_replication_factor' 'site_search_factor' policies. This is to mimic the behavior of single-site clustering.
* When this setting is "false", buckets in non-multisite clusters can replicate across sites, and must meet the specified 'site_replication_factor' and 'site_search_factor' policies.
* Default: true

heartbeat_timeout = <positive integer>
* Only valid for 'mode-master'.
* Specifies, in seconds, when the master considers a slave down. After a slave is down, the master initiates fixup steps to replicate buckets from the dead slave to its peers.
* Default: 60
access_logging_for_heartbeats = <boolean>
* Only valid for 'mode=master'.
* Enables/disables logging to the splunkd_access.log file for peer heartbeats.
* NOTE: you do not have to restart master to set this config parameter.
  Simply run the cli command on master:
  `% splunk edit cluster-config -access_logging_for_heartbeats <<boolean>>`
* Default: false (logging disabled)

restart_timeout = <positive integer>
* Only valid for 'mode=master'.
* This is the amount of time, in seconds, the master waits for a peer to come back when the peer is restarted (to avoid the overhead of trying to fix up the buckets that were on the peer).
* More specifically, the amount of time that the master waits for a peer in the 'Restarting' status to transition to the 'Down' status.
* Note that this only works with the offline command or if the peer is restarted via the UI.
* Default: 60

quiet_period = <positive integer>
* Only valid for 'mode=master'.
* This setting determines the amount of time, in seconds, that the master is quiet upon start-up.
* However, if peers are still registering themselves with the master after the initial quiet_period has elapsed, the master continues to remain quiet until all peers finish registering, up to a total quiet time not to exceed 3x the specified 'quiet_period', including the initial quiet time.
* During the quiet time, the master does not initiate any actions. At the end of this period, the master builds its view of the cluster based on the registered information. It then starts normal operations.
* Default: 60

reporting_delay_period = <positive integer>
* Only valid for 'mode=master'.
* The acceptable amount of delay, in seconds, for reporting both unmet search and unmet replication factors for newly created buckets.
* This setting helps provide more reliable cluster status reporting by limiting updates to the specified granularity.
* Default: 30

generation_poll_interval = <positive integer>
* How often, in seconds, the search head polls the master for generation information.
* This setting is valid only if 'mode=master' or 'mode=searchhead'.
* Default: 5

max_peer_build_load = <integer>
* This is the maximum number of concurrent tasks to make buckets searchable that can be assigned to a peer.
* Default: 2

max_peer_rep_load = <integer>
* This is the maximum number of concurrent non-streaming replications that a peer can take part in as a target.
* Default: 5

max_peer_sum_rep_load = <integer>
* This is the maximum number of concurrent summary replications that a peer can take part in as either a target or source.
max_nonhot_rep_kBps = <integer>
* This is the maximum throughput (kB(Byte)/s) for warm/cold/summary replications on a specific source peer. Similar to forwarder's 'maxKBps' setting in the limits.conf file.
* This setting throttles total bandwidth consumption for all outgoing non-hot replication connections from a given source peer. It does not throttle at the per-replication-connection, per-target level.
* This setting can be updated without restart on the source peers by using the command "splunk edit cluster-config" or by making the corresponding REST call.
* If set to 0, signifies unlimited throughput.
* Default: 0

max_replication_errors = <integer>
* Only valid for 'mode=slave'.
* This is the maximum number of consecutive replication errors (currently only for hot bucket replication) from a source peer to a specific target peer. Until this limit is reached, the source continues to roll hot buckets on streaming failures to this target. After the limit is reached, the source no longer rolls hot buckets if streaming to this specific target fails. This is reset if at least one successful (hot bucket) replication occurs to this target from this source.
* The special value of 0 turns off this safeguard; so the source always rolls hot buckets on streaming error to any target.
* Default: 3

searchable_targets = <boolean>
* Only valid for 'mode=master'.
* Tells the master to make some replication targets searchable even while the replication is going on. This only affects hot bucket replication for now.
* Default: true

searchable_target_sync_timeout = <integer>
* Only valid for 'mode=slave'.
* If a hot bucket replication connection is inactive for this time, in seconds, a searchable target flushes out any pending search related in-memory files.
* Regular syncing - when the data is flowing through regularly and the connection is not inactive - happens at a faster rate (default of 5 secs controlled by streamingTargetTsidxSyncPeriodMsec in indexes.conf).
* The special value of 0 turns off this timeout behavior.
* Default: 60

target_wait_time = <positive integer>
* Only valid for 'mode=master'.
* Specifies the time, in seconds, that the master waits for the target of a replication to register itself before it services the bucket again and potentially schedules another fixup.
* Default: 150 (2 minutes 30 seconds)

summary_wait_time = <positive integer>
* Only valid when 'mode=master' and 'summary_replication=true'.
* Specifies the time, in seconds, that the master waits before scheduling fixups for a newly 'done' summary that transitioned from 'hot_done'. This allows for other copies of the 'hot_done' summary to also make their transition into 'done', avoiding
unnecessary replications.
* Default: 660 (11 minutes)

commit_retry_time = <positive integer>
* Only valid for 'mode=master'.
* Specifies the interval, in seconds, after which, if the last
generation commit failed, the master forces a retry. A retry is usually
automatically kicked off after the appropriate events. This is just
a backup to make sure that the master does retry no matter what.
* Default: 300 (5 minutes)

percent_peers_to_restart = <integer between 0-100>
* Suggested percentage of maximum peers to restart for rolling-restart.
* Actual percentage may vary due to lack of granularity for smaller peer
sets.
* Regardless of setting, a minimum of 1 peer is restarted per round.

max_peers_to_download_bundle = <positive integer>
* Only valid for mode=master
* Maximum no. of peers to simultaneously download the configuration bundle
from the master, in response to the 'splunk apply cluster-bundle' command.
* When a peer finishes the download, the next waiting peer, if any, begins
its download.
* If set to 0, all peers try to download at once.
* Default: 5

auto_rebalance_primaries = <boolean>
* Only valid for 'mode=master'.
* Specifies if the master should automatically rebalance bucket
primaries on certain triggers. Currently the only defined
trigger is when a peer registers with the master. When a peer
registers, the master redistributes the bucket primaries so the
cluster can make use of any copies in the incoming peer.
* Default: true

rebalance_primaries_execution_limit = <non-negative integer>
* DEPRECATED. Use the 'rebalance_primaries_execution_limit_ms' setting instead.

rebalance_primaries_execution_limit_ms = <non-negative integer>
* Only valid for 'mode=master'.
* Specifies, in milliseconds, the maximum period for one execution
of the rebalance primary operation.
* This setting is useful for large clusters with large numbers of
buckets, to prevent the primary rebalance operation from blocking
other operations for significant amounts of time.
* The default value of 0 signifies auto mode. In auto mode, the cluster
master uses the value of the 'service_interval' setting to determine the
maximum time for the operation.
* Default: 0

commit_generation_execution_limit_ms = <non-negative integer>
* Only valid for 'mode=master'.
* Specifies, in milliseconds, the maximum period for one execution
of the committing pending generation.
* This setting is useful for large clusters with large numbers of
buckets, to prevent the commit-generation operation from blocking
other operations for significant amounts of time.
* The default value of 0 signifies auto mode. In auto mode, the cluster
master uses the value of the 'service_interval' setting to determine the
maximum time for the operation.
* If 'service_interval' is auto, the range of this value will be within the
range of 10ms and 25ms.
idle_connections_pool_size = <integer>
* Only valid for 'mode=master'.
* Specifies how many idle http(s) connections that should be kept alive to reuse.
* Reusing connections improves the time it takes to send messages to peers in the cluster.
* -1 corresponds to "auto", letting the master determine the number of connections to keep around based on the number of peers in the cluster.
* Default: -1

use_batch_mask_changes = <boolean>
* Only valid for mode=master
* Specifies if the master should process bucket mask changes in batch or individually one by one.
* Set to false when there are version 6.1 peers in the cluster for backwards compatibility.
* Default: true

service_jobs_msec = <positive integer>
* Only valid for 'mode=master'.
* Max time, in milliseconds, that the cluster master spends in servicing finished jobs for each service call. Increase this if the metrics.log file has very high 'current_size' values.
* Default: 100 (0.1 seconds)

summary_replication = true|false|disabled
* Valid for both 'mode=master' and 'mode=slave'.
* Cluster Master:
  - If set to true, summary replication is enabled.
  - If set to false, summary replication is disabled, but can be enabled at runtime.
  - If set to disabled, summary replication is disabled. Summary replication cannot be enabled at runtime.
* Peers:
  - If set to true or false, there is no effect. The indexer follows whatever setting is on the Cluster Master.
  - If set to disabled, summary replication is disabled. The indexer does no scanning of summaries (increased performance during peers joining the cluster for large clusters).
* Default: false (for both Cluster Master and Peers)

rebalance_threshold = <number between 0.10 and 1.00>
* Only valid for mode=master'.
* During rebalancing buckets amongst the cluster, this threshold is used as a percentage to determine when the cluster is balanced.
* 1.00 is 100% indexers fully balanced.

max_auto_service_interval = <positive integer>
* Only valid for 'mode=master'.
* Only valid when 'service_interval' is in auto mode.
* Indicates the maximum value, in seconds, that service interval is bounded by when the 'service_interval' is in auto mode. If the previous service call took more than 'max_auto_service_interval' seconds, the next service call runs after 'max_auto_service_interval' seconds.
* NOTE: It is highly recommended that you choose a value that is one-half of the smaller of 'heartbeat_timeout' or 'restart_timeout'. For example, the default value of 30 is based on the default value of 60 for both...
'heartbeat_timeout' and 'restart_timeout'.
* Default: 30

buckets_to_summarize = <primaries|primaries_and_hot|all>
* Only valid for 'mode=master'.
* Determines which buckets are sent to ' summarize' searches (searches that
build report acceleration and data models).
* Set to "primaries" to apply only to primary buckets.
* Set to "primaries_and_hot" to also apply it to all hot searchable
buckets.
* Set to "all" to apply the search to all buckets.
* If "summary_replication' is enabled, then 'buckets_to_summarize' defaults
to "primaries_and_hot".
* Do not change this setting without first consulting with Splunk Support.
* Default: primaries

maintenance_mode = <boolean>
* Only valid for 'mode=master'.
* To preserve the maintenance mode setting in case of master
restart, the master automatically updates this setting in the
/etc/system/local/server.conf file whenever the user enables or disables
maintenance mode using CLI or REST.
* NOTE: Do not manually update this setting. Instead use CLI or REST
to enable or disable maintenance mode.

backup_and_restore_primaries_in_maintenance = <boolean>
* Only valid for 'mode=master'.
* Determines whether the master performs a backup/restore of bucket
primary masks during maintenance mode or rolling-restart of cluster peers.
* If set to true, restoration of primaries occurs automatically when the peers
rejoin the cluster after a scheduled restart or upgrade.
* Default: false

max_primary_backups_per_service = <zero or positive integer>
* Only valid for 'mode=master'.
* For use with the 'backup_and_restore_primaries_in_maintenance' setting.
* Determines the number of peers for which the master backs up primary
masks for each service call.
* The special value of 0 causes the master to back up the primary masks for
all peers in a single service call.
* Default: 10

allow_default_empty_p4symmkey = <boolean>
* Only valid for 'mode=master'.
* Affects behavior of master during start-up, if 'pass4SymmKey'resolves
to the null string or the default password ("changeme").
* If set to true, the master posts a warning but still launches.
* If set to false, the master posts a warning and stops.
* Default: true

register_replication_address = <IP address or fully qualified machine/domain name>
* Only valid for 'mode=slave'.
* This is the address on which a slave is available for accepting
replication data. This is useful in the cases where a slave host machine
has multiple interfaces and only one of them can be reached by another
splunkd instance

register_forwarder_address = <IP address or fully qualified machine/domain name>
* Only valid for 'mode=slave'.
* This is the address on which a slave is available for accepting
data from forwarder. This is useful in the cases where a splunk host
machine has multiple interfaces and only one of them can be reached by
another splunkd instance.

register_search_address = <IP address, or fully qualified machine/domain name>
* Only valid for 'mode=slave'.
* This is the address on which a slave is available as search head.
  This is useful in the cases where a splunk host machine has multiple
  interfaces and only one of them can be reached by another splunkd
  instance.

executor_workers = <positive integer>
* Only valid if 'mode=master' or 'mode=slave'.
* Number of threads that can be used by the clustering thread pool.
* A value of 0 defaults to 1.
* Default: 10

local_executor_workers = <positive integer>
* Only valid if 'mode=slave'
* Number of threads that can be used by the local clustering thread pool.
* executor_workers is used mostly for communication between the peer
  and the master. local_executor_workers are used for any jobs that
  must be spawned to take care of housekeeping tasks only related
  to the peer such as a peer synchronizing itself with remote storage.
* A value of 0 defaults to 1.
* Default: 10

manual_detention = on|on_ports_enabled|off
* Only valid for 'mode=slave'.
* Puts this peer node in manual detention.
* Default: off

allowed_hbmiss_count = <non-zero positive integer>
* Only valid for 'mode=slave'.
* Sets the count of number of heartbeat failures before the peer node
  disconnects from the master.
* Default: 3

buckets_per_addpeer = <non-negative integer>
* Only valid for 'mode=slave'.
* Controls the number of buckets for each add peer request.
  When a peer is added or re-added to the cluster, it sends the master
  information for each of its buckets. Depending on the number of buckets,
  this could take a while. For example, a million buckets could require
  more than a minute of the master's processing time. To prevent the master
  from being occupied by this single task too long, you can use this setting to
  split large numbers of buckets into several "batch-add-peer" requests.
  If it is invalid or non-existent, the peer uses the default setting instead.
  If it is set to 0, the peer sends only one request with all buckets
  instead of batches.
* Default: 1000

heartbeat_period = <non-zero positive integer>
* Only valid for 'mode=slave'.
* Controls the frequency the slave attempts to send heartbeats.

remote_storage_upload_timeout = <non-zero positive integer>
* Only valid for 'mode=slave'.
* For a remote storage enabled index, this attribute specifies the interval
  in seconds, after which target peers assume responsibility for
  uploading a bucket to the remote storage, if they do not hear from
  the source peer.
* Default: 60 (1 minute)
report_remote_storage_bucket_upload_to_targets = <boolean>
* Only valid for 'mode=slave'.
* For a remote storage enabled index, this attribute specifies whether the source peer reports the successful bucket upload to target peers. This notification is used by target peers to cancel their upload timers and synchronize their bucket state with the uploaded bucket on remote storage.
* Do not change the value from the default unless instructed by Splunk Support.
* Default: false

remote_storage_retention_period = <non-zero positive integer>
* Only valid for 'mode=master'.
* Controls the length, in seconds, of peer-node retention for buckets in remote storage enabled indexes. When this length is exceeded, the master freezes the buckets on the peer nodes.
* Default: 900 (15 minutes)

recreate_bucket_attempts_from_remote_storage = <positive integer>
* Only valid for 'mode=master'.
* Controls the number of attempts the master makes to recreate the bucket of a remote storage enabled index on a random peer node in these scenarios:
  * Master detects that the bucket is not present on any peers.
  * A peer informs the master about the bucket as part of the re-creation of an index.
  * See recreate_index_attempts_from_remote_storage attribute.
* Re-creation of the bucket involves the following steps:
  1. Master provides a random peer with the bucket ID of the bucket that needs to be recreated.
  2. Peer fetches the metadata of the bucket corresponding to this bucket ID from the remote storage.
  3. Peer creates a bucket with the fetched metadata locally and informs the master that a new bucket has been added.
  4. Master initiates fix-ups to add the bucket on the necessary number of additional peers to match the replication and search factors.
* If set to 0, disables the re-creation of the bucket.
* Default: 10

recreate_bucket_max_per_service = <positive integer>
* Only valid for 'mode=master'.
* Only applies when using remote storage enabled indexes.
* Controls the maximum number of buckets that the cluster can recreate during a service interval.
* Do not change the value from the default unless instructed by Splunk Support.
* If set to 0, recreating buckets will go at full speed.
* Default: 20000

recreate_bucket_fetch_manifest_batch_size = <positive integer>
* Only valid for 'mode=master'.
* Controls the maximum number of bucket IDs for which a slave attempts to initiate a parallel fetch of manifests at a time in the process of recreating buckets that have been requested by the master.
* The master sends this setting to all the slaves that are involved in the process of recreating the buckets.
* Default: 50

recreate_index_attempts_from_remote_storage = <positive integer>
* Only valid for 'mode=master'.
* Controls the number of attempts the master makes to recreate
a remote storage enabled index on a random peer node when the master is informed about the index by a peer.

* Re-creation of an index involves the following steps:
  1. Master pushes a bundle either when it is ready for service or when requested by the user.
  2. Master waits for the bundle to be applied successfully on the peer nodes.
  3. Master requests that a random peer node provide it with the list of newly added remote storage enabled indexes.
  4. Master distributes a subset of indexes from this list to random peer nodes.
  5. Each of those peer nodes fetches the list of bucket IDs for the requested index from the remote storage and provides it to the master.
  6. The master uses the list of bucket IDs to recreate the buckets. See `recreate_bucket_attempts_from_remote_storage`.
* If set to 0, disables the re-creation of the index.
* Default: 10

* `recreate_index_fetch_bucket_batch_size = <positive integer>`
  * Only valid for 'mode=master'.
  * Controls the maximum number of bucket IDs that the master requests a random peer node to fetch from remote storage as part of a single transaction for a remote storage enabled index. The master uses the bucket IDs for re-creation of the index. See the 'recreate_index_attempts_from_remote_storage' setting.
  * Default: 2000

* `use_batch_remote_rep_changes = <boolean>`
  * Only valid for 'mode=master'.
  * Specifies whether the master processes bucket copy changes (to meet replication_factor and search_factor) in batch or individually.
  * This is applicable to buckets belonging to remote storage enabled indexes only.
  * Do not change this setting without consulting with Splunk Support.
  * Default: false

* `buckets_status_notification_batch_size = <positive integer>`
  * Only valid for 'mode=slave'.
  * Controls the number of existing buckets IDs that the slave reports to the master every notify_scan_period seconds. The master then initiates fix-ups for these buckets.
  * CAUTION: Do not modify this setting without guidance from Splunk personnel.
  * Default: 1000

* `notify_scan_period = <non-zero positive integer>`
  * Only valid for 'mode=slave'.
  * Controls the frequency, in seconds, that the indexer handles the following options:
    1. summary_update_batch_size
    2. summary_registration_batch_size
  * CAUTION: Do not modify this setting without guidance from Splunk personnel.
  * Default: 10

* `notify_scan_min_period = <non-zero positive integer>`
  * Only valid for 'mode=slave'.
  * Controls the highest frequency, in milliseconds, that the indexer scans summary folders for summary updates/registrations. The notify_scan_period temporarily becomes notify_scan_min_period when there are more summary
updates/registration events to be processed but has been limited due to either summary_update_batch_size or summary_registration_batch_size.
* CAUTION: Do not modify this setting without guidance from Splunk personnel.
* Default: 10

notify_buckets_period = <non-zero positive integer>
* Only valid for 'mode=slave'.
* Controls the frequency, in milliseconds, that the indexer handles
  buckets_status_notification_batch_size
* CAUTION: Do not modify this setting without guidance from Splunk personnel.
* Default: 10

summary_update_batch_size = <non-zero positive integer>
* Only valid for 'mode=slave'.
* Controls the number of summary updates the indexer sends per batch to
  the master every notify_scan_period.
* CAUTION: Do not modify this setting without guidance from Splunk personnel.
* Default: 10

summary_registration_batch_size = <non-zero positive integer>
* Only valid for 'mode=slave'.
* Controls the number of summaries that get asynchronously registered
  on the indexer and sent as a batch to the master every
  notify_scan_period.
  * Caution: Do not modify this setting without guidance from Splunk personnel.
  * Default: 1000

enableS2SHeartbeat = <boolean>
* Only valid for 'mode=slave'.
* Splunk software monitors each replication connection for
  presence of a heartbeat, and if the heartbeat is not seen for
  's2sHeartbeatTimeout' seconds, it closes the connection.
* Default: true

s2sHeartbeatTimeout = <seconds>
* This specifies the global timeout value, in seconds, for monitoring
  heartbeats on replication connections.
* Splunk software closes a replication connection if heartbeat is not seen
  for 's2sHeartbeatTimeout' seconds.
* Replication source sends heartbeats every 30 seconds.
* Default: 600 (10 minutes)

throwOnBucketBuildReadError = <boolean>
* Valid only for 'mode=slave'.
* If set to true, index clustering slave throws an exception if it
  encounters a journal read error while building the bucket for a new
  searchable copy. It also throws all the search & other files generated
  so far in this particular bucket build.
* If set to false, index clustering slave just logs the error and preserves
  all the search & other files generated so far & finalizes them as it
  cannot proceed further with this bucket.
* Default: false

cluster_label = <string>
* This specifies the label of the indexer cluster

warm_bucket_replication_pre_upload = <boolean>
* Valid only for 'mode=slave'.
* This setting applies to remote storage enabled indexes only.
* If set to true, the target peers replicate all warm bucket contents when necessary for bucket-fixing if the source peer has not yet uploaded the bucket to remote storage.
* If set to false, the target peers never replicate warm bucket contents.
* In either case the target peers replicate metadata only, once the source peer uploads the bucket to remote storage.
* Default: false

[clustermaster:<stanza>]
* Only valid for 'mode=searchhead' when the search head is a part of multiple clusters.

master_uri = <uri>
* Only valid for 'mode=searchhead' when present in this stanza.
* URI of the cluster master that this search head should connect to.

pass4SymmKey = <password>
* Secret shared among the nodes in the cluster to prevent any arbitrary node from connecting to the cluster. If a search head is not configured with the same secret as the master, it not be able to communicate with the master.
* If it is not present here, the key in the clustering stanza is used.
* If it is not present in the clustering stanza, the value in the general stanza is used.
* Unencrypted passwords must not begin with "$1$", as this is used by Splunk software to determine if the password is already encrypted.
* No default.

site = <site-id>
* Specifies the site this search head belongs to for this particular master when multisite is enabled (see below).
* Valid values for site-id include site0 to site63.
* The special value "site0" disables site affinity for a search head in a multisite cluster. It is only valid for a search head.

multisite = <boolean>
* Turns on the multisite feature for this master_uri for the search head.
* Make sure the master has the multisite feature turned on.
* Make sure you specify the site in case this is set to true. If no configuration is found in the [clustermaster] stanza, the search head defaults to any value for 'site' that might be defined in the [general] stanza.
* Default: false

[replication_port://<port>]
Configure Splunk to listen on a given TCP port for replicated data from another cluster member.
If 'mode=slave' is set in the [clustering] stanza at least one 'replication_port' must be configured and not disabled.

disabled = <boolean>
* Set to true to disable this replication port stanza.
* Default: false

listenOnIPv6 = no|yes|only
* Toggle whether this listening port listens on IPv4, IPv6, or both.
* If not present, the setting in the [general] stanza is used.

acceptFrom = <network_acl> ...
* Lists a set of networks or addresses from which to accept connections.
* 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)

[replication_port-ssl://<port>]
* This configuration is same as the [replication_port] stanza above, but
  uses SSL.

   disabled = <boolean>
   * Set to true to disable this replication port stanza.
   * Default: false

   listenOnIPv6 = no|yes|only
   * Toggle whether this listening port listens on IPv4, IPv6, or both.
   * If not present, the setting in the [general] stanza is used.

   acceptFrom = <network_acl> ...
   * This setting is the same as the setting in the [replication_port] stanza.

   serverCert = <path>
   * Full path to file containing private key and server certificate.
   * The <path> must refer to a PEM format file.
   * No default.

   sslPassword = <password>
   * Server certificate password, if any.
   * No default.

   password = <password>
   * DEPRECATED; use 'sslPassword' instead.

   rootCA = <path>
   * DEPRECATED; use '[sslConfig]/sslRootCAPath' instead.
   * Full path to the root CA (Certificate Authority) certificate store.
   * The <path> must refer to a PEM format file containing one or more root CA
     certificates concatenated together.
   * No default.

   cipherSuite = <cipher suite string>
   * If set, uses the specified cipher string for the SSL connection.
   * Must specify 'dhFile' to enable any Diffie-Hellman ciphers.
   * Default: The default can vary (See the cipherSuite setting in
     the $SPLUNK_HOME/etc/system/default/server.conf file for the current default)

   sslVersions = <versions_list>
   * 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 a version is prefixed with "+" it is removed from the list.
   * SSLv2 is always disabled; "+-ssl2" is accepted in the version list but
     does nothing.
   * When configured in FIPS mode, ssl3 is always disabled regardless
     of this configuration.
* Default: The default can vary (See the sslVersions setting in the `$SPLUNK_HOME/etc/system/default/server.conf` file for the current default)

ecdhCurves = <comma separated list of ec curves>
* ECDH curves to use for ECDH key negotiation.
* The curves should be specified in the order of preference.
* The server supports only the curves specified in the list.
* The client sends these curves as a part of Client Hello.
* Splunk software only supports named curves specified by their SHORT names.
* The list of valid named curves by their short/long names can be obtained by executing this command:
  `$SPLUNK_HOME/bin/splunk cmd openssl ecparam -list_curves`
* e.g. ecdhCurves = prime256v1,secp384r1,secp521r1
* Default: The default can vary (See the 'ecdhCurves' setting in the `$SPLUNK_HOME/etc/system/default/server.conf` file for the current default)

dhFile = <path>
* PEM format Diffie-Hellman parameter file name.
* DH group size should be no less than 2048 bits.
* This file is required in order to enable any Diffie-Hellman ciphers.
* No default.

dhfile = <path>
* DEPRECATED; use 'dhFile' instead.

supportSSLV3Only = <boolean>
* DEPRECATED. SSLv2 is now always disabled. The exact set of SSL versions allowed is now configurable by using the 'sslVersions' setting above.

useSSLCompression = <boolean>
* If true, enables SSL compression.
* Default: true

compressed = <boolean>
* DEPRECATED. Use 'useSSLCompression' instead.
* Used only if 'useSSLCompression' is not set.

requireClientCert = <boolean>
* Requires that any peer that connects to replication port has a certificate that can be validated by certificate authority specified in rootCA.
* Default: false

allowSslRenegotiation = <boolean>
* 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

sslCommonNameToCheck = <commonName1>, <commonName2>, ...
* Optional.
* Check the common name of the client's certificate against this list of names.
* requireClientCert must be set to "true" for this setting to work.
* No default.

sslAltNameToCheck = <alternateName1>, <alternateName2>, ...
* Optional.
* Check the alternate name of the client's certificate against this list of names.
* If there is no match, assume that Splunk is not authenticated against this server.
* requireClientCert must be set to true for this setting to work.
* No default.

**Introspection settings**

```
[introspection:generator:disk_objects]
* For 'introspection_generator_addon', packaged with Splunk; provides the data ('i-data') consumed, and reported on, by 'introspection_viewer_app' (due to ship with a future release).
* This stanza controls the collection of i-data about: indexes; bucket superdirectories (homePath, coldPath, ...); volumes; search dispatch artifacts.
* On forwarders the collection of index, volumes and dispatch disk objects is disabled.

acquireExtra_i_data = true | false
* If true, extra Disk Objects i-data is emitted; you can gain more insight into your site, but at the cost of greater resource consumption both directly (the collection itself) and indirectly (increased disk and bandwidth utilization, to store the produced i-data).
* Please consult documentation for list of regularly emitted Disk Objects i-data, and extra Disk Objects i-data, appropriate to your release.
* Default: false

collectionPeriodInSecs = <positive integer>
* Controls frequency of Disk Objects i-data collection; higher frequency (hence, smaller period) gives a more accurate picture, but at the cost of greater resource consumption both directly (the collection itself) and indirectly (increased disk and bandwidth utilization, to store the produced i-data).
* Default: 600 (10 minutes)
```

```
[introspection:generator:disk_objects__indexes]
* This stanza controls the collection of i-data about indexes.
* Inherits the values of 'acquireExtra_i_data' and 'collectionPeriodInSecs' attributes from the 'introspection:generator:disk_objects' stanza, but may be enabled/disabled independently of it.
* This stanza should only be used to force collection of i-data about indexes on dedicated forwarders.
* Default: Data collection is disabled on universal forwarders and enabled on all other installations.

[introspection:generator:disk_objects__volumes]
* This stanza controls the collection of i-data about volumes.
* Inherits the values of 'acquireExtra_i_data' and 'collectionPeriodInSecs' attributes from the 'introspection:generator:disk_objects' stanza, but may be enabled/disabled independently of it.
* This stanza should only be used to force collection of i-data about volumes on dedicated forwarders.
* Default: Data collection is disabled on universal forwarders and enabled on all other installations.

[introspection:generator:disk_objects__dispatch]
* This stanza controls the collection of i-data about search dispatch artifacts.
* Inherits the values of 'acquireExtra_i_data' and 'collectionPeriodInSecs' attributes from the 'introspection:generator:disk_objects' stanza, but
may be enabled/disabled independently of it.
* This stanza should only be used to force collection of i-data about
  search dispatch artifacts on dedicated forwarders.
* Default: Data collection is disabled on universal forwarders and
  enabled on all other installations.

[introspection:generator:disk_objects__fishbucket]
* This stanza controls the collection of i-data about:
  $SPLUNK_DB/fishbucket, where per-input status of file-based
  inputs is persisted.
* Inherits the values of 'acquireExtra_i_data' and 'collectionPeriodInSecs'
  attributes from the 'introspection:generator:disk_objects' stanza, but may
  be enabled/disabled independently of it.

[introspection:generator:disk_objects__bundle_replication]
* This stanza controls the collection of i-data about:
  bundle replication metrics of distributed search
* Inherits the values of 'acquireExtra_i_data' and 'collectionPeriodInSecs'
  attributes from the 'introspection:generator:disk_objects' stanza, but may
  be enabled/disabled independently of it.

[introspection:generator:disk_objects__partitions]
* This stanza controls the collection of i-data about: disk partition space
  utilization.
* Inherits the values of 'acquireExtra_i_data' and 'collectionPeriodInSecs'
  attributes from the 'introspection:generator:disk_objects' stanza, but may
  be enabled/disabled independently of it.

[introspection:generator:disk_objects__summaries]
* Introspection data about summary disk space usage. Summary disk usage
  includes both data model and report summaries. The usage is collected
  for each summaryId, locally at each indexer.
  
  disabled = true | false
  * If not specified, inherits the value from
  
  collectionPeriodInSecs = <positive integer>
  * Controls frequency, in seconds, of Disk Objects - summaries
    collection; higher frequency (hence, smaller period) gives a more accurate
    picture, but at the cost of greater resource consumption directly
    (the summaries collection itself);
    it is not recommended for a period less than 15 minutes.
  * If you enable summary collection, the first collection happens 5 minutes
    after the Splunk instance is started. For every subsequent collection, this
    setting is honored.
  * If 'collectionPeriodInSecs' is smaller than 5 * 60, it resets to
    30 minutes internally.
  * Set to (N*300) seconds. Any remainder is ignored.
  * Default: 1800 (30 minutes)

[introspection:generator:resource_usage]
* For 'introspection_generator_addon', packaged with Splunk; provides the
data ("i-data") consumed, and reported on, by 'introspection_viewer_app'
due to ship with a future release).
* "Resource Usage" here refers to: CPU usage; scheduler overhead; main
  (physical) memory; virtual memory; pager overhead; swap; I/O; process
  creation (a.k.a. forking); file descriptors; TCP sockets; receive/transmit
  networking bandwidth.
* Resource Usage i-data is collected at both hostwide and per-process
  levels; the latter, only for processes associated with this SPLUNK_HOME.
* Per-process i-data for Splunk search processes include additional,
acquireExtra_i_data = true | false
* If set to true, extra Resource Usage i-data is emitted; you can gain more insight into your site, but at the cost of greater resource consumption both directly (the collection itself) and indirectly (increased disk and bandwidth utilization, to store the produced i-data).
* Please consult documentation for list of regularly emitted Resource Usage i-data, and extra Resource Usage i-data, appropriate to your release.
* Default: false

collectionPeriodInSecs = <positive integer>
* Controls frequency of Resource Usage i-data collection; higher frequency (hence, smaller period) gives a more accurate picture, but at the cost of greater resource consumption both directly (the collection itself) and indirectly (increased disk and bandwidth utilization, to store the produced i-data).
* Default: 600 (10 minutes) on Universal Forwarders, and 10 (1/6th of a minute) on non-Universal Forwarders

[introspection:generator:resource_usage__iostats]
* This stanza controls the collection of i-data about: IO Statistics data
  * "IO Statistics" here refers to: read/write requests; read/write sizes; io service time; cpu usage during service
  * IO Statistics i-data is sampled over the collectionPeriodInSecs
  * Does not inherit the value of the 'collectionPeriodInSecs' attribute from the 'introspection:generator:resource_usage' stanza, and may be enabled/disabled independently of it.

collectionPeriodInSecs = <positive integer>
* Controls interval of IO Statistics i-data collection; higher intervals gives a more accurate picture, but at the cost of greater resource consumption both directly (the collection itself) and indirectly (increased disk and bandwidth utilization, to store the produced i-data).
* Default: 60 (1 minute)

[introspection:generator:kvstore]
* For 'introspection_generator_addon', packaged with Splunk Enterprise.
  * "KV Store" here refers to: statistics information about KV Store process.

serverStatsCollectionPeriodInSecs = <positive integer>
* Controls frequency, in seconds, of KV Store server status collection
* Default: 27

operationStatsCollectionPeriodInSecs = <positive integer>
* Controls frequency, in seconds, of KV Store operation statistics collection (currentOp).
* Default: 60 seconds

collectionStatsCollectionPeriodInSecs = <positive integer>
* Controls frequency, in seconds, of KV Store db statistics collection.
* Default: 600 (10 minutes)

profilingStatsCollectionPeriodInSecs = <positive integer>
* Controls frequency, in seconds, of KV Store profiling data collection.
* Default: 5 seconds

rsStatsCollectionPeriodInSecs = <positive integer>
* Controls frequency, in seconds, of KV Store replica set stats collection
* Default: 60 seconds

[introspection:distributed-indexes]
* This stanza controls the collection of information for distributed indexes.
disabled = <boolean>
* Whether or not collection of introspection information on distributed
  indexes is disabled.
* If set to "false", information on distributed indexes is collected.
* This provides additional insight into index usage at the cost of greater
  resource consumption.
* Default: true

collectionPeriodInSecs = <positive integer>
* The frequency, in seconds, of distributed index data collection.
  Shorter intervals provide more accurate results, at the cost of
  greater resource consumption.
* Default: 3600 (60 minutes)

Settings used to control commands started by Splunk

[commands:user_configurable]

prefix = <path>
* All non-internal commands started by splunkd are prefixed with this
  string, allowing for "jailed" command execution.
* Should be only one word. In other words, commands are supported, but
  commands and arguments are not.
* Applies to commands such as: search scripts, scripted inputs, SSL
  certificate generation scripts. (Any commands that are
  user-configurable).
* Does not apply to trusted/non-configurable command executions, such as:
  splunk search, splunk-optimize, gunzip.
* No default.

Data Fabric Search (DFS) Configuration

[dfs]

disabled = <boolean>
* When set to 'false' for the [dfs] stanza, this setting enables data fabric
  search functionality for this instance.
* A 'false' setting causes the Splunk software to start the DFSMaster Java
  process in a separate process. This process is central to Data Fabric Search
  functionality.
* Default: true

dfc_ip_address = <IP address>
* This setting provides the host IP address of the data fabric coordinator
  (DFC) process.
* Data Fabric Search uses the DFC process to run DFS queries on DFS
  worker nodes. The DFC process needs to know its own IP address.
* Default: If an IP address is not provided, the DFC process uses the
  IP address of the local host.

port = <port>
* Identifies the port on which the DFSMaster Java process runs.
* Default: 9000
extra_kryo_registered_classes = <class names>
* This setting provides a comma-separated list of fully-qualified Java class names.
* When this list exists, Spark registers the list of classes for Kryo serialization. Spark uses the Kryo library to serialize Java objects.
* Default: empty string

spark_master_host = [<IP address>|<host name>]
* This setting identifies the Spark master.
* Default: 127.0.0.1

spark_master_webui_port = <port>
* Identifies the port for the Spark master web UI.
* Default: 8080

spark_master_connect_timeout = <unsigned integer>
* Sets the timeout (in seconds) for the initial connection from the DFS master to the Spark master.
* A value of 0 indicates no timeout. The process will attempt to establish a connection indefinitely.
* Default: 10 seconds

spark_home = <path>
* Absolute location of Spark home.
* Used only if SPARK_HOME is unset.
* No default.

connection_timeout = <integer>
* Low-level timeout, in seconds, for establishing a connection between a search peer and a DFS worker.
* Default: 180

connection_retries = <integer>
* Number of retries to establish a connection between a search peer and a DFS worker.
* Default: 5

[app_backup]
backup_path = <path>
* Full path to the directory that contains configuration backups created by Splunk Enterprise.
* For search head clusters, this directory resides on the deployer.
* Default: $SPLUNK_HOME/var/backup

search head clustering configuration

[shclustering]
disabled = <boolean>
* Disables or enables search head clustering on this instance.
* When enabled, the captain needs to be selected via a bootstrap mechanism. Once bootstrapped, further captain selections are made via a dynamic election mechanism.
* When enabled, you must also specify the cluster member's own server address / management URI for identification purpose. This can be done in 2 ways: by specifying the 'mgmt_uri' setting individually on each member or by specifying pairs of 'GUID, mgmt-uri' strings in the servers_list attribute.
* Default: true

mgmt_uri = [ mgmt-URI ]
* The management URI is used to identify the cluster member's own address to itself.
* Either 'mgmt_uri' or 'servers_list' is necessary.
* The 'mgmt_uri' setting is simpler to author but is unique for each member.
* The 'servers_list' setting is more involved, but can be copied as a config string to all members in the cluster.

servers_list = [ <(GUID, mgmt-uri);>+ ]
* A semicolon separated list of instance GUIDs and management URIs.
* Each member uses its GUID to identify its own management URI.

adhoc_searchhead = <boolean>
* This setting configures a member as an ad-hoc search head; i.e., the member does not run any scheduled jobs.
* Use the setting 'captain_is_adhoc_searchhead' to reduce compute load on the captain.
* Default: false

no_artifact_replications = <boolean>
* Prevent this Search Head Cluster member to be selected as a target for replications.
* This is an advanced setting, and not to be changed without proper understanding of the implications.
* Default: false

captain_is_adhoc_searchhead = <boolean>
* This setting prohibits the captain from running scheduled jobs.
* The captain is dedicated to controlling the activities of the cluster, but can also run adhoc search jobs from clients.
* Default: false

preferred_captain = <boolean>
* The cluster tries to assign captaincy to a member with 'preferred_captain=true'.
* Note that it is not always possible to assign captaincy to a member with preferred_captain=true - for example, if none of the preferred members is reachable over the network. In that case, captaincy might remain on a member with preferred_captain=false.
* Default: true

prevent_out_of_sync_captain = <boolean>
* This setting prevents a node that could not sync config changes to current captain from becoming the cluster captain.
* This setting takes precedence over the preferred_captain setting. For example, if there are one or more preferred captain nodes but the nodes cannot sync config changes with the current captain, then the current captain retains captaincy even if it is not a preferred captain.
* This must be set to the same value on all members.
* Default: true

replication_factor = <positive integer>
* Determines how many copies of search artifacts are created in the cluster.
* This must be set to the same value on all members.
* Default: 3

pass4SymmKey = <password>
* Secret shared among the members in the search head cluster to prevent any arbitrary instance from connecting to the cluster.
* All members must use the same value.
* If set in the [shclustering] stanza, it takes precedence over any setting in the [general] stanza.
* Unencrypted passwords must not begin with "$1$", as this is used by
Splunk software to determine if the password is already encrypted.
* Default: The 'changeme' from the [general] stanza in the default the
server.conf file.

async_replicate_on_proxy = <boolean>
* If the jobs/${sid}/results REST endpoint had to be proxied to a different
member due to missing local replica, this attribute automatically
schedules an async replication to that member when set to true.
* Default: true

master_dump_service_periods = <integer>
* If SHPMaster info is switched on in log.cfg, then captain statistics
are dumped in splunkd.log after the specified number of service periods.
Purely a debugging aid.
* Default: 500

long_running_jobs_poll_period = <integer>
* Long running delegated jobs are polled by the captain every
"long_running_jobs_poll_period" seconds to ascertain whether they are
still running, in order to account for potential node/member failure.
* Default: 600 (10 minutes)

scheduling_heuristic = <string>
* This setting configures the job distribution heuristic on the captain.
* There are currently two supported strategies: 'round_robin' or
'scheduler_load_based'.
* Default: 'scheduler_load_based'

id = <GUID>
* Unique identifier for this cluster as a whole, shared across all cluster
members.
* Default: Splunk software arranges for a unique value to be generated and
shared across all members.

cxn_timeout = <integer>
* Low-level timeout, in seconds, for establishing connection between
cluster members.
* Default: 60

send_timeout = <integer>
* Low-level timeout, in seconds, for sending data between search head
cluster members.
* Default: 60

rcv_timeout = <integer>
* Low-level timeout, in seconds, for receiving data between search head
cluster members.
* Default: 60

cxn_timeout_raft = <integer>
* Low-level timeout, in seconds, for establishing connection between search
head cluster members for the raft protocol.
* Default: 2

send_timeout_raft = <integer>
* Low-level timeout, in seconds, for sending data between search head
cluster members for the raft protocol.
* Default: 5

rcv_timeout_raft = <integer>
* Low-level timeout, in seconds, for receiving data between search head
cluster members for the raft protocol.
* Default: 5

`rep_cxn_timeout = <integer>`
* Low-level timeout, in seconds, for establishing connection for replicating data.
* Default: 5

`rep_send_timeout = <integer>`
* Low-level timeout, in seconds, for sending replication slice data between cluster members.
  * This is a soft timeout. When this timeout is triggered on source peer, it tries to determine if target is still alive. If it is still alive, it reset the timeout for another `rep_send_timeout` interval and continues. If target has failed or cumulative timeout has exceeded `rep_max_send_timeout`, replication fails.
* Default: 5

`rep_rcv_timeout = <integer>`
* Low-level timeout, in seconds, for receiving acknowledgement data from members.
  * This is a soft timeout. When this timeout is triggered on source member, it tries to determine if target is still alive. If it is still alive, it reset the timeout for another `rep_send_timeout` interval and continues. If target has failed or cumulative timeout has exceeded the 'rep_max_rcv_timeout' setting, replication fails.
* Default: 10

`rep_max_send_timeout = <integer>`
* Maximum send timeout, in seconds, for sending replication slice data between cluster members.
  * On 'rep_send_timeout' source peer determines if total send timeout has exceeded `rep_max_send_timeout`. If so, replication fails.
  * If cumulative `rep_send_timeout` exceeds 'rep_max_send_timeout', replication fails.
* Default: 600 (10 minutes)

`rep_max_rcv_timeout = <integer>`
* Maximum cumulative receive timeout, in seconds, for receiving acknowledgement data from members.
  * On 'rep_rcv_timeout' source member determines if total receive timeout has exceeded 'rep_max_rcv_timeout'. If so, replication fails.
* Default: 600 (10 minutes)

`log_heartbeat_append_entries = <boolean>`
* If true, Splunk software logs the the low-level heartbeats between members in `splunkd_access.log` file. These heartbeats are used to maintain the authority of the captain authority over other members.
* Default: false

`election_timeout_ms = <positive_integer>`
* The amount of time, in milliseconds, that a member waits before trying to become the captain.
  * Note that modifying this value can alter the heartbeat period (See `election_timeout_2_hb_ratio` for further details)
  * A very low value of `election_timeout_ms` can lead to unnecessary captain elections.
* Default: 60000 (1 minute)

`election_timeout_2_hb_ratio = <positive_integer>`
* The ratio between the election timeout, set in `election_timeout_ms`, and the raft heartbeat period.
  * Raft heartbeat period = `election_timeout_ms / election_timeout_2_hb_ratio`
* A typical ratio between 5 - 20 is desirable. Default is 12 to keep the
timeout period at 5s, i.e election_timeout_ms(60000ms) / 12
* This ratio determines the number of heartbeat attempts that would fail
  before a member starts to timeout and tries to become the captain.

heartbeat_timeout = <positive integer>
* The amount of time, in seconds, that the captain considers a member down.
  After a member is down, the captain initiates fixup steps to replicate
  artifacts from the dead member to its peers.
* This heartbeat exchanges data between the captain and members, which helps in
  maintaining the in-memory centralized state for all the cluster members.
* Note that this heartbeat is different from the Raft heartbeat described
  in the 'election_timeout_2_hb_ratio' setting.
* Default: 60 (1 minute)

raft_rpc_backoff_time_ms = <positive integer>
* Provides a delay, in milliseconds, should a raft RPC request fail.
* This avoids rapid connection requests being made to unreachable peers.
* This setting should not normally be changed from the default.
* Default: 5000 (5 seconds)

access_logging_for_heartbeats = <boolean>
* Only valid on captain
* Enables/disables logging to the splunkd_access.log file for member heartbeats
* NOTE: you do not have to restart captain to set this config parameter.
  Simply run the cli command on master:
  % splunk edit shcluster-config -access_logging_for_heartbeats <<boolean>>
* Default: false (logging disabled)

restart_timeout = <positive integer>
* This is the amount of time the captain waits for a member to come
  back when the instance is restarted (to avoid the overhead of
  trying to fixup the artifacts that were on the peer).

quiet_period = <positive integer>
* Determines the amount of time, in seconds, for which a newly
  elected captain waits for members to join. During this period the
  captain does not initiate any fixups but instead waits for the
  members to register themselves. Job scheduling and conf
  replication still happen as usual during this time. At the end
  of this time period, the captain builds its view of the cluster
  based on the registered peers and starts normal
  processing.
* Default: 60

max_peer_rep_load = <integer>
* This is the maximum number of concurrent replications that a
  member can take part in as a target.
* Default: 5

target_wait_time = <positive integer>
* Specifies the time, in seconds, that the captain waits for the target
  of a replication to register itself before it services the artifact again
  and potentially schedules another fixup.
* Default: 150

manual_detention = on|off
* This property toggles manual detention on member.
* When a node is in manual detention, it does not accept new search jobs,
  including both scheduled and ad-hoc searches. It also does not receive
  replicated search artifacts from other nodes.
* Default: off
percent_peers_to_restart = <integer>
* The percentage of members to restart at one time during rolling restarts.
* Actual percentage may vary due to lack of granularity for smaller peer sets regardless of setting, a minimum of 1 peer is restarted per round.
* Valid values are between 0 and 100.
* CAUTION: Do not set this attribute to a value greater than 20%.
  Otherwise, issues can arise during the captain election process.

rolling_restart_with_captaincy_exchange = <boolean>
* If this boolean is turned on, captain tries to exchange captaincy with another node during rolling restart.
* If set to false, captain restarts and captaincy transfers to some other node.
* Default: true

rolling_restart = restart|searchable|searchable_force
* Determines the rolling restart mode for a search head cluster.
  * If set to restart, a rolling restart runs in classic mode.
  * If set to searchable, a rolling restart runs in searchable (minimal search disruption) mode.
  * If set to searchable_force, the search head cluster performs a searchable rolling restart, but overrides the health check
  * Note: You do not have to restart any search head members to set this parameter.
  Run this CLI command from any member:
  % splunk edit shcluster-config -rolling_restart restart|searchable|searchable_force
  * Default: restart (runs in classic rolling-restart mode)

decommission_search_jobs_wait_secs = <positive integer>
* The amount of time, in seconds, that a search head cluster member waits for existing searches to complete before restarting.
* Applies only when rolling restart is triggered in searchable or searchable_force mode (i.e. 'rolling_restart' is set to "searchable" or "searchable_force").
* Note: You do not have to restart search head members to set this parameter.
  Run this CLI command from any member:
  % splunk edit shcluster-config -decommission_search_jobs_wait_secs <positive integer>
* Default: 180

register_replication_address = <IP address or machine/domain name>
* This setting is the address on which a member is available for accepting replication data. This is useful in the cases where a member host machine has multiple interfaces and only one of them can be reached by another splunkd instance.
* Can be an IP address, or fully qualified machine/domain name.

executor_workers = <positive integer>
* Number of threads that can be used by the search head clustering threadpool.
* A value of 0 is interpreted as 1.
* Default: 10

heartbeat_period = <non-zero positive integer>
* Controls the frequency, in seconds, with which the member attempts to send heartbeats to the captain.
* This heartbeat exchanges data between the captain and members, which helps in maintaining the in-memory centralized state for all the cluster members.
* Note that this heartbeat period is different from the Raft heartbeat period in the election_timeout_2_hb_ratio setting.  
* Default: 5

enableS2SHeartbeat = <boolean>  
* Splunk software monitors each replication connection for presence of a heartbeat.  
* If the heartbeat is not seen for s2sHeartbeatTimeout seconds, it closes the connection.  
* Default: true

s2sHeartbeatTimeout = <integer>  
* This specifies the global timeout, in seconds, value for monitoring heartbeats on replication connections.  
* Splunk software closes a replication connection if a heartbeat is not seen for 's2sHeartbeatTimeout' seconds.  
* Replication source sends a heartbeat every 30 seconds.  
* Default: 600 (10 minutes)

captain_uri = [ static-captain-URI ]  
* The management URI of static captain is used to identify the cluster captain for a static captain.

election = <boolean>  
* This is used to classify a cluster as static or dynamic (RAFT based).  
* If set to "false", a static captain, which is used for DR situation.  
* If set to "true", a dynamic captain election enabled through RAFT protocol.

mode = <member>  
* Accepted values are captain and member, mode is used to identify the function of a node in static search head cluster. Setting mode as captain assumes it to function as both captain and a member.

proxying related
sid_proxying = <boolean>  
* Enable or disable search artifact proxying.  
* Changing this affects the proxying of search results, and jobs feed is not cluster-aware.  
* Only for internal/expert use.  
* Default: true

ss_proxying = <boolean>  
* Enable or disable saved search proxying to captain.  
* Changing this affects the behavior of Searches and Reports page in Splunk Web.  
* Only for internal/expert use.  
* Default: true

ra_proxying = <boolean>  
* Enable or disable saved report acceleration summaries proxying to captain.  
* Changing this affects the behavior of report acceleration summaries page.  
* Only for internal/expert use.  
* Default: true

alert_proxying = <boolean>  
* Enable or disable alerts proxying to captain.  
* Changing this impacts the behavior of alerts, and essentially make them not cluster-aware.  
* Only for internal/expert use.  
* Default: true

261
csv_journal_rows_per_hb = <integer>
* Controls how many rows of CSV from the delta-journal are sent per hb
* Used for both alerts and suppressions
* Do not alter this value without contacting Splunk Support.
* Default: 10000

conf_replication_period = <integer>
* Controls how often, in seconds, a cluster member replicates
  configuration changes.
* A value of 0 disables automatic replication of configuration changes.
* Default: 5

conf_replication_max_pull_count = <integer>
* Controls the maximum number of configuration changes a member
  replicates from the captain at one time.
* A value of 0 disables any size limits.
* Default: 1000

conf_replication_max_push_count = <integer>
* Controls the maximum number of configuration changes a member
  replicates to the captain at one time.
* A value of 0 disables any size limits.
* Default: 100

conf_replication_max_json_value_size = [<integer>|<integer>[KB|MB|GB]]
* Controls the maximum size of a JSON string element at any nested
  level while parsing a configuration change from JSON representation.
* If a knowledge object created on a member has some string element
  that exceeds this limit, the knowledge object is not replicated
  to the rest of the search head cluster, and a warning that mentions
  conf_replication_max_json_value_size is written to splunkd.log.
* If you do not specify a unit for the value, the unit defaults to bytes.
* The lower limit of this setting is 512KB.
* When increasing this setting beyond the default, you must take into
  account the available system memory.
* Default: 15MB

conf_replication_include.<conf_file_name> = <boolean>
* Controls whether Splunk replicates changes to a particular type of *.conf
  file, along with any associated permissions in *.meta files.
* Default: false

conf_replication_summary.whitelist.<name> = <whitelist_pattern>
* Whitelist files to be included in configuration replication summaries.

conf_replication_summary.blacklist.<name> = <blacklist_pattern>
* Blacklist files to be excluded from configuration replication summaries.

conf_replication_summary.concerning_file_size = <integer>
* Any individual file within a configuration replication summary that is
  larger than this value (in MB) triggers a splunkd.log warning message.
* Default: 50

conf_replication_summary.period = <timespan>
* Controls how often configuration replication summaries are created.
* Default: 1m (1 minute)

conf_replication_purge.eligible_count = <integer>
* Controls how many configuration changes must be present before any become
  eligible for purging.
* In other words: controls the minimum number of configuration changes
  Splunk software remembers for replication purposes.
conf_replication_purge.eligibile_age = <timespan>
* Controls how old a configuration change must be before it is eligible for
  purging.
* Default: 1d (1 day).

conf_replication_purge.period = <timespan>
* Controls how often configuration changes are purged.
* Default: 1h (1 hour)

conf_replication_find_baseline.use_bloomfilter_only = <boolean>
* Controls whether or not a search head cluster only uses bloom filters to
determine a baseline, when it replicates configurations.
* Set to true to only use bloom filters in baseline determination during
  configuration replication.
* Set to false to first attempt a standard method, where the search head
  cluster captain interacts with members to determine the baseline, before
  falling back to using bloom filters.
* Default: false

conf_deploy_repository = <path>
* Full path to directory containing configurations to deploy to cluster
  members.

conf_deploy_staging = <path>
* Full path to directory where preprocessed configurations may be written
  before being deployed cluster members.

conf_deploy_concerning_file_size = <integer>
* Any individual file within <conf_deploy_repository> that is larger than
  this value (in MB) triggers a splunkd.log warning message.
* Default: 50

conf_deploy_fetch_url = <URL>
* Specifies the location of the deployer from which members fetch the
  configuration bundle.
* This value must be set to a <URL> in order for the configuration bundle to
  be fetched.
* No default.

conf_deploy_fetch_mode = auto|replace|none
* Controls configuration bundle fetching behavior when the member starts up.
* When set to "replace", a member checks for a new configuration bundle on
  every startup.
* When set to "none", a member does not fetch the configuration bundle on
  startup.
* Regarding "auto":
  * If no configuration bundle has yet been fetched, "auto" is equivalent
to "replace".
  * If the configuration bundle has already been fetched, "auto" is
equivalent to "none".
* Default: replace

artifact_status_fields = <field> ...
* Give a comma separated fields to pick up values from status.csv and
  info.csv for each search artifact.
* These fields are shown in the CLI/REST endpoint splunk list
  shcluster-member-artifacts
* Default: user, app, label

encrypt_fields = <field> ...
* These are the fields that need to be re-encrypted when a Search Head Cluster does its own first time run on syncing all members with a new splunk.secret key
* Give a comma separated fields as a triple elements
  <conf-file>:<stanza-prefix>:<key elem>
  * For matching all stanzas from a conf, leave the stanza-prefix empty. For example: "server: :pass4SymmKey" matches all stanzas with pass4SymmKey as key in server.conf
* Default: storage/passwords, secret key for clustering/shclustering,
  server ssl config

enable_jobs_data_lite = <boolean>
* This is for memory reduction on the captain for Search head clustering,
  leads to lower memory in captain while slaves send the artifacts
  status.csv as a string.
* Default: true

shcluster_label = <string>
* This specifies the label of the search head cluster.

retry_autosummarize_or_data_model_acceleration_jobs = <boolean>
* Controls whether the captain tries a second time to delegate an
  auto-summarized or data model acceleration job, if the first attempt to
  delegate the job fails.
* Default: true

deployerPushThreads = <positive integer>|auto
* The maximum number of threads to use when performing a deployer bundle push
to target members.
* If set to "auto", the deployer auto-tunes the number of threads it uses
  for a deployer bundle push. There will be one thread per target member.
* Default: 1

[replication_port://<port>]
Configures the member to listen on a given TCP port for replicated data
from another cluster member.
At least one replication_port must be configured and not disabled.

disabled = <boolean>
* Set to true to disable this replication port stanza.
* Default: false

listenOnIPv6 = no|yes|only
* Toggle whether this listening port listens on IPv4, IPv6, or both.
* If not present, the setting in the [general] stanza is used.

acceptFrom = <network_acl> ...
* Lists a set of networks or addresses from which to accept connections.
* 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)

[replication_port-ssl://<port>]
* This configuration is the same as the replication_port stanza, but uses SSL.

disabled = <boolean>
* Set to true to disable this replication port stanza.
* Default: false

listenOnIPv6 = no|yes|only
* Toggle whether this listening port listens on IPv4, IPv6, or both.
* Default: The setting in the [general] stanza

acceptFrom = <network_acl> ...
* This setting is the same as the setting in the [replication_port] stanza.

serverCert = <path>
* Full path to file containing private key and server certificate.
* The <path> must refer to a PEM format file.
* No default.

sslPassword = <password>
* Server certificate password, if any.
* No default.

password = <password>
* DEPRECATED; use 'sslPassword' instead.
* Used only if 'sslPassword' is not set.

rootCA = <path>
* DEPRECATED; use '[sslConfig]/sslRootCAPath' instead.
* Used only if '[sslConfig]/sslRootCAPath' is not set.
* Full path to the root CA (Certificate Authority) certificate store.
* The <path> must refer to a PEM format file containing one or more root CA certificates concatenated together.
* No default.

cipherSuite = <cipher suite string>
* If set, uses the specified cipher string for the SSL connection.
* 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.

supportSSLV3Only = <boolean>
* DEPRECATED. SSLv2 is now always disabled. The exact set of SSL versions allowed is now configurable via the "sslVersions" setting above.

useSSLCompression = <boolean>
* If true, enables SSL compression.
* Default: false

compressed = <boolean>
* DEPRECATED; use 'useSSLCompression' instead.
* Used only if 'useSSLCompression' is not set.

requireClientCert = <boolean>
* Requires that any peer that connects to replication port has a certificate that can be validated by certificate authority specified in rootCA.
* Default: false

allowSslRenegotiation = <boolean>
* 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

**KV Store configuration**

[kvstore]

disabled = <boolean>
* Set to true to disable the KV Store process on the current server. To completely disable KV Store in a deployment with search head clustering or search head pooling, you must also disable KV Store on each individual server.
* Default: false

port = <port>
* Port to connect to the KV Store server.
* Default: 8191

replicaset = <replset>
* Replicaset name.
* Default: splunkrs

distributedLookupTimeout = <seconds>
* This setting has been removed, as it is no longer needed.

shutdownTimeout = <integer>
* Time, in seconds, to wait for a clean shutdown of the KV Store. If this time is reached after signaling for a shutdown, KV Store is forcibly terminated.
* Default: 100

initAttempts = <integer>
* The maximum number of attempts to initialize the KV Store when starting splunkd.
* Default: 300

replication_host = <host>
* The host name to access the KV Store.
* This setting has no effect on a single Splunk instance.
* When using search head clustering, if the "replication_host" value is not set in the [kvstore] stanza, the host you specify for "mgmt_uri" in the [shclustering] stanza is used for KV Store connection strings and replication.
* In search head pooling, this host value is a requirement for using KV Store.
* This is the address on which a kvstore is available for accepting remotely.

verbose = <boolean>
* Set to true to enable verbose logging.
* Default: false

verboseLevel = <nonnegative integer>
* When verbose logging is enabled specify verbose level for logging from 0 to 5, where 5 is the most verbose.
* Default: 2
dbPath = <path>
* Path where KV Store data is stored.
  * Changing this directory after initial startup does not move existing data.
    The contents of the directory should be manually moved to the new location.
  * Default: $SPLUNK_DB/kvstore

oplogSize = <integer>
* The size of the replication operation log, in MB, for environments
  with search head clustering or search head pooling.
  In a standalone environment, 20% of this size is used.
  * After the KV Store has created the oplog for the first time, changing this
    setting does NOT affect the size of the oplog. A full backup and restart
    of the KV Store is required.
  * Do not change this setting without first consulting with Splunk Support.
  * Default: 1000MB (1GB)

replicationWriteTimeout = <integer>
* The time to wait, in seconds, for replication to complete while saving KV
  store operations. When the value is 0, the process never times out.
  * Used for replication environments (search head clustering or search
    head pooling).
  * Default: 1800 (30 minutes)

certsFile = <path>
* DEPRECATED; use '[sslConfig]/sslRootCAPath' instead.
* Used only if 'sslRootCAPath' is not set.
* Full path to a CA (Certificate Authority) certificate(s) PEM format file.
  * If specified, it is used in KV Store SSL connections and
    authentication.
  * Only used when Common Criteria is enabled (SPLUNK_COMMON_CRITERIA=1)
    or FIPS is enabled (i.e. SPLUNK_FIPS=1).
  * NOTE: Splunk plans to submit Splunk Enterprise for Common Criteria
    evaluation. Splunk does not support using the product in Common
    Criteria mode until it has been certified by NIAP. See the "Securing
    Splunk Enterprise" manual for information on the status of Common
    Criteria certification.
  * Default: $SPLUNK_HOME/etc/auth/cacert.pem

certsPath = <filepath>
* DEPRECATED; use '[sslConfig]/sslRootCAPath' instead.

serverCert = <filepath>
* A certificate file signed by the signing authority specified above by
  certsPath.
  * In search head clustering or search head pooling, the certificates at
different members must share the same 'subject'.
  * The Distinguished Name (DN) found in the certificate's subject, must
specify a non-empty value for at least one of the following attributes:
Organization (O), the Organizational Unit (OU) or the
Domain Component (DC).
  * Only used when Common Criteria is enabled (SPLUNK_COMMON_CRITERIA=1)
    or FIPS is enabled (i.e. SPLUNK_FIPS=1).
  * NOTE: Splunk plans to submit Splunk Enterprise for Common Criteria
    evaluation. Splunk does not support using the product in Common
    Criteria mode until it has been certified by NIAP. See the "Securing
    Splunk Enterprise" manual for information on the status of Common
    Criteria certification.

sslKeysPath = <filepath>
* DEPRECATED; use 'serverCert' instead.
* Used only when 'serverCert' is empty.

`sslPassword = <password>`
* Password of the private key in the file specified by 'serverCert' above.
* Must be specified if FIPS is enabled (i.e. SPLUNK_FIPS=1), otherwise, KV Store is not available.
* Only used when Common Criteria is enabled (SPLUNK_COMMON_CRITERIA=1) or FIPS is enabled (i.e. SPLUNK_FIPS=1).
* NOTE: Splunk plans to submit Splunk Enterprise for Common Criteria evaluation. Splunk does not support using the product in Common Criteria mode until it has been certified by NIAP. See the "Securing Splunk Enterprise" manual for information on the status of Common Criteria certification.
* No default.

`sslKeysPassword = <password>`
* DEPRECATED; use 'sslPassword' instead.
* Used only when 'sslPassword' is empty.

`sslCRLPath = <filepath>`
* Only used when Common Criteria is enabled (SPLUNK_COMMON_CRITERIA=1) or FIPS is enabled (i.e. SPLUNK_FIPS=1).
* NOTE: Splunk plans to submit Splunk Enterprise for Common Criteria evaluation. Splunk does not support using the product in Common Criteria mode until it has been certified by NIAP. See the "Securing Splunk Enterprise" manual for information on the status of Common Criteria certification.
* Optional.
* Default: empty string (no Revocation List)

`modificationsReadIntervalMillisec = <integer>`
* How often, in milliseconds, to check for modifications to KV Store collections in order to replicate changes for distributed searches.
* Default: 1000 (1 second)

`modificationsMaxReadSec = <integer>`
* Maximum time interval KVStore can spend while checking for modifications before it produces collection dumps for distributed searches.
* Default: 30

`initialSyncMaxFetcherRestarts = <positive integer>`
* Specifies the maximum number of query restarts an oplog fetcher can perform before failing the ongoing Initial Sync attempt.
* Increasing this value might help in dynamic deployments with very large KV Store databases where Initial Sync might take a long time.
* NOTE: This setting should be changed only if you have been asked to set it by a Splunk Support engineer. It might increase KV Store cluster failover time.
* Default: 0

**Indexer Discovery configuration**

`[indexer_discovery]`

`pass4SymmKey = <password>`
* Security key shared between master node and forwarders.
* If specified here, the same value must also be specified on all forwarders connecting to this master.
* Unencrypted passwords must not begin with "$1$", as this is used by Splunk software to determine if the password is already encrypted.
polling_rate = <integer>
* A value between 1 to 10. This value affects the forwarder polling frequency to achieve the desired polling rate. The number of connected forwarders is also taken into consideration.
* The formula used to determine effective polling interval, in Milliseconds, is:
  \( \text{number of forwarders/polling_rate + 30 seconds} \times 1000 \)
* Default: 10

indexerWeightByDiskCapacity = <boolean>
* If set to true, it instructs the forwarders to use weighted load balancing. In weighted load balancing, load balancing is based on the total disk capacity of the target indexers, with the forwarder streaming more data to indexers with larger disks.
* The traffic sent to each indexer is based on the ratio of:
  \( \frac{\text{indexer_disk_capacity}}{\text{total_disk_capacity_of_indexers_combined}} \)
* Default: false

**Cascading Replication Configuration**

[cascading_replication]
pass4SymmKey = <password>
* Security key shared between indexers participating in cascading replication.
* The same value must be specified on all indexers participating in cascading replication.
* Unencrypted passwords must not begin with "$1$", as this is used by Splunk software to determine if the password is already encrypted.
* Empty passwords will not be accepted.
* Default: None

max_replication_threads = <integer>
* Maximum threads used for replicating metadata and payload to search peers.
* If set to "auto", the peer auto-tunes the number of threads it uses for cascading replication.
  * If the peer has 3 or fewer CPUs, it allocates 2 threads.
  * If the peer has 4-7 CPUs, it allocates up to \( \frac{\text{CPUs}}{2} \) threads.
  * If the peer has 8-15 CPUs, it allocates up to \( \frac{\text{CPUs}}{3} \) threads.
  * If the peer has 16 or more CPUs, it allocates up to \( \frac{\text{CPUs}}{4} \) threads.
  * Maximum accepted value for this setting is 16.
* Default: auto

max_replication_jobs = <integer>
* Maximum jobs used for replicating metadata and payload to search peers.
* Default: 5

cascade_replication_plan_reap_interval = <interval>
* The interval at which the cascade replication plans are reaped.
* The interval can be specified as a string for minutes, seconds, hours, or days. For example: 60s, 1m, 1h, 1d etc.
* Maximum accepted value is 5h
* Default: 1h

cascade_replication_plan_age = <interval>
* The age of the cascade replication plan when it gets reaped.
* The interval can be specified as a string for minutes, seconds, hours, or days. For example: 60s, 1m, 1h, 1d etc.
* Maximum accepted value is 24h
* Default: 8h
cascade_replication_plan_fanout = auto|<positive integer>
* Number of receivers that each sender replicates to at a time.
* If set to auto, Splunk automatically calculates an optimal fanout, based on
  the maximum number of replication threads, as determined by the
  'max_replication_threads' setting under [cascading_replication] in server.conf.
* If set to an integer, the integer must be no greater than the number of cluster
  peers, or, in the case of multisite clustering, no greater than the least number
  of peers on any one site.
* Default: auto

cascade_replication_plan_topology = size_balanced
* Topology used for building a cascading plan.
* When set to size_balanced, receivers are evenly distributed among senders.
  Senders on the same layer have same or similar number of receivers.
* Default: size_balanced

cascade_replication_plan_select_policy = random
* Policy for deciding which receivers the senders pick.
* When set to random, receivers are randomly picked.
* Default: random

Node level authentication

[node_auth]
signatureVersion = <comma-separated list>
* A list of authentication protocol versions that nodes of a Splunk
  deployment use to authenticate to other nodes.
* Each version of node authentication protocol implements an algorithm
  that specifies cryptographic parameters to generate authentication data.
* Nodes may only communicate using the same authentication protocol version.
* For example, if you set "signatureVersion = v1,v2" on one node, that
  node sends and accepts authentication data using versions "v1" and "v2"
  of the protocol, and you must also set "signatureVersion" to one of
  "v1", "v2", or "v1,v2" on other nodes for those nodes to mutually
  authenticate.
* For higher levels of security, set 'signatureVersion' to "v2".
* Default: v1,v2

Cache Manager Configuration

[cachemanager]
max_concurrent_downloads = <unsigned integer>
* The maximum number of buckets that can be downloaded simultaneously from
  external storage
* Default: 8

max_concurrent Uploads = <unsigned integer>
* The maximum number of buckets that can be uploaded simultaneously to external
  storage.
* Default: 8

eviction_policy = <string>
* The name of the eviction policy to use.
* Current options: lru, clock, random, lrlt, noevict, lruk
* Do not change the value from the default unless instructed by
  Splunk Support.
* Default: lru

enable_eviction_priorities = <boolean>
* When requesting buckets, search peers can give hints to the Cache Manager about the relative importance of buckets.
* When enabled, the Cache Manager takes the hints into consideration; when disabled, hints are ignored.
* Default: true

eviction_padding = <positive integer>
* Specifies the additional space, in megabytes, beyond 'minFreeSpace' that the cache manager uses as the threshold to start evicting data.
* If free space on a partition falls below ('minFreeSpace' + 'eviction_padding'), then the cache manager tries to evict data from remote storage enabled indexes.
* Default: 5120 (~5GB)

max_cache_size = <positive integer>
* Specifies the maximum space, in megabytes, per partition, that the cache can occupy on disk. If this value is exceeded, the cache manager starts evicting buckets.
* A value of 0 means this feature is not used, and has no maximum size.
* Default: 0

persist_pending_upload_from_external = <bool>
* Currently not supported. This setting is related to a feature that is still under development.
* Specifies whether the information of the buckets that have been uploaded to remote storage can be serialized to disk or not.
* When set to true, this information is serialized to disk and the bucket is deemed to be on remote storage.
* Otherwise, the bucket is deemed to be not on remote storage and bucket is then uploaded to remote storage.
* Default: true

persistent_id_set_remove_min_sync_secs = <unsigned integer>
* Currently not supported. This setting is related to a feature that is still under development.
* Cache manager persists the set of objects that are no longer pending upload to the remote storage based on when the previous set of changes were persisted to disk.
* This setting controls the interval from the last persist time that cache manager waits to persist the current set of changes to disk.
* Default: 5

enable_open_on_stale_object = <bool>
* Currently not supported. This setting is related to a feature that is still under development.
* Specifies whether the buckets with stale files can be opened for search.
* When set to true, these buckets can be opened for search.
* Otherwise, searches are not allowed to open these buckets.
* Default: true

hotlist_recency_secs = <unsigned integer>
* The cache manager attempts to defer bucket eviction until the interval between the bucket's latest time and the current time exceeds this setting, in seconds.
* This setting can be overridden on a per-index basis in indexes.conf.
* Default: 86400 (24 hours)

hotlist_bloom_filter_recency_hours = <unsigned integer>
* The cache manager attempts to defer eviction of the non-journal and non-tsidx
bucket files, such as the bloomfilter file, until the interval between the bucket's latest time and the current time exceeds this setting.
* This setting can be overridden on a per-index basis in indexes.conf.
* Default: 360 (15 days)

**evict_on_stable** = <boolean>
* When the source peer completes upload of a bucket to remote storage, it notifies the target peers so that they can evict any local copies of the bucket.
* When set to true, each target peer evicts its local copy, if any, upon such notification.
* When set to false, each target peer continues to store its local copy, if any, until its cache manager eventually evicts the bucket according to its cache eviction policy.
* Default: false

**max_file_exists_retry_count** = <unsigned integer>
* The cache manager retries its check on whether the file exists on remote storage when the check fails due to network errors until the retry count exceeds this setting.
* Default: 5

**Raft Statemachine configuration**

[raft_statemachine]

**disabled** = <boolean>
* Set to true to disable the raft statemachine.
* This feature require search head clustering to be enabled.
* Any consensus replication among search heads use this feature.
* Default: true

**replicate_search_peers** = <boolean>
* Add/remove search-server request is applied on all members of a search head cluster, when this value to set to true.
* Require a healthy search head cluster with a captain.

[watchdog]

**disabled** = <boolean>
* Enables thread monitoring functionality.
* Any thread that has been blocked for more than 'responseTimeout' milliseconds is logged to $SPLUNK_HOME/var/log/watchdog/watchdog.log
* Default: false.

**responseTimeout** = <decimal>
* Maximum time, in seconds, that a thread can take to respond before the watchdog logs a 'thread blocked' incident.
* The minimum value for 'responseTimeout' is 0.1.
* If you set 'responseTimeout' to lower than 0.1, the setting uses the minimum value instead.
* Default: 8

**actions** = <actions_list>
* A comma-separated list of actions that execute sequentially when a blocked thread is encountered.
* Currently, the only available actions are 'pstacks', 'script' and 'bulletin'.
* 'pstacks' enables call stack generation for a blocked thread.
* Call stack generation gives the user immediate information on the potential bottleneck or deadlock.
* The watchdog saves each call stack in a separate file in $SPLUNK_HOME/var/log/watchdog with the following file name format: wd_stack_<pid>_<thread_name>_%Y_%m_%d_%H_%M_%S.%f_<uid>.log.
* 'script' executes specified script.
'bulletin' shows a message on the web interface.

* NOTE: This setting should be used only during troubleshooting, and if you have been asked to set it by a Splunk Support engineer. It might degrade performance by increasing CPU and disk usage.
  * Default: empty list (no action executed)

actionsInterval = <decimal>
* The timeout, in seconds, that the watchdog uses while tracing a blocked thread. The watchdog executes each action every 'actionsInterval' seconds.
  * The minimum value for 'actionsInterval' is 0.01.
  * If you set 'actionsInterval' to lower than 0.01, the setting uses the minimum value instead.
  * NOTE: Very small timeout may have impact performance by increasing CPU usage. Splunk may be also slowed down by frequently executed action.
  * Default: 1

pstacksEndpoint = <boolean>
* Enables pstacks endpoint at /services/server/pstacks
  * Endpoint allows ad-hoc pstacks generation of all running threads.
  * This setting is ignored if 'watchdog' is not enabled.
  * NOTE: This setting should be used only during troubleshooting and only if you have been explicitly asked to set it by a Splunk Support engineer.
  * Default: true

[watchdog:timeouts]
reaperThread = <decimal>
* Maximum time, in seconds, that a reaper thread can take to respond before the watchdog logs a 'thread blocked' incident.
  * The minimum value for 'reaperThread' is 0.1.
  * If you set 'reaperThread' to lower than 0.1, the setting uses the minimum value instead.
  * This value is used only for threads dedicated to clean up dispatch directories and search artifacts.
  * Default: 30

[watchdogaction:pstacks]
* Setting under this stanza are ignored if 'pstacks' is not enabled in the 'actions' list.
  * NOTE: Change these settings only during troubleshooting, and if you have been asked to set it by a Splunk Support engineer. It can affect performance by increasing CPU and disk usage.

dumpAllThreads = <boolean>
* Determines whether or not the watchdog saves stacks of all monitored threads when it encounters a blocked thread.
  * If you set 'dumpAllThreads' to true, the watchdog generates call stacks for all threads, regardless of thread state.
  * Default: true

stacksBufferSizeOrder = <unsigned integer>
* Controls the maximum number of call stacks an internal queue can hold.
  * The watchdog uses the internal queue to temporarily store a call stack between the time the watchdog generates the call stack and the time it saves the call stack to a file.
  * Increase the value of this setting if you see gaps in stack files due to high frequency of call stack generation. This might occur when, for example, you set 'stacksBufferSizeOrder' to a very low value, or if the number of threads is high.
  * This number must be in the range 1 to 16.
  * The watchdog uses this value to calculate the real size of the buffer, whose value must be a power of 2. For example, if 'stacksBufferSizeOrder' is 4, the size of the buffer is 4 ^ 2, or 16.
* CAUTION: Setting to too low a value can cause dropped call stacks, and too high a value can cause increased memory consumption.
* Default: 14

maxStacksPerBlock = <unsigned integer>
* Maximum number of stacks that the watchdog can generate for a blocked thread.
* If you set 'dumpAllThreads' to true, the watchdog generates call stacks for all threads.
* If the blocked thread starts responding again, the count of stacks that the watchdog has generated resets to zero.
* If another thread blockage occurs, the watchdog begins generating stacks again, up to 'maxStacksPerBlock' stacks.
* When set to 0, an unlimited number of stacks will be generated.
* Default: 60

batchStacksThreshold = <unsigned integer>|auto
* The timeout, in milliseconds, after which the watchdog generates a new call stack file.
* This setting controls the batching up of call stacks when saving them to files, and can decrease the number of files the watchdog creates.
* When set to 0, batching is disabled.
* When set to 'auto', Splunk Enterprise determines the best frequency to create new call stack files.
* Default: auto

[watchdogaction:script]
* Setting under this stanza are ignored if 'script' is not enabled in the 'actions' list.
* NOTE: Change these settings only during troubleshooting, and if you have been asked to set it by a Splunk Support engineer. It can affect performance by increasing CPU and disk usage.

path = <string>
* The path to the script to execute when the watchdog triggers the action.
* If you do not set 'path', the watchdog ignores the action.
* No default.

useShell = <boolean>
* If set to true, the script runs from the OS shell ("/bin/sh -c" on UNIX, "cmd.exe /c" on Windows)
* If set to false, the program will be run directly without attempting to expand shell metacharacters.
* Default: false

forceStop = <boolean>
* Whether or not the watchdog forcefully stops an active watchdog action script when a blocked thread starts to respond.
* Use this setting when, for example, the watchdog script has internal logic that controls its lifetime and must run without interruption.
* Default: false

forceStopOnShutdown = <boolean>
* If you set this setting to "true", the watchdog forcefully stops active watchdog scripts upon receipt of a shutdown request.
* Default: true

Parallel Reduce Configuration

[parallelreduce]
pass4SymmKey = <password>
* Security key shared between reducers and regular indexers.
* The same value must also be specified on all intermediaries.
* Unencrypted passwords must not begin with "$1\$", as this is used by Splunk software to determine if the password is already encrypted.

```bash
@([bucket_catalog_service]
@uri = <uri>
@* Points to the tenant bucket catalog service.
@* Required.
@* Currently, only HTTP is supported by the service.
@* Example: <scheme>://<hostname>:<port>/<tenantId>/<bucket_catalog_path>
)

@([cache_manager_service]
@uri = <uri>
@* Points to the cache manager service.
@* Required.
@ ping_enabled = <boolean>
@* Currently not supported. This setting is related to a feature that is still under development.
@* Enables "ping" keep-alive transactions to the Cache Manager Service.
@* Default: true
@ timeout.ping = <unsigned integer>
@* Currently not supported. This setting is related to a feature that is still under development.
@* Sets the ping timeout, in milliseconds, to use when interacting with the Cache Manager Service.
@* Default: 30000
@ timeout.connect = <unsigned integer>
@* Currently not supported. This setting is related to a feature that is still under development.
@* Sets the connection timeout, in milliseconds, to use when connecting to the Cache Manager Service.
@* Default: 5000
@ timeout.read = <unsigned integer>
@* Currently not supported. This setting is related to a feature that is still under development.
@* Sets the read timeout, in milliseconds, to use when interacting with the Cache Manager Service.
@* Default: 60000
@ timeout.write = <unsigned integer>
@* Currently not supported. This setting is related to a feature that is still under development.
@* Sets the write timeout, in milliseconds, to use when interacting with the Cache Manager Service.
@* Default: 60000
```
**Remote Storage of Search Artifacts Configuration**

[search_artifact_remote_storage]
disabled = <boolean>
* Currently not supported. This setting is related to a feature that is
  still under development.
* Optional.
* Specifies whether or not search artifacts should be stored remotely.
  * Splunkd does not clean up artifacts from remote storage. Set up cleanup
    separately with the remote storage provider.
  * Default: true

path = <path on server>
* The path attribute points to the remote storage location where
  artifacts reside.
* The format for this attribute is: <scheme>://<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".
    For example, "path=s3://mybucket/some/path".
  * This is a required setting. If you do not set the path, the search artifact
    remote storage feature is disabled.
  * No default.

**S3 specific settings**

remote.s3.header.<http-method-name>.<header-field-name> = <String>
* Optional.
* Enable server-specific features, such as reduced redundancy, encryption,
  and so on, by passing extra HTTP headers with the REST requests.
* The <http-method-name> can be any valid HTTP method. For example, GET,
  PUT, or ALL, for setting the header field for all HTTP methods.
* Example: remote.s3.header.PUT.x-amz-storage-class = REDUCED_REDUNDANCY

remote.s3.access_key = <String>
* Optional.
* Specifies the access key to use when authenticating with the remote storage
  system supporting the S3 API.
* If not specified, the indexer looks for these environment variables:
  AWS_ACCESS_KEY_ID or AWS_ACCESS_KEY (in that order).
* If the environment variables are not set and the indexer is running on EC2,
  the indexer attempts to use the access key from the IAM role.
* No default.

remote.s3.secret_key = <String>
* Optional.
* Specifies the secret key to use when authenticating with the remote storage
  system supporting the S3 API.
* If not specified, the indexer looks for these environment variables:
  AWS_SECRET_ACCESS_KEY or AWS_SECRET_KEY (in that order).
* If the environment variables are not set and the indexer is running on EC2,
  the indexer attempts to use the secret key from the IAM role.
* No default.

remote.s3.list_objects_version = v1|v2
* The AWS S3 Get Bucket (List Objects) Version to use.
* See AWS S3 documentation "GET Bucket (List Objects) Version 2" for details.
* Default: v1

remote.s3.signature_version = v2|v4
* Optional.
* The signature version to use when authenticating with the remote storage system supporting the S3 API.
* For 'sse-kms' server-side encryption scheme, you must use signature_version=v4.
* Default: v4

remote.s3.auth_region = <String>
* Optional.
* The authentication region to use for signing requests when interacting with the remote storage system supporting the S3 API.
* Used with v4 signatures only.
* If unset and the endpoint (either automatically constructed or explicitly set with remote.s3.endpoint setting) uses an AWS URL (for example, https://s3-us-west-1.amazonaws.com), the instance attempts to extract the value from the endpoint URL (for example, "us-west-1"). See the description for the remote.s3.endpoint setting.
* If unset and an authentication region cannot be determined, the request will be signed with an empty region value.
* No default.

remote.s3.use_delimiter = true | false
* Optional.
* Specifies whether a delimiter (currently "guidSplunk") should be used to list the objects that are present on the remote storage.
* A delimiter groups objects that have the same delimiter value so that the listing process can be more efficient as it does not need to report similar objects.
* Default: true

remote.s3.supports_versioning = true | false
* Optional.
* Specifies whether the remote storage supports versioning.
* Versioning is a means of keeping multiple variants of an object in the same bucket on the remote storage.
* Default: true

remote.s3.endpoint = <URL>
* Optional.
* The URL of the remote storage system supporting the S3 API.
* The scheme, 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 is 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.multipart_download.part_size = <unsigned integer>
* Optional.
* Sets the download size of parts during a multipart download.
* This setting uses HTTP/1.1 Range Requests (RFC 7233) to improve throughput overall and for retransmission of failed transfers.
* A value of 0 disables downloading in multiple parts, i.e., files are always downloaded as a single (large) part.
* Do not change this value unless that value has been proven to improve throughput.
* Minimum value: 5242880 (5 MB)
* Default: 134217728 (128 MB)

remote.s3.multipart_upload.part_size = <unsigned integer>
* Optional.
* Sets the upload size of parts during a multipart upload.
* Minimum value: 5242880 (5 MB)
* Default: 134217728 (128 MB)

remote.s3.multipart_max_connections = <unsigned integer>
* Specifies the maximum number of HTTP connections to have in progress for
  either multipart download or upload.
* A value of 0 means unlimited.
* Default: 8

remote.s3.retry_policy = max_count
* Sets the retry policy to use for remote file operations.
* Optional.
* A retry policy specifies whether and how to retry file operations that fail
  for those failures that might be intermittent.
* Retry policies:
  + "max_count": Imposes a maximum number of times a file operation is
    retried upon intermittent failure both for individual parts of a multipart
    download or upload and for files as a whole.
* Default: max_count

remote.s3.max_count.max_retries_per_part = <unsigned integer>
* When the remote.s3.retry_policy setting is max_count, sets the maximum number
  of times a file operation is retried upon intermittent failure.
* Optional.
* The count is maintained separately for each file part in a multipart download
  or upload.
* Default: 9

remote.s3.max_count.max_retries_in_total = <unsigned integer>
* Optional.
* When the remote.s3.retry_policy setting is max_count, sets the maximum number
  of times a file operation is retried upon intermittent failure.
* The count is maintained for each file as a whole.
* Default: 128

remote.s3.timeout.connect = <unsigned integer>
* Optional
* Set the connection timeout, in milliseconds, to use when interacting with
  S3 for this volume.
* Default: 5000 (5 seconds)

remote.s3.timeout.read = <unsigned integer>
* Optional
* Set the read timeout, in milliseconds, to use when interacting with S3
  for this volume.
* Default: 60000 (60 seconds)

remote.s3.timeout.write = <unsigned integer>
* Optional
* Set the write timeout, in milliseconds, to use when interacting with S3
  for this volume.
* Default: 60000 (60 seconds)

remote.s3.sslVerifyServerCert = <boolean>
* Optional.
* If this is set to true, Splunk verifies certificate presented by S3
  server and checks that the common name/alternate name matches the
ones specified in 'remote.s3.sslCommonNameToCheck' and 'remote.s3.sslAltNameToCheck'.
* Default: false

remote.s3.sslVersions = <versions_list>
* Optional.
  * Comma-separated list of SSL versions to connect to 'remote.s3.endpoint'.
  * 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 a version is prefixed with "-" it is removed from the list.
  * SSLv2 is always disabled; "-ssl2" is accepted in the version list but does nothing.
  * When configured in FIPS mode, ssl3 is always disabled regardless of this configuration.
  * Default: tls1.2

remote.s3.sslCommonNameToCheck = <commonName1>, <commonName2>, ..
* If this value is set, and 'remote.s3.sslVerifyServerCert' is set to true, splunkd checks the common name of the certificate presented by the remote server (specified in 'remote.s3.endpoint') against this list of common names.
* No default.

remote.s3.sslAltNameToCheck = <alternateName1>, <alternateName2>, ..
* If this value is set, and 'remote.s3.sslVerifyServerCert' is set to true, splunkd checks the alternate name(s) of the certificate presented by the remote server (specified in 'remote.s3.endpoint') against this list of subject alternate names.
* No default.

remote.s3.sslRootCAPath = <path>
* Optional
  * Full path to the Certificate Authority (CA) certificate PEM format file containing one or more certificates concatenated together. S3 certificate is validated against the CAs present in this file.
  * Default: [sslConfig/caCertFile] in the server.conf file

remote.s3.cipherSuite = <cipher suite string>
* Optional.
  * If set, uses the specified cipher string for the SSL connection.
  * If not set, uses the default cipher string.
  * Must specify 'dhFile' to enable any Diffie-Hellman ciphers.
  * Default: TLSv1+HIGH:TLSv1.2+HIGH:0STRENGTH

remote.s3.ecdhCurves = <comma separated list of ec curves>
* Optional
  * ECDH curves to use for ECDH key negotiation.
  * The curves should be specified in the order of preference.
  * The client sends these curves as a part of Client Hello.
  * Splunk software only supports named curves specified by their SHORT names.
  * The list of valid named curves by their short/long names can be obtained by executing this command:
    $SPLUNK_HOME/bin/splunk cmd openssl ecparam -list_curves
  * e.g. ecdhCurves = prime256v1,secp384r1,secp521r1
  * No default.

remote.s3.dhFile = <path>
* Optional
  * PEM format Diffie-Hellman parameter file name.
  * DH group size should be no less than 2048bits.
This file is required in order to enable any Diffie-Hellman ciphers.
* No default.

remote.s3.encryption = sse-s3 | sse-kms | sse-c | none
* Optional
* Specifies the scheme to use for Server-side Encryption (SSE) for data-at-rest.
* none: no Server-side encryption enabled. Data is stored unencrypted on the remote storage.
* Default: none

remote.s3.encryption.sse-c.key_type = kms
* Optional
* Determines the mechanism Splunk uses to generate the key for sending over to S3 for SSE-C.
* The only valid value is 'kms', indicating AWS KMS service.
* One must specify required KMS settings: e.g. remote.s3.kms.key_id for Splunk to start up while using SSE-C.
* Default: kms

remote.s3.encryption.sse-c.key_refresh_interval = <unsigned integer>
* Optional
* Specifies the period, in seconds, at which a new key is generated and used for encrypting any new data being uploaded to S3.
* Default: 86400 (24 hours)

remote.s3.kms.key_id = <string>
* Required if remote.s3.encryption = sse-c | sse-kms
* Specifies the identifier for Customer Master Key (CMK) on KMS. It can be the unique key ID or the Amazon Resource Name (ARN) of the CMK or the alias name or ARN of an alias that refers to the CMK.
* Examples:
  Unique key ID: 1234abcd-12ab-34cd-56ef-1234567890ab
  CMK ARN: arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
  Alias name: aliases/ExampleAlias
* No default.

remote.s3.kms.access_key = <string>
* Optional.
* Similar to 'remote.s3.access_key'.
* If not specified, KMS access uses 'remote.s3.access_key'.
* No default.

remote.s3.kms.secret_key = <string>
* Optional.
* Similar to 'remote.s3.secret_key'.
* If not specified, KMS access uses 'remote.s3.secret_key'.
* No default.

remote.s3.kms.auth_region = <string>
* Required if 'remote.s3.auth_region' is not set and Splunk can not automatically extract this information.
* Similar to 'remote.s3.auth_region'.
* If not specified, KMS access uses 'remote.s3.auth_region'.
* No default.

remote.s3.kms.max_concurrent_requests = <unsigned integer>
* Optional.
* Limits maximum concurrent requests to KMS from this Splunk instance.
* NOTE: Can severely affect search performance if set to very low value.
* Default: 10

remote.s3.kms.<ssl_settings> = <...>
* Optional.
* Check the descriptions of the SSL settings for remote.s3.<ssl_settings>
  above. e.g. remote.s3.sslVerifyServerCert.
* Valid ssl_settings are sslVerifyServerCert, sslVersions, sslRootCAPath,
  sslAltNameToCheck, sslCommonNameToCheck, cipherSuite, ecdhCurves and dhFile.
* All of these are optional and fall back to same defaults as
  the 'remote.s3.<ssl_settings>'.

**server.conf.example**

```
# Version 8.0.0
#
# This file contains an example server.conf. Use this file to configure SSL
# and HTTP server options.
#
# To use one or more of these configurations, copy the configuration block
# into server.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
#
# Allow users 8 hours before they time out
[general]
  sessionTimeout=8h
  pass4SymmKey = changeme
#
# Listen on IPv6 in addition to IPv4...
  listenOnIPv6 = yes
# ...but make all outgoing TCP connections on IPv4 exclusively
  connectUsingIpVersion = 4-only
#
# Turn on SSL:
[sslConfig]
  enableSplunkdSSL = true
  useClientSSLCompression = true
  serverCert = $SPLUNK_HOME/etc/auth/server.pem
  sslPassword = password
  sslRootCAPath = $SPLUNK_HOME/etc/auth/cacert.pem
  certCreateScript = genMyServerCert.sh
[proxyConfig]
  http_proxy = http://proxy:80
  https_proxy = http://proxy:80
  proxy_rules = *
  no_proxy = localhost, 127.0.0.1, ::1

######## SSO Example ########
# This example trusts all logins from the splunk web server and localhost
# Note that a proxy to the splunk web server should exist to enforce
# authentication
[general]
```
trustedIP = 127.0.0.1

###### Cascading Replication Example ######
[cascading_replication]
pass4SymmKey = someSecret
max_replication_threads = auto
max_replication_jobs = 5
cascade_replication_plan_reap_interval = 1h
cascade_replication_plan_age = 8h
cascade_replication_plan_fanout = auto
cascade_replication_plan_topology = size_balanced
cascade_replication_plan_select_policy = random

# Set this node to be a cluster master.

[clustering]
mode = master
replication_factor = 3
pass4SymmKey = someSecret
search_factor = 2

# Set this node to be a slave to cluster master "SplunkMaster01" on port # 8089.

[clustering]
mode = slave
master_uri = https://SplunkMaster01.example.com:8089
pass4SymmKey = someSecret

# Set this node to be a searchhead to cluster master "SplunkMaster01" on # port 8089.

[clustering]
mode = searchhead
master_uri = https://SplunkMaster01.example.com:8089
pass4SymmKey = someSecret

# Set this node to be a searchhead to multiple cluster masters - # "SplunkMaster01" with pass4SymmKey set to 'someSecret and "SplunkMaster02" # with no pass4SymmKey set here.

[clustering]
mode = searchhead
master_uri = clustermaster:east, clustermaster:west

[clustermaster:east]
master_uri=https://SplunkMaster01.example.com:8089
pass4SymmKey=someSecret

[clustermaster:west]
master_uri=https://SplunkMaster02.example.com:8089
# Open an additional non-SSL HTTP REST port, bound to the localhost
# interface (and therefore not accessible from outside the machine) Local
# REST clients like the CLI can use this to avoid SSL overhead when not
# sending data across the network.

[httpServerListener:127.0.0.1:8090]
ssl = false

[dfs]
disabled = false
dfc_ip_address = 192.0.2.0
port = 9000
extra_kryo_registered_classes = com.splunk.df.search.compute.objects._String,java.lang.Number
spark_master_host = 192.0.2.0
spark_master_webui_port = 8080
connection_timeout = 180
connection_retries = 5