# Table of Contents

## About securing Splunk Enterprise
- About securing Splunk software ........................... 1
- How to secure and harden your Splunk software installation .................. 1

## Install Splunk securely
- Install Splunk Enterprise securely.................. 3
- Create secure administrator credentials .......... 4
- About TLS encryption and cipher suites ........ 6
- Securing Splunk Enterprise with FIPS ............. 9
- About default certificate authentication ....... 11
- Harden your Windows installation .................. 11
- Secure Splunk Enterprise on your network ....... 12
- Disable unnecessary Splunk Enterprise components .. 13
- Secure your service accounts ....................... 13
- Deploy secure passwords across multiple servers .. 13
- Harden your KV store port ........................... 14
- Some best practices for your servers and operating system .................. 14

## Manage your passwords
- Password best practices for administrators .......... 16
- Configure Splunk password policies ................. 17
- Configure a Splunk password policy in Authentication.conf .......... 18
- Password best practices for users .................. 20
- Unlock a user password ................................ 20
- Change a password ..................................... 21
- Manage out of sync passwords in a Search Head Cluster .................. 21

## Users and role-based access control
- Use access control to secure Splunk data ............. 22
- About user authentication ............................... 22
- About configuring role-based user access ........ 23
- Define roles on the Splunk platform with capabilities .......... 25
- Add and edit users ...................................... 31
- Create and manage roles with Splunk Web .......... 31
- Add and edit roles with authorize.conf ............ 37
- Setting access to manager consoles and apps ....... 40
- Find existing users and roles ........................ 42
- Delete all user accounts ................................ 42
- Secure access for Splunk knowledge objects .... 43
- Use Access Control Lists .............................. 43

## Native Splunk Enterprise authentication
- Set up Splunk authentication .......................... 45
- Configure users with Splunk Web ........................ 45
- Configure users with the CLI .......................... 46
- Add a user to a role with Splunk Web .................. 46
# Table of Contents

**Authentication using single sign-on with SAML**
- Configure advanced settings for SSO................................................................. 109
- Map SAML groups to roles.............................................................................. 109
- Modify or remove role mappings................................................................. 110
- Configure SAML SSO in the configuration files........................................... 110
- SAML SSO best practices............................................................................... 114
- Troubleshoot SAML SSO.............................................................................. 114

**Authentication using Proxy SSO**.......................................................................................................................... 119
- About ProxySSO............................................................................................ 119
- Configure ProxySSO...................................................................................... 119
- Troubleshoot Proxy SSO............................................................................... 120

**Authentication using single sign-on with reverse proxy**......................................................... 122
- About Single Sign-On using reverse proxy.................................................. 122
- Configure Single Sign-On with reverse proxy.......................................... 124
- Troubleshoot reverse-proxy SSO.............................................................. 126

**Scripted authentication**.................................................................................. 127
- Set up user authentication with external systems..................................... 127
- Create the authentication script.............................................................. 128
- Edit authentication.conf............................................................................. 130
- Use PAM authentication............................................................................ 131
- Use the getSearchFilter function to filter at search time...................... 132

**Securing Splunk Enterprise communications with SSL**........................................... 133
- About securing Splunk Enterprise with SSL............................................. 133
- About using SSL tools on Windows and Linux........................................ 135
- Configure allowed and restricted SSL versions..................................... 136

**Certificates for Splunk**.................................................................................. 138
- About creating certificates for Splunk....................................................... 138
- Things to know about your certificates................................................. 138
- About cipher suites and TLS encryption.............................................. 139
- How to prepare your signed certificates for Splunk authentication.......... 141
- Determine your cipher suite.................................................................... 143
- Working with multiple intermediate certificates.................................... 144

**Secure browser to Splunk Web communication**................................................ 145
- About securing Splunk Web................................................................. 145
- Turn on encryption (https) with Splunk Web........................................ 146
- Turn on encryption (https) using web.conf........................................... 146
- Secure Splunk Web with your own certificate........................................ 147
- Troubleshoot your Splunk Web authentication..................................... 147
About securing Splunk Enterprise

About securing Splunk software

During and after your Splunk Enterprise installation, you must take steps to secure both your configuration and your data. Securing your Splunk Enterprise installation reduces its attack surface and mitigates the risk and impact of most vulnerabilities. Some procedures are simple, such as confirming that your servers are physically secure and that your properly manage your credentials. Others, such as configuring encryption, are more complex, but are equally as important to the integrity of your data.

Read this manual to learn about all of the areas of security that you should consider:

- Secure installation of Splunk Enterprise
- Management of users and role-based access control using your chosen form of authentication
- Usage of certificates to secure indexers, forwarders, and Splunk Web, where data is most vulnerable
- Usage of encryption to secure your configuration information
- Usage of auditing to keep track of activity in your system.

Use the How to secure and harden your Splunk software installation as a checklist and roadmap to ensure that you make your configuration and data as secure as possible.

How to secure and harden your Splunk software installation

Use this checklist as a roadmap for this manual to help you secure your Splunk Enterprise configuration and protect your data.

Set up authenticated users and manage user access

- Administrator credentials provide unrestricted access to a Splunk Enterprise instance and should be the first thing you change and secure. See Secure your Admin password.
- Access control lists prevent unauthorized user access to your Splunk Enterprise instance. See Use Access Control Lists.
- Set up users and configure roles and capabilities to control user access. See About configuring role-based user access.
- Configure user authentication with one of the following methods:
  - The built-in authentication scheme. See Set up user authentication with Splunk's built-in system.
  - Splunk Enterprise authentication tokens, which are based on the native authentication scheme. Tokens let you provide access to the instance through web requests to Splunk Enterprise Representational State Transfer (REST) endpoints. See Set up authentication with tokens.
  - The Lightweight Directory Access Protocol (LDAP) authentication scheme. See Set up user authentication with LDAP.
  - A scripted authentication API for use with an external authentication system, such as Pluggable Authentication Modules (PAM) or Remote Access Dial-In User Server (RADIUS). See Set up user authentication with external systems.
- Use one of the following to create secure one-step login, or single sign-on (SSO), for users:
  - Single Sign on with SAML
  - Multi-factor authentication
  - ProxySSO
Use certificates and encryption to secure communications for your Splunk Enterprise configuration

Splunk Enterprise comes with a set of default certificates and keys that demonstrate encryption. Where possible, deploy your own certificates and configure them to secure Splunk Enterprise communications. See About securing Splunk with SSL.

Harden your Splunk Enterprise instances to reduce vulnerability and risk

- Secure communication within indexer clusters and search head clusters. See Secure your indexer clusters and search head clusters.
- Ensure that credentials in a distributed deployment are consistent across individual instances. See Deploy secure passwords across multiple servers.
- Confirm that the credentials and access levels for the accounts that run Splunk Enterprise are secure. See Secure your service accounts.
- Where possible, limit access to the app key value store network port on any Splunk Enterprise instances. See Harden your KV store port.
- Disable automatic chart recovery in the analytics workspace. See Charts in the Splunk Analytics Workspace Using the Splunk Analytics Workspace manual.

Audit your system regularly

Audit events provide information about what has changed in your Splunk Enterprise configuration. It gives you the where and when, as well as the identity of the actor who implemented the change. Leveraging audit events provides better security and other benefits.

- Audit your system regularly to monitor user and administrator access, as well as other activities that could tip you off to unsafe practices or security breaches.
- Keep an eye on activities within Splunk Enterprise, such as searches or configuration changes. You can use this information for compliance reporting, troubleshooting, and attribution during incidence response.
- Audit events are especially useful in distributed Splunk Enterprise configurations for detecting configuration and access control changes across many Splunk Enterprise instances. To learn more, see Audit Splunk Enterprise activity.
- Use the file system-based monitoring available out of the box on most Splunk-supported operating systems. For more information about monitoring, see Monitor Files and Directories in the Getting Data In Manual.
Install Splunk securely

Install Splunk Enterprise securely

To install Splunk Enterprise securely, you must have an installation package that you have confirmed is authentic and has not been modified in any way since Splunk created it. Splunk provides a Message Digest 5 (MD5) secure hash for every package it generates. You can download this hash to quickly verify that the package you downloaded is authentic and has not been changed since its creation.

You can also compare the Secure Hash Algorithm-512 (SHA-512) hashes by opening a case with Splunk Support.

Prerequisites for verifying package integrity

You must have the following to verify the contents of packages you download:

- The `md5sum` program, which prints the hash of the file that you supply, and comes with most versions of Linux. On Windows, you can use the `certutil` tool to verify MD5 hashes.
- Alternatively, the `sha512sum` program prints SHA512 hashes for the file that you supply.
- The MD5 or SHA512 hashes, in text format, which Splunk provide
- Access to a shell prompt

Verify installation package integrity

After you download the Splunk Enterprise package, verify it by using a trusted version of the OpenSSL suite to compare the MD5 or SHA-512 hashes to the hash of the installation package. If the hash of the package you downloaded matches the hash that Splunk provides, then you have downloaded a valid, secure installation package and can proceed with installation.

Download Splunk Enterprise installation package and MD5 hash

Confirm that you download the MD5 hash that exactly matches the version of installation package that you downloaded. Downloading a different file results in the hashes not matching.

1. Go to the Splunk.com download page.
2. Click Splunk Enterprise.
3. Click the tab for the operating system that you want to download Splunk software.
4. Click the Download Now link for the OS version and installation package type that you want to install with.
5. On the next page that loads, read the Splunk Software License Agreement.
6. Click the I have read, understood, and hereby accept the Splunk Software License Agreement checkbox.
7. Click Start your download now. The page refreshes and the download begins.
8. On the next page that loads, in the Useful tools box, click MD5 to verify. A second file, the MD5 hash, begins to download.
9. After both downloads finish, complete the "Verify hashes" procedure.

Download Splunk Enterprise installation package and request SHA512 hash from Splunk Support

1. Complete Steps 1 through 7 of the "Download Splunk Enterprise installation package and MD5 hash" procedure.
2. Open a case with Splunk Support to receive the SHA512 hash. When you open the case, provide a link to the version, operating system, and type of installation package you downloaded.
3. After you receive a link to the hash, follow the link to download it.
4. After the package and SHA512 hash downloads finish, complete the “Verify hashes” procedure.

**Verify hashes**

After you download the package, verify it by running either the `md5sum` or `sha512sum` utilities:

1. Open a shell prompt.
2. Change to the directory where you downloaded the installation package and the MD5 hash.
3. Print the contents of the hash file that you downloaded:

<table>
<thead>
<tr>
<th>MD5</th>
<th>SHA512</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat splunk-xxxx-release.tgz.md5</td>
<td>cat splunk-xxxx-release.tgz.sha512</td>
</tr>
</tbody>
</table>

4. Run the `md5sum` or `sha512sum` tool on the installation package that you downloaded:

<table>
<thead>
<tr>
<th>MD5</th>
<th>SHA512</th>
</tr>
</thead>
<tbody>
<tr>
<td>md5sum splunk-xxxx-release.tgz.md5</td>
<td>sha512sum splunk-xxxx-release.tgz.sha512</td>
</tr>
</tbody>
</table>

5. Compare the output from the MD5 or SHA512 hash file against the result from the `md5sum` or `sha512sum` utilities.
6. If the hashes match exactly, then the package you downloaded is authentic and has not been modified. You can continue with installation.
7. If the hash does not match, then either the package you downloaded has been modified. Retry the download.

**Verify Signatures**

You can verify the authenticity of the downloaded RPM package using the Splunk GnuPG Public key as follows:

1. Download the GnuPG Public key file (yes, this link is over TLS).
2. Install the GnuPG public key:

   ```bash
   rpm --import <filename>
   ```
3. Verify the package signature using:

   ```bash
   rpm -K <filename>
   ```

**Proceed with installation from your authenticated installation package**

After you have successfully verified your installation package as authentic, you can proceed with installation.

- Installation instructions in the *Installation Manual*

**Create secure administrator credentials**

When you install Splunk Enterprise, you must create a username and password for your administrator account. If you do not specify any arguments when you install the software, it prompts you to create a username and a password during the installation process.

If you do not create the password during installation, an unusable installation can occur. This can happen, for example, if you use the `--no-prompt` Splunk CLI argument for starting Splunk Enterprise and also do not provide an administrator password in `user-seed.conf`. In such a case, you must create the administrator credentials manually for the instance to
If you upgrade from an older version of Splunk Enterprise, the installation uses the old administrator credentials.

**Create admin credentials after starting Splunk Enterprise**

If you installed Splunk Enterprise and did not create the administrator credentials, you can use one of the following methods to create the credentials.

**Create admin credentials with user-seed.conf**

This is currently the most secure method to create administrative credentials. Other methods can introduce security risks, mainly around access to command line history or process output.

1. Edit the `$SPLUNK_HOME/etc/system/local/user-seed.conf` file as follows:

   ```
   [user_info]
   USERNAME = admin
   PASSWORD = <your password>
   ```

2. Restart Splunk Enterprise.

**Create admin credentials using REST**

Administrators with access to the machine file system can create a user and enter a password using the `splunkd rest --noauth` command.

This method is not secure because the password appears in plain text in the command line history unless you immediately delete the history after running the command.

You must restart Splunk Enterprise after using `splunkd REST` commands.

```
$ splunk cmd splunkd rest
--noauth POST /services/authentication/users
"name=admin&password=<your password>&roles=admin"
```

**Create admin credentials using the --seed-passwd or --gen-and-print-passwd CLI arguments**

This method of creating the credentials is not secure because the password appears in the command line history, process output (`ps aux`), and other items. Splunk Enterprise does not prompt you to create an administrator username in these cases, and instead uses the default of `admin`.

- Create a password when you start Splunk Enterprise with the `--seed-passwd` argument:

  ```
splunk start --accept-license
  --answer-yes --no-prompt --seed-passwd <your password>
  ```

- Generate a random password and print the random password immediately:

  ```
splunk start --accept-license
  --answer-yes --no-prompt --gen-and-print-passwd
  ```
Create admin credentials for automated installations with the 'hash-passwd' CLI command

You can use this method in automated installations where you save and distribute `user-seed.conf` to other instances. In most cases, you place `user-seed.conf` in the `$SPLUNK_HOME/etc/system/local` directory on these instances.

This method is secure as long as you delete the command line history after completing the procedure.

1. Create a hash from a plain-text password.

   splunk hash-passwd <plaintext password>

2. Copy the hash and place it into the `user-seed.conf` file. For example:

   $ splunk hash-passwd <your password>
   $6$hf3syG/qxy6RloBp...
   You can then safely write the output of the hash-passwd command in `user-seed.conf`.

   For example:

   [user_info]
   USERNAME = admin
   HASHED_PASSWORD = $6$hf3syG/qxy6RloBp...

3. To validate a password and make sure it conforms to the password complexity requirements, you can use the `splunk validate-passwd` CLI command. For example:

   splunk validate-passwd <your password>
   cat passwd.txt | splunkd validate-passwd -
   $ splunk validate-passwd weakpas
   ERROR: Password did not meet complexity requirements. Password must contain at least:
   * 8 total printable ASCII character(s).

Reset a lost password

If you lose or forget the admin password, you can reset it. You must have the ability to write to the underlying password file ($SPLUNK_HOME/etc/passwd).

   splunk cmd splunkd rest --noauth POST /services/admin/users/admin "password=<your password>"
   You must restart Splunk Enterprise after making this change.

About TLS encryption and cipher suites

As of version 6.6., Splunk provides the following default cipher suites and TLS encryption.

`alert_actions.conf`

    sslVersions = tls1.2
ecdhCurves = prime256v1, secp384r1, secp521r1
This configuration does not support Windows Server 2008 R2. To add support for Windows Server 2008 R2:

1. Set `sslVersions` to `tls`

2. Add the following ciphers to the end of the existing `cipherSuite`:

   ```
   ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:
   ECDHE-ECDSA-AES128-SHA:
   ECDHE-RSA-AES128-SHA
   ```

   To enable TLS 1.2 support on Windows Server 2008 R2:

1. Add key to the registry:

   ```
   HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Server
   ```

2. In the `TLS 1.2\Server` key, create the following:

   ```
   DWORD (32-bit) Value ? DisabledByDefault; set to 0
   DWORD (32-bit) Value ? Enabled; set to 1
   ```

3. Restart Windows


**inputs.conf**

```conf
sslVersions = tls1.2
cipherSuite = ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:
            ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:
            ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256
ecdhCurves = prime256v1, secp384r1, secp521r1
```

This configuration does not support Splunk 5.x. To add support for Splunk 5.x:

1. Set `sslVersions` to `tls`

2. Add the following ciphers to the end of the existing `cipherSuite`:

   ```
   DHE-RSA-AES256-SHA:AES256-SHA:DHE-RSA-AES128-SHA:AES128-SHA:
   AES256-SHA:AES128-SHA
   ```

**outputs.conf**

```conf
sslVersions = tls1.2
cipherSuite = ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:
            ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:
            ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256
ecdhCurves = prime256v1, secp384r1, secp521r1
```

This configuration does not support Splunk 5.x. To add support for Splunk 5.x:
1. Set `sslVersions` to `tls`.

2. Add the following ciphers to the end of the existing `cipherSuite`:

   ```
   DHE-RSA-AES256-SHA:AES256-SHA:DHE-RSA-AES128-SHA:
   AES256-SHA:AES128-SHA
   ```

   **server.conf**

   ```
   sslVersions = tls1.2
   sslVersionsForClient = tls1.2
   cipherSuite = ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:
   ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:
   ecdhCurves = prime256v1, secp384r1, secp521r1
   ```

   This configuration does not support Splunk 5.x. To add support for Splunk 5.x:

   1. Set `sslVersions` = `tls`
   2. Set `sslVersionsForClient` = `tls`
   3. Append `AES256-SHA` to the existing `cipherSuite`.

   **applicationsManagement**

   ```
   sslVersions = tls1.2
   cipherSuite = ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:
   ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:
   ecdhCurves = prime256v1, secp384r1, secp521r1
   ```

   This configuration does not support Splunk 5.x. To add support for Splunk 5.x:

   1. Set `sslVersions` = `tls`

   2. Add the following ciphers to the end of the existing `cipherSuite`:

   ```
   DHE-RSA-AES256-SHA:AES256-SHA:DHE-RSA-AES128-SHA:
   AES256-SHA:AES128-SHA
   ```

   **web.conf**

   ```
   sslVersions = tls1.2
   cipherSuite = ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:
   ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:
   ecdhCurves = prime256v1, secp384r1, secp521r1
   ```
This configuration does not support Windows Vista. To add support for Windows Vista:

1. Set `sslVersions = tls`

2. Add the following ciphers to the existing cipherSuite:

```
ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA:
ECDHE-RSA-AES128-SHA
```

`ldap.conf`

```
TLS_PROTOCOL_MIN: 3.1 for TLSv1.0, 3.2 for TLSv1.1, 3.3 for TLSv1.2.

TLS_PROTOCOL_MIN 3.3

TLS_CIPHER_SUITE ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-GCM-
SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-
AES128-SHA256:ECDHE-RSA-AES128-SHA256
```

This configuration does not support Windows Server 2008 R2. To add support for Windows Server 2008 R2:

1. Set `TLS_PROTOCOL_MIN = TLS1.0/SSL3.1`

2. Add the following ciphers to the existing `TLS_CIPHER_SUITE`:

```
ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA:
ECDHE-RSA-AES128-SHA
```

To enable TLS 1.2 support on Windows Server 2008 R2:

1. Add key:

```plaintext
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Server
```

2. In the `TLS 1.2\Server` key, create the following:

```plaintext
DWORD (32-bit) Value ? DisabledByDefault; set to 0
DWORD (32-bit) Value ? Enabled; set to 1
```


**Securing Splunk Enterprise with FIPS**

The Federal Information Processing Standard (FIPS) uses government-certified versions of some algorithms to meet regulatory guidelines. It should not be considered a security enhancement by itself, and might potentially reduce performance on your system. Enable FIPS if it is a regulatory requirement for your environment.

Splunk Enterprise and the Universal Forwarder use an embedded FIPS 140-2-validated cryptographic module (Certificate #3126 Module Version fips-2.0.12) running on various platforms per FIPS 140-2 Implementation Guidance section G.5 guidelines.
The certificate is listed on the National Institute of Standards and Technology (NIST) site. See Certificate Detail | Cryptographic Module Validation Program
See the consolidated validation certificate also on the NIST website.

Key points to enabling FIPS

There are several things that you must understand when you enable FIPS on Splunk Enterprise:

- You must enable FIPS mode before you start Splunk Enterprise. FIPS mode is disabled except when it runs on a Linux machine that runs a kernel in FIPS mode.
- FIPS is automatically enabled if you run Splunk software on a Linux machine that runs a kernel in FIPS mode.
- The FIPS module disables the use of some cryptographic algorithms in the instance of Python that Splunk software uses to run apps (such as Message Digest 5 (MD5) and Rivest Cipher 4 (RC4)).
- Any Splunk apps you want to run on a FIPS-enabled instance must be certified to run in FIPS mode and cannot have dependencies on algorithms like MD5 or RC4.

Enable FIPS

Always enable FIPS mode upon initial Splunk software installation. If you install the software without FIPS mode enabled, you cannot later upgrade it to a FIPS version, and must either reinstall, or install a new version.

1. Before you start Splunk Enterprise for the first time, use a text editor to edit the $SPLUNK_HOME/etc/splunk-launch.conf configuration file.
2. Add the following line to the file:
   SPLUNK_FIPS=1
3. Start Splunk software. It enables FIPS mode during the installation.

Use indexes with FIPS enabled

Running Splunk in FIPS mode does not alter indexed data in any way. You can copy indexes between FIPS and non-FIPS indexers.

Confirm FIPS mode status

You can use the CLI, a REST endpoint, or Splunk search to determine whether or not the Splunk instance is in FIPS mode.

The following CLI command returns FIPS mode status:

```
splunk show fips-mode -auth <username>:<password>
```
- If FIPS mode is enabled, the CLI command returns **FIPS mode enabled**.
- Otherwise, it returns **FIPS mode disabled**.

The following REST call returns FIPS mode status:

```
curl -s -k -u admin:changeme https://localhost:8089/services/server/info | grep fips_mode
```
If FIPS mode is enabled, this call returns the following:

```
<s:key name="fips_mode">1</s:key>
```
The following Splunk search returns FIPS mode status:

```
| rest splunk_server=local /services/server/info | fields fips_mode
```

If FIPS mode is enabled, the search returns the following:

```
fips_mode
--------
1
```

**Troubleshoot FIPS**

- If you are in FIPS mode and your usual RSA encrypted private keys do not work, they might be incompatible with FIPS. To mitigate this issue, you can convert your Privacy Enhanced Mail (PEM) private key to PKCS#8 format to make them compatible.

- After you install Splunk software without FIPS mode enabled, you cannot enable FIPS mode. If you require FIPS compliance, confirm that your initial Splunk installation is FIPS-enabled. To change to a version running FIPS mode, reinstall Splunk software and use the procedure in this topic to enable FIPS.

- If you have problems running a Splunk app, confirm that it is certified to run in FIPS mode and does not have dependencies on cryptographic algorithms that FIPS disables (such as MD5 and RC4).

**About default certificate authentication**

Splunk Enterprise 6.6 and higher comes with default certificates that are signed with Secure Hash Algorithm (SHA)-256 using a 2048-bit key. These certificates are part of a fresh installation.

When you upgrade from a previous release, Splunk Enterprise replaces the existing `cacert.pem.default` and `ca.pem.default` Privacy Enhanced Mail (PEM) files. Existing certificates, for example `cacert.pem` and `ca.pem`, are not affected.

Because of the new default PEM, you must upgrade all certificates and PEM files to SHA-256 using a 2048-bit key to avoid validation errors. For example, your indexers and forwarders might require updates to meet the same standards as your Splunk Enterprise instance. You might also want to check the certificate for your license manager. If all certificates and PEM files are not updated, Splunk Enterprise logs the following error in `splunkd.log` when it attempts to connect to another instance over SSL:

```
ERROR TcpOutputFd - Connection to host=10.140.130.102:9997 failed. sock_error = 0.
SSL Error - error:04091077:rsa routines:INT_RSA_VERIFY:wrong signature length
```

**Harden your Windows installation**

If you choose a directory that is outside of the **Program Files** directory on the drive that booted your Windows machine, that directory does not get the automatic protections that directories inside **Program Files** get.

Follow these instructions to ensure that Windows enforces the proper access controls to the Splunk software installation directory and that low-privilege users cannot access that folder.
Harden Splunk software installation directories with Windows Explorer

1. From an Explorer window, right click on the target installation folder and select **Security > Advanced > Disable Inheritance > Remove all inherited permissions from this object.**
2. Click **Add > Select a principle > Administrator > Check Names.**
3. Click **OK.**
4. Click **Check full control.**
5. Click **OK.**
6. Click **Add > Select a principle > SYSTEM > Check Names.**
7. Click **OK.**
8. Click **Check full control.**
9. Click **OK.**
10. Click **Apply**, then click **OK.**

Harden Splunk software installation directories from a PowerShell window or command prompt

These instructions were tested and work as described on Windows versions 8.1 and higher, and Window Server versions 2008 R2 and 2012 R2. As an example, this procedure uses `C:\Splunk` as the installation directory.

1. Open a command prompt or PowerShell window.
2. Run the following command to break inheritance from parent directories.
   
   ```
   C:\>icacls C:\Splunk /inheritance:d
   ```
3. Run the following command to remove the Users group from the directory.
   
   ```
   C:\>icacls C:\Splunk /remove "Users" /T
   ```
4. Run the following command to remove the “Authenticated Users” group from the directory.
   
   ```
   C:\>icacls C:\Splunk /remove "Authenticated Users" /T
   ```

Secure Splunk Enterprise on your network

Under certain conditions, Splunk Enterprise ports can become susceptible to attacks. Prevent access by shielding your Splunk Enterprise configuration from the Internet.

If possible, use a host-based firewall to restrict access to Splunkweb, management, and data ports. Keep Splunk Enterprise within a host-based firewall. Have your remote users access Splunk Enterprise on a Virtual Private Network.

You also can protect Splunk Enterprise from attacks in the following ways:

- Restrict CLI security by restricting this port to local calls only, from behind a host firewall.
- Unless necessary, do not allow access to forwarders on any port.
- Install Splunk Enterprise on an isolated network segment that only trustworthy machines can access.
- Limit port accessibility to only necessary connections. The necessary connections are:
  - End users and administrators must access Splunkweb (TCP port 8000 by default).
  - Search heads must access search peers on the management port (TCP port 8089 by default).
  - Deployment clients must access the deployment server on the management port (TCP port 8089 by default).
  - Forwarders must access the index server data port (TCP port 9997 by default).
  - Remote CLI calls use the management port.
• Restrict access to the KV store port on the search head. (The KV store port, by default, is 8191, and by default that port is open to the network.) On each search head cluster member, allow access to the KV store port only for the other members, so that the cluster can replicate KV store data.

Disable unnecessary Splunk Enterprise components

For single-server Splunk Enterprise deployments:

• Forwarders should not run Splunkweb and should not be configured to receive data on TCP or UDP ports or from other Splunk Enterprise instances.

For multiserver Splunk Enterprise deployments:

• Search heads should not receive data on TCP or UDP ports or from other Splunk Enterprise instances.
• If users are not logging in to Splunkweb on indexers in a distributed environment, Splunkweb should be disabled on the indexers.

Secure your service accounts

Practice the principle of least privilege by running Splunk software as an unprivileged user rather than using a privileged account such as root or Administrator.

• On Unix or Linux, use the "splunk" user created with the PKG or RPM packages, or create your own user that only has privilege and ownership over $SPLUNK_HOME.
• On Windows, the local system context is often the best choice. However, if you require communication using a windows communication channel, such as WMI, use a restricted access account.

Deploy secure passwords across multiple servers

At initial startup, Splunk Enterprise creates a file $SPLUNK_HOME/etc/auth/splunk.secret. This file contains a key used to encrypt some of your authentication information in configuration files. Each of the following files can be encrypted across a deployment using splunk.secret. Note that the passwords and encryption methods used for each file are not necessarily interchangeable.

• web.conf: Your SSL passwords on every instance.
• authentication.conf: Your LDAP passwords, if you have any.
• inputs.conf: Your SSL passwords, if you use splunktcp-ssl to set up ssl for data distribution.
• outputs.conf: Your SSL passwords, if you use splunktcp-ssl to configure splunktcp-ssl in inputs.conf to set up ssl for data distribution.
• server.conf: pass4symmkey, if you have one.
• passwords.conf: Your password for a given app.

When Splunk software starts, if it detects a clear-text password, in one of these settings, it will create or overwrite the configuration in the equivalent local folder with the encrypted password.

Note: If the pass4symmkey or SSLPassword is specified in a default apps file, the password is obfuscated in the local version of the file upon restart. The default version of the file remains in clear text. However, if the file is listed using curl
or a splunkd endpoint, the passwords appear encrypted.

When you deploy Splunk software on multiple servers, you must encrypt the passwords and ensure that they are consistent across your deployment. Splunk recommends that you use an encryption tool such as Vault's "secret/*" to secure your passwords as you distribute them.

You should perform these steps at initial deployment and also any time you need to deploy a new password for your instances:

1. Using your encryption tool, deploy the passwords to all servers.
2. On each server, place the password, in clear text, in the relevant file.
3. Immediately start/restart to encrypt all the passwords under the server's unique secret.

In a search head cluster, the captain replicates its `splunk.secret` file to all other cluster members during initial deployment of the cluster, so you do not need to copy it manually. As part of its normal operation, the cluster also automatically replicates any credentials that are stored by apps for their own use.

**Harden your KV store port**

We recommend that you secure your environment by restricting KV store access to your port. By default, port 8191 is opened to the network. We recommend that you restrict this port when possible.

For search head clustering, you should open the port only to other members of the cluster so that other members can replicate KV store data.

For more information about working with KV store, see About the app key value store

**Some best practices for your servers and operating system**

**Operating System**

To maximize security, harden the operating system on all computers where you run Splunk software.

- If your organization does not have internal hardening standards, consult the CIS hardening benchmarks.
- As a minimum, limit shell/command line access to your Splunk servers.

**Splunk**

- Configure redundant Splunk instances, both indexing a copy of the same data.
- Backup Splunk data and configurations, regularly.
- Execute a periodic recovery test by attempting to restore Splunk Enterprise from backup.
- Verify your Splunk download using a hash function such as MD5 to compare the hashes. For example:

  `./openssl dgst md5 <filename-splunk-downloaded.zip>`
Client browser

- Use a current version of a supported browser, such as Firefox or Internet Explorer.
- Use a client-side JavaScript blocker such as noscript on Firefox or Internet Explorer 8 Filters to help protect against XSS, XSRF, and similar exploits.
- Ensure that users have the latest Flash version installed.

Physical security

- Secure physical access to all Splunk servers.
- Ensure that Splunk end users practice sound physical and endpoint security.
  - Set a short time-out for Splunk Web user sessions. See Configure timeouts for more information.

More opportunities to secure your configuration

- Use a configuration management tool, such as subversion, to provide version control for Splunk configurations.
- Integrate Splunk configuration changes into your existing change management framework.
- Configure Splunk Enterprise to monitor its own configuration files and alert on changes.
Manage your passwords

Password best practices for administrators

Passwords are one of the most common points of unauthorized access in successful security breaches.

The National Institute of Standards and Technology (NIST) proposes significant changes to the rules governing passwords, upending many of the classic ways to prevent weak passwords. Splunk provides the traditional means to drive more secure passwords and continues to monitor and update security requirements based on customer feedback.

Splunk Enterprise provides configuration options for the following password characteristics:

- Length
- Complexity
- Expiration and password reuse (history)
- Lockout after failed attempts

To set up a password policy, see the following topics:

- Configure Splunk password policies
- Configure a Splunk password policy in Authentication.conf

Password policy best practices

The following best practices help significantly lower the chances of unauthorized access to Splunk and other systems:

- Require a longer password length

  Long passwords or passphrases are the best way to secure access. Using 10 or more characters prevents the use of rainbow tables and dramatically inhibits password-cracking software that decodes password hashes. A password is more secure when it requires more characters. The required minimum is 8 characters and you can set a length up to 256 characters.

- Require a complex password

  Traditional guidance requires the use of multiple character types in a password, such as both lowercase and uppercase letters, numbers, and special characters. This helps prevent dictionary words in passwords and increases resistance to brute-force attacks.

Because complex passwords can be difficult to remember and do not add significantly to the strength of passwords, NIST no longer suggests this requirement. However, Splunk Enterprise supplies this option because it is still commonly part of regulatory compliance for many organizations.

- Use Password expiration (off by default)

  Traditional guidance is to force users to change passwords four or more times per year. Many compliance regulations require this value be set to 90 days or less. NIST guidance no longer recommends this, because users tend to change the last character in their password. As a result, any compromised passwords can be used to guess new passwords. Set the maximum password age to any value from 0 to 3650 days, where a value of 0
means that passwords do not expire.

- Enforce Password History (off by default)

Enable password history to ensure that a user's most recent passwords are saved. Any password saved to a user's history cannot be used again. With this policy, you discourage users from alternating between several common passwords.

- Use password lockout (enabled by default)

Password lockout temporarily locks users out of the system after a certain number of failed attempts. Password lockout is on by default and occurs five attempts. You can specify how many attempts the user has, and how long the lockout period must last. The default is 30 minutes. In less secure environments, you can set the lockout time for longer to be sure that an unverified user cannot access the system.

Configure Splunk password policies

Use the Password Policy Management page in Splunk Web to create a password policy for your users. Password policies set standards and minimum requirements for complexity.

This task applies to Splunk Enterprise native authentication and does not apply to SAML or LDAP passwords.

1. In Splunk Web, click **Settings > Access Controls > Password Policy Management**.
2. In the **Minimum characters** field, specify the minimum number of characters to require for user passwords. The maximum number of characters Splunk software supports is 256. The default value is 8.
3. In the **Numeral** field, specify the number of digits to require for user passwords. A best practice is to require at least one number and to not allow passwords that are all numbers. The default is 0.
4. In the **Lowercase** field, specify the number of lowercase letters to require for user passwords. A best practice is to require at least one lowercase letter. The default is 0.
5. In the **Uppercase** field, specify the number of uppercase letters to require for user passwords. A best practice is to require at least one uppercase letter. The default is 0.
6. In the **Special character** field, specify the number of special characters to require for user passwords. A best practice is to require at least one special character. A user can create a password with any printable ASCII characters. The default is 0.
7. Check **Force existing users to change weak passwords** to make existing users upgrade passwords to meet the requirements specified on this page.
8. Enable **Expiration** to force a user to change their password after the specified period of time.
9. In the **Days until password expires** field, specify the number of days until the user must change their password.
10. In the **Expiration alert in days** field, specify the number of days before expiration that warnings appear.
11. Enable **Lockout** to lock a user out of the system after a certain number of failed login attempts.
12. In the **Failed login attempts** field, specify how many failed login attempts a user can make before they are locked out. The default is 5.
13. In the **Lockout threshold in minutes** field, specify the number of minutes between the time of the first failed login until the failed login attempt counter resets.
14. In the **Lockout duration in minutes** field, specify how many minutes the user must wait before they can attempt to log in again. The default value is 30 minutes.
15. Enable **History** to prevent users from reusing previous passwords. Note that if you disable this value and enable it later, previously saved password history is preserved. Delete `$SPLUNK_HOME/etc/opasswd` to remove the password history.
16. In the **Password History Count** field, specify the number of previous passwords that may not be reused. The default is 24.
17. Click **Save**.

Your new password requirements are applied to the **Set Password** field in the Create User page.

**Configure a Splunk password policy in Authentication.conf**

Edit `authentication.conf` to create a password policy for your Splunk software users. Password policies set standards and minimum requirements for complexity. The policy you create applies to Splunk Enterprise native authentication. This policy does not apply to SAML or LDAP passwords.

To configure a password policy for Splunk software users, edit the following settings in the `[splunk_auth]` stanza in `authentication.conf`.

- `minPasswordLength = <positive integer>`
- `minPasswordUppercase = <positive integer>`
- `minPasswordLowercase = <positive integer>`
- `minPasswordSpecial = <positive integer>`
- `minPasswordDigit = <positive integer>`
- `expirePasswordDays = <positive integer>`
- `expireAlertDays = <positive integer>`
- `expireUserAccounts = <boolean>`
- `forceWeakPasswordChange = <boolean>`
- `lockoutUsers = <boolean>`
- `lockoutMins = <positive integer>`
- `lockoutAttempts = <positive integer>`
- `lockoutThresholdMins = <positive integer>`
- `enablePasswordHistory = <positive integer>`
- `passwordHistoryCount = <positive integer>`

The following table describes the details of each setting:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
<th>Recommended value</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>minPasswordLength</code></td>
<td>Specify the minimum ASCII-based alphanumeric or special characters required when a user creates a new password. Splunk Enterprise supports values between 1 and 256 characters.</td>
<td>At least 8 characters</td>
<td>8 characters</td>
</tr>
<tr>
<td><code>minPasswordUppercase</code></td>
<td>The number of uppercase letters required for user passwords.</td>
<td>At least one uppercase letter</td>
<td>Zero uppercase characters</td>
</tr>
<tr>
<td><code>minPasswordLowercase</code></td>
<td>The number of lowercase letters required for user passwords.</td>
<td>At least one lowercase letter</td>
<td>Zero lowercase characters</td>
</tr>
<tr>
<td><code>minPasswordSpecial</code></td>
<td>The number of special characters required for user passwords. When this setting is populated, a user can enter any special character except a semicolon.</td>
<td>At least one special character</td>
<td>Zero special characters</td>
</tr>
<tr>
<td><code>minPasswordDigit</code></td>
<td>The number of numeric characters required for user passwords. Value can be 0 through the length of the password. The total allowed length of</td>
<td>At least one number</td>
<td>Zero numbers</td>
</tr>
<tr>
<td>Setting</td>
<td>Description</td>
<td>Recommended value</td>
<td>Default value</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>expirePasswordDays</td>
<td>The number of days before the password expires. Allowed values are 0 to 3650 days.</td>
<td>90 days</td>
<td>90 days</td>
</tr>
<tr>
<td>expireAlertDays</td>
<td>The number of days in advance of password expiration that Splunk software issues user alerts. Allowed values are 0 to 120 days.</td>
<td>15 days</td>
<td>15 days</td>
</tr>
<tr>
<td>expireUserAccounts</td>
<td>Set to true to enable password expiration.</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>forceWeakPasswordChange</td>
<td>Set to true to force users to change a weak password the next time they log in.</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>lockoutUsers</td>
<td>Specifies whether the system locks users out after a number of failed login attempts. When this setting is enabled, user lockout is local to each search head peer in a search cluster configuration and is not replicated to other peers.</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>lockoutMins</td>
<td>The number of minutes that a user must wait before attempting to log in again after a lockout. The user is locked out after entering an incorrect password more times than specified in the lockoutAttempts value in the lockoutThresholdMins setting. Supported values are 1 - 1440 minutes.</td>
<td>At least 30 minutes</td>
<td>30 minutes</td>
</tr>
<tr>
<td>lockoutAttempts</td>
<td>The number of unsuccessful logins a user can attempt before they are locked out. The unsuccessful login attempts must occur within lockoutThresholdMins minutes. Supported values are 1 - 64 login attempts.</td>
<td>At least 5 attempts</td>
<td>5 attempts</td>
</tr>
<tr>
<td>lockoutThresholdMins</td>
<td>The number of minutes that must pass from the time of the first failed login before the failed login attempt counter resets. Supported values are 1 - 120 minutes.</td>
<td>At least 5 minutes</td>
<td>5 minutes</td>
</tr>
<tr>
<td>enablePasswordHistory</td>
<td>Set to true to keep a user's password history. If you disable this value and enable it later, previously saved password history is preserved. Delete $SPLUNK_HOME/etc/opasswd to remove the password history.</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>passwordHistoryCount</td>
<td>The number of passwords that are stored in history. If you enable password history and provide a number such as 25, then the user is not able to reuse their previous 25 passwords. Supported values are 1 - 128 stored passwords.</td>
<td>24 stored passwords</td>
<td>24 stored passwords</td>
</tr>
</tbody>
</table>
Password best practices for users

A few steps can help you create strong passwords that protect you and your system. Keep the following best practices in mind when creating a new password in Splunk Enterprise.

Tips for creating strong passwords

• Create unique passwords with a combination of words, numbers, symbols, and both lowercase and capitalized letters.
• Consider groups of words that form a phrase or sentence, such as the opening sentence of your favorite novel or the opening line to a good joke. The ideal password could be an obscure, random phrase that is easy for you to remember, but impossible for an automated system to understand.
• Make your password as long as your system allows. It is increasingly easy to build password-cracking tools that can try hundreds of billions of possible password combinations per second. Each character you add to a password or passphrase increases immunity to brute-force methods.

Avoid the following insecure practices

• Do not choose passwords based on personal information, such as your birth date, your Social Security or phone number, or names of family members.
• Do not use a word from the dictionary. Password-cracking tools are freely available online often come with dictionary lists that will try thousands of common names and passwords. Try using multiple words, adding a numeral to the words, and adding well as punctuation at the beginning or end of the word (or both).
• Never use the same password for different sites.
• Never use the password you've picked for your email account at any online site.
• Do not store your list of passwords on your computer in plain text.

Unlock a user password

If your user locks themselves out of their account or forgets their password, you can unlock their account.

To change a user password, see Change a password.

Unlock a user account in Splunk Web

If a user or admin is locked out an admin can:

• Wait for the lockout period to expire (not recommended for lengthy lockout periods.)
• Manually reset the user that has locked the peer.

1. In Splunk Web click Settings > Access Control > Users.

2. In the Users page, check the Status column to locate the user that is locked.

3. In the Action column for that user, click Unlock.
Unlock a user from the command line

A Splunk user with privilege to write to disk on a Splunk instance can execute this command.

In the command line, type the following CLI command:

```
splunk edit user <locked username> -locked-out false -auth admin:<yourpassword>
```

About unlocking users in distributed environments

If a user on a search head cluster is locked out, they are locked out on a single member of the cluster. Results from other search heads will not show the user as locked out.

If a user or admin is locked out, an admin can:

- Wait for the user’s lockout period to expire
- Unlock the user, using the instructions on this page.

Change a password

This topic describes how to change a user password. To unlock a user password, see Unlock a user password.

Change a user password in Splunk Web

When resetting a password, use the following task.

1. In Splunk Web click Settings > Access Controls > Users.
2. In the Users page, select the user you want to change.
3. Type a new password for the user. Distribute this password to your user.
4. Click Save.

Change a user password in the command line

A Splunk admin user with privilege to write to disk on a Splunk instance can execute this command.

In the command line, type the following CLI command:

```
splunk edit user <username> -auth admin:<admin_password> -password <password>
```

Manage out of sync passwords in a Search Head Cluster

If passwords become out of sync in your cluster, you can force replication to sync your passwords. See Use the deployer to distribute apps and configuration updates.
Users and role-based access control

Use access control to secure Splunk data

Role-based access control provides flexible and effective tools that you can use to protect Splunk data.

Splunk Enterprise masks data to the user much like the way a relational database manages role-based access control. In some cases total segmentation of data may be necessary. In other cases, controlling the searches and results at the presentation layer (something you can do with many of our Splunk Apps) may meet your security needs.

Consider your use cases when deciding how to set up your configurations and whether role-based access might fit your needs. For example:

- For extremely sensitive data, where even allowing access to a system that might have sensitive data incurs legal risk, consider installing and configuring more than one instance of Splunk Enterprise, and then configuring each instance with the data for the appropriate audience.

- When intentionally or unintentionally exposing sensitive data to the wrong user might incur legal ramifications, then consider creating indexes specifically for privileged and non-privileged accounts and assigning them to roles created for each level of access.

- When there are security concerns but not so much legal risk, you can restrict access using Apps. For example, you can create an App with static dashboards and assign roles with lower clearance to those dashboards, limiting the type of information the user assigned to the role may access.

- Field encryption (optional feature), search exclusions, and field aliasing to redacted data are also great ways to tighten up a limited search case. If you have a limited search case and only able to search some specific data from a shared index, you can restrict shared reports to restrict ad hoc searches and funneling summary indexing into an index that is secured.

About user authentication

Splunk Enterprise authentication allows you to add users, assign them to roles, and give those roles custom permissions as needed for your organization.

Options for authentication systems are as follows:

- Splunk authentication: Provides Admin, Power and User by default, and you can define your own roles using a list of capabilities. If you have an Enterprise license, Splunk authentication is enabled by default. See Set up user authentication with Splunk's built-in system for more information.

- LDAP: Splunk Enterprise supports authentication with its internal authentication services or your existing LDAP server. See Set up user authentication with LDAP for more information.

- Scripted authentication API: Use scripted authentication to integrate Splunk authentication with an external authentication system, such as RADIUS or PAM. See Set up user authentication with external systems for more information.
Note: Authentication, including native authentication, LDAP, and scripted authentication, is not available in Splunk Free.

You can create and assign users to flexible roles either in Splunk Web or by editing authorize.conf. For more information about roles and capabilities, read About role-based user access.

Important: Splunk authentication takes precedence over any external systems. Users are authenticated in the following order:

1. Splunk authentication
2. LDAP or scripted authentication (if enabled)

About configuring role-based user access

Roles let you control access to resources on the Splunk platform. When you configure role-based user access, you determine what permissions and capabilities that users have through the roles that they hold. As users cannot receive permissions and capabilities directly, roles connect users to how they interact with the Splunk platform.

You can assign roles to users to control the scope of the tasks that they can perform, the data they can search, and the amount of resources they can access on the platform. Users can hold multiple roles, and each role gives the user specific access to resources or platform actions, as the role defines them. Roles do not take away access, so if you do not want a user to perform a certain function, then that user must not hold the role that lets them perform that function.

For more information about users, see About user authentication.

Predefined roles

The Splunk platform comes with the following predefined roles:

- **admin**: This role is intended for administrators who manage all or most of the users, objects, and configuration and comes with the most assigned capabilities.
- **power**: This role can edit all shared objects (saved searches, etc) and alerts, tag events, and other similar tasks.
- **user**: This role can create, edit, and run its own searches, save those searches, edit its own preferences, create and edit event types, and other similar tasks.
- **can_delete**: This role lets the user delete by keyword. This capability is necessary when using the delete search operator.
- **sc_admin** (Splunk Cloud only): This role lets users create other users and roles, but does not grant any other administrative capabilities.

Set permission granularity with custom roles

You can create custom roles and assign the roles to your users. Custom roles let you make granular adjustments to user access, including the following:

- **Role inheritance**: You can have the role inherit certain properties from one or more existing roles. For more information, see "Role inheritance" in this topic.
- **Capabilities**: You can specify which actions that a user that holds the role can perform, for example, change their password, change forwarder settings, and so on. See About defining roles with capabilities for more information.
- **Allowed and default indexes**: You can limit access to specific indexes and set which indexes the Splunk platform searches by default.
• **Search restrictions**: In addition to specifying the indexes that users that hold the role can search, you can also specify a search filter that limits the search results that these users can see. For additional information, see "Search restrictions" in this topic.

• **Resource access**: You can control how many standard and real-time searches that all users that hold the role can run at one time, as well as individual limits for each user. You can restrict searches to a certain time window, and control how much disk space is available for search jobs that a user with this role creates.

You can create and manage any roles, including the predefined ones, by either using Splunk Web or making edits to configuration files.

• To use Splunk Web for role management, see Add and edit roles with Splunk Web.
• To manage roles with the authorize.conf configuration file, see Add and edit roles with authorize.conf.

**Search restrictions**

In addition to controlling the indexes that a role holder can search, you can further limit what results that searches of those indexes return. The search filter combines with the base search that the user runs to determine the final data set that the user sees. The Splunk platform removes results that match the filter, which means there are fewer results than if there was no filter.

Search filters are limited to certain specific fields and operators. You can create a search filter manually by typing it in, or you can use the search filter generator to create it automatically, based on the number of indexes you select and the indexed fields and values that those indexes contain. With the search filter generator, you can create complex search filters without a need to worry about syntax. You can preview what a search with this filter applied will look like when you run it, so that you can be confident your users get the search results you expect when they use it.

See Create and manage roles with Splunk Web for information on how to set search filters and use the search filter generator.

**Role inheritance**

Roles do not take away access. Users that hold multiple roles inherit the permissions and capabilities of the role that has the broadest permissions. Roles that have more permissions supersede roles that have fewer. If you want to limit access to resources, create and assign roles that establish those limits, and do not let those roles inherit from roles that do not establish those limits.

**How users inherit search filter restrictions**

If a user holds roles with different search filters, the Splunk platform combines the filters and applies the restrictions of each search filter.

For example, the "power" and "user" roles do not define any search filters to restrict searches by default. If a user holds both these roles, and you assign another role to them that does have a defined filter, then they inherit the search restrictions that come with the third role, even though the "power" and "user" roles do not have a search filter.

**How users inherit allowed indexes**

Users that hold multiple roles receive the most permissive access that each role that they hold can provide.

For example, say you have a custom role called "simple_user" which limits access to a single index, and another custom role called "advanced_user", which has more capabilities and permits access to all indexes. If you assign both roles to the
same user, that user receives access to all indexes through the “advanced user” role, even though the “simple user” role limits access to a single index. As roles do not take away access, if you want to grant the capabilities of the “advanced_user” role while limiting index access to one index with the “simple_user” role, the best practice is to create a custom role that combines the capability and index access that you want the user to have.

**How users inherit capabilities**

Users that hold multiple roles receive the most permissive amount of capabilities that each role that they hold can provide.

For example, if you assign a user the "admin" and "advanced_user" roles, the user receives the capabilities that come with both roles, even though the "advanced_user" role might have fewer capabilities than the "admin" role.

**Define roles on the Splunk platform with capabilities**

When you create a user on the Splunk platform, you assign one or more roles to the user as part of the user creation process. Each role contains a set of **capabilities**. These capabilities define what users who hold a certain role can do.

For example, if a user ‘finn’ holds the edit_tokens_settings role, this means that ‘finn’ can make changes to the Token Authentication scheme on the instance. If they hold the admin_all_objects capability, they can make changes to any object on the instance.

You can add, edit, or remove capabilities for new, existing, and default roles. Doing this changes the kind of access that the role provides. For example, you might give a role the capability to add inputs or edit saved searches.

Capabilities are always additive in nature. There is no way to take away an ability to do something by adding a capability. If you don't want users who hold a role to perform a certain function on your Splunk platform instance, then do not assign a capability that grants the ability to perform that function to that role.

Similarly, users who hold multiple roles receive all the benefits of any capabilities that are assigned to those roles. If you do not want a certain user to have access to all the capabilities that a role provides, do not assign that role to that user.

**Add, edit, and remove capabilities from roles**

- To add or change the capabilities of a role in Splunk Web, see Create and manage roles with Splunk Web.
- To create roles and assign capabilities by editing authorize.conf, see Add and edit roles with authorize.conf.
- To learn more about roles and how they work, see About configuring role-based user access.

**Table of Splunk platform capabilities**

This list shows the capabilities that you can add to any role, and whether or not the capabilities are assigned by default to the user, power, or admin roles.

Capabilities are subject to change. For the most up-to-date list of capabilities, see the authorize.conf specification file.

For the most up-to-date list of capabilities that are assigned to a role, see the "Imported Capabilities" text box in the "Create a role" page in Splunk Web on your instance.

<table>
<thead>
<tr>
<th>Capability name</th>
<th>What it lets you do</th>
<th>User</th>
<th>Power</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerate_datamodel</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>
Enable or disable acceleration for data models. Set acceleration to true to enable automatic acceleration of this data model. Additional space is required depending on the number of events, fields, and distinct field values in the data. See the Knowledge Manager Manual for more information.

<table>
<thead>
<tr>
<th>Capability name</th>
<th>What it lets you do</th>
<th>User</th>
<th>Power</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>accelerate_search</td>
<td>Lets the user enable or disable acceleration for reports. The user must also have the schedule_search capability assigned. Works for searches that use transforming commands. See the Knowledge Manager Manual for more information.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>admin_all_objects</td>
<td>Lets the user access and modify any object in the system regardless of any restrictions set in the objects. For example user objects, search jobs, reports, and knowledge objects. Lets the user bypass any ACL restrictions, much the way root access in a *nix environment does.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>change_authentication</td>
<td>Lets the user change authentication settings and reload authentication. See the Securing Splunk Enterprise Manual for more about authentication.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>change_own_password</td>
<td>Lets the user change their own password.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>delete_by_keyword</td>
<td>Lets the user use the &quot;delete&quot; operator. The &quot;delete&quot; command marks all of the events returned by the search as deleted. This masks the data from showing up in search results but does not actually delete the raw data on disk. See the Search Manual for more information.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>delete_messages</td>
<td>Lets a user delete system messages that appear in the UI navigation bar.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>dispatch_rest_to_indexers</td>
<td>Lets a user dispatch the REST search command to indexers.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_bookmarks_mc</td>
<td>Lets a user add bookmark URLs within the Monitoring Console. The URLs redirect administrators to Monitoring Console instances in other Splunk deployments.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_deployment_client</td>
<td>Lets the user change deployment client settings. See the Managing Indexers and Clusters of Indexers Manual for more about the deployment client.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_deployment_server</td>
<td>Lets the user change deployment server settings. User can change or create remote inputs that are pushed to the forwarders and other deployment clients. See the Managing Indexers and Clusters of Indexers manual for more about the deployment server.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_dist_peer</td>
<td>Lets the user add and edit peers for distributed search. See the Managing Indexers and Clusters of Indexers Manual for more information.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_encryption_key_provider</td>
<td>Lets the user view and edit key provider properties when they use Server-Side Encryption (SSE) for a remote storage volume.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_forwarders</td>
<td>Lets the user change forwarder settings, including settings for SSL, backoff schemes, etc. Also used by TCP and Syslog output admin handlers.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_health</td>
<td>Lets a user enable/disable health reporting, set health status alerts, and set indicator thresholds for a feature in the splunkd health status tree through the server/health-config/endpoint.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_httpauths</td>
<td>Lets the user edit and end user sessions through the httpauth-tokens endpoint.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_indexer_cluster</td>
<td>Lets the user edit indexer clusters. See the Managing Indexers and Clusters of Indexers Manual for more about indexers.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_indexerdiscovery</td>
<td>Lets the user edit settings for indexer discovery, including settings for master_url, pass4SymmKey, and so on. Used by Indexer Discovery admin handlers.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

26
<table>
<thead>
<tr>
<th>Capability name</th>
<th>What it lets you do</th>
<th>User</th>
<th>Power</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>edit_input_defaults</td>
<td>Lets the user use the server settings endpoint to change default hostnames for input data.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_local_apps</td>
<td>Lets the user edit actions for application management. Applies only when you set the <code>enable_install_apps</code> setting to &quot;true&quot; in <code>authorize.conf</code>.</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>edit_metric_schema</td>
<td>Lets the user set up log-to-metrics transformations, which can convert single log events into multiple metric data points.</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>edit_metrics_rollup</td>
<td>Lets the user create and edit metrics rollup policies, which set rules for the aggregation and summarization of metrics on a specific metric index.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_monitor</td>
<td>Lets the user add inputs and edit settings for monitoring files. Also used by the standard inputs endpoint and the one-shot input endpoint.</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>edit_roles</td>
<td>Lets the user edit roles and change user/role mappings. Used by both the user and role endpoint.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_roles_grantable</td>
<td>Lets the user edit roles and change user/role mappings for a limited set of roles. Can assign any role to other users. To limit this ability, configure <code>grantableRoles</code> in <code>authorize.conf</code>. For example: <code>grantableRoles = role1;role2;role3</code></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>edit_scripted</td>
<td>Lets the user create and edit scripted inputs.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_search_concurrency_all</td>
<td>Lets a user edit settings related to maximum concurrency of searches.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_search_concurrency_scheduled</td>
<td>Lets a user edit settings related to concurrency of scheduled searches.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_search_head_clustering</td>
<td>Lets the user edit search head clustering settings.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_search_schedule_priority</td>
<td>Lets the user assign a search a higher-than-normal schedule priority. For information about the search scheduler, see the Knowledge Manager Manual.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_search_schedule_window</td>
<td>Lets the user assign schedule windows to scheduled reports. Requires the <code>schedule_search</code> capability. For more about the search scheduler, see the Knowledge Manager Manual.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>edit_search_scheduler</td>
<td>Lets the user enable and disable the search scheduler. See the Knowledge Manager Manual.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_search_server</td>
<td>Lets the user edit general distributed search settings like timeouts, heartbeats, and blacklists.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_server</td>
<td>Lets the user edit general server settings like server name, log levels, etc.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_server_crl</td>
<td>Lets the user edit general server settings like server name, log levels, etc. Inherits the ability to read general server and introspection settings.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_sourcetypes</td>
<td>Lets the user edit sourcetypes. See the Knowledge Manager manual for more information about sourcetypes.</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>edit_splunktcp</td>
<td>Lets the user change settings for receiving TCP inputs from another Splunk instance.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_splunktcp_ssl</td>
<td>Lets the user view or edit any SSL-specific settings for Splunk TCP input.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_splunktcp_token</td>
<td>Lets the user edit the Splunktcp token.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_tcp</td>
<td>Lets the user change settings for receiving general TCP inputs.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>edit_telemetry_settings</td>
<td>Lets the user edit general settings.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Capability name</td>
<td>What it lets you do</td>
<td>User</td>
<td>Power</td>
<td>Admin</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>edit_token_http</td>
<td>Lets the user create, edit, display, and remove settings for HTTP token input. Also enables the HTTP Event Collector feature.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_tokens_all</td>
<td>Lets the user issue tokens to all users.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_tokens_own</td>
<td>Lets the user issue tokens to themself.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_tokens_settings</td>
<td>Lets the user manage token settings.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_udp</td>
<td>Lets the user change settings for UDP inputs.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_user</td>
<td>Lets the user create, edit, or remove users. A role with the edit_user capability can assign any role to other users. To limit this ability, configure <code>grantableRoles</code> in <code>authorize.conf</code>. For example: <code>grantableRoles = role1;role2;role3</code>. Also lets a user manage certificates for distributed search.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_view_html</td>
<td>Lets the user create, edit, or modify HTML-based views.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_web_settings</td>
<td>Lets the user change settings for <code>web.conf</code> through the system settings endpoint.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_workload_pools</td>
<td>Lets the user create and edit workload pools through the <code>workloads/pools</code> endpoint.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_workload_rules</td>
<td>Lets the user create and edit workload rules through the <code>workloads/rules</code> endpoint.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>embed_report</td>
<td>Lets the user embed reports and disable embedding for embedded reports.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>export_results_is_visible</td>
<td>Lets the user display or hide the <em>Export Results</em> button in Splunk Web. The default value is to display the button.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>get_diag</td>
<td>Lets the user get a remote diag from a Splunk instance using the <code>/streams/diag</code> endpoint.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>get_metadata</td>
<td>Lets the user use the &quot;metadata&quot; search processor.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>get_typeahead</td>
<td>Lets the user use typeahead in the endpoint and the typeahead search field.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>indexes_edit</td>
<td>Lets the user change any index settings such as file size and memory limits.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>input_file</td>
<td>Lets the user add a file as an input through <code>inputcsv</code> (except for dispatch=t mode) and <code>inputlookup</code>.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>install_apps</td>
<td>Lets the user install, uninstall, create, and make updates to apps. Applies only when you configure the <code>enable_install_apps</code> setting to &quot;true&quot; in <code>authorize.conf</code>.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>license_edit</td>
<td>Lets the user edit the license.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>license_tab</td>
<td>Lets the user access and change the license. This attribute is deprecated.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>license_view_warnings</td>
<td>Lets the user see a warning message when they are exceeding data limits or reaching the expiration date of their license. These warnings appear on the system banner.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>list_accelerate_search</td>
<td>Lets the user view accelerated reports. User cannot accelerate reports.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>list_deployment_client</td>
<td>Lets the user view deployment client settings.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Capability name</td>
<td>What it lets you do</td>
<td>User</td>
<td>Power</td>
<td>Admin</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>list_deployment_server</td>
<td>View deployment server settings.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_forwarders</td>
<td>Lets a user list and view settings for data forwarding. Can be used by TCP and Syslog output admin handlers.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_health</td>
<td>Lets a user monitor the health of Splunk Enterprise features (such as inputs, outputs, clustering, and so on) through REST endpoints.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_httpauths</td>
<td>Lets the user view user sessions through the httpauth-tokens endpoint.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_indexer_cluster</td>
<td>Lets the user view the list of indexer clusters as well as indexer cluster objects such as buckets, peers, etc.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_indexer_discovery</td>
<td>Lets the user view settings for indexer discovery. Also used by indexer discovery handlers.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_inputs</td>
<td>Lets the user view lists of various inputs, including input from files, TCP, UDP, scripts, etc.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>list_introspection</td>
<td>Lets the user read introspection settings and statistics for indexers, search, processors, queues, etc.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_metrics_catalog</td>
<td>Lets the user query for lists of metrics catalog information such as metric names, dimensions, and dimension values.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>list_search_head_clustering</td>
<td>Lets the user list and view search head clustering objects like artifacts, delegated jobs, members, captain, etc.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_search_scheduler</td>
<td>Lets the user view lists of search scheduler jobs.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_settings</td>
<td>Lets the user list and view server and introspection settings such as the server name, log levels, etc.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_storage_passwords</td>
<td>Lets the user list and view the /storage/passwords endpoint, lets the user perform GETs. The admin_all_objects capability must be added to the role for the user to perform POSTs to the /storage/passwords endpoint.</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_tokens_all</td>
<td>Lets the user view all tokens.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_tokens_own</td>
<td>Lets the user view their own tokens.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>list_workload_pools</td>
<td>Lets a user list and view workload pool and workload status information from the workloads/rules endpoint.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>list_workload_rules</td>
<td>Lets a user list and view workload rule information from the workloads/rules endpoint.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>metric_alerts</td>
<td>Lets a user create, update, enable, disable, and delete a streaming metric alert.</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>never_expire</td>
<td>Lets a user account never expire.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>never_lockout</td>
<td>Lets a user account never lock the user out.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>output_file</td>
<td>Lets the user create file outputs, including outputcsv (except for dispatch=t mode) and outputlookup.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>pattern_detect</td>
<td>Lets the user see and use the Patterns tab in the Search view.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>request_remote_tok</td>
<td>Lets the user obtain a remote authentication token, which lets the user perform some distributed peer management and bundle replication and distribute searches to old 4.0.x Splunk instances.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>rest_apps_management</td>
<td>Lets the user edit settings for entries and categories in the python remote apps handler. See restmap.conf for more information.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Capability name</td>
<td>What it lets you do</td>
<td>User</td>
<td>Power</td>
<td>Admin</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>rest_apps_view</td>
<td>Lets the user list and view various properties in the Python remote apps handler. See restmap.conf for more information.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>rest_properties_get</td>
<td>Lets the user get information from the services/properties endpoint.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>rest_properties_set</td>
<td>Lets the user edit the services/properties endpoint.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>restart_splunkd</td>
<td>Lets the user restart Splunk Enterprise through the server control handler.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rtsearch</td>
<td>Lets the user run real-time searches.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>run_collect</td>
<td>Lets the user run the collect command.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>run_mcollect</td>
<td>Lets the user run the mcollect and meventcollect commands.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>run_msearch</td>
<td>Lets the user run the msearch command.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>run_multi_phased_searches</td>
<td>Lets the user run searches with the redistribute command, which invokes parallel reduce search processing in distributed search environments. This capability is not assigned to any role by default.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>schedule_rtsearch</td>
<td>Lets the user schedule real-time saved searches. The schedule_search capability must also be assigned to the role.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>schedule_search</td>
<td>Lets the user schedule saved searches, create and update alerts, and review triggered alert information.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>search</td>
<td>Lets the user run a search. See the Search Manual for more information.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>search_process_config_refresh</td>
<td>Lets the user use the &quot;refresh search-process-config&quot; CLI command to manually flush idle search processes.</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>select_workload_pools</td>
<td>Lets a user assign a scheduled search or ad-hoc search to a workload pool.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>srchFilter</td>
<td>Lets the user manage search filters. See the Search Manual for more information.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>srchIndexesAllowed</td>
<td>Lets the user run search indexes. See the Search Manual for more information.</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>srchIndexesDefault</td>
<td>Lets the user set default search indexes.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>srchJobsQuota</td>
<td>Lets the user set search job quotas.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>srchMaxTime</td>
<td>Lets the user set the maximum time for a search.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>upload_lookup_files</td>
<td>Lets the user upload files that can be used in conjunction with lookup definitions. Only affects lookup types that involve the upload of a file, such as CSV and geospatial lookups.</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>use_file_operator</td>
<td>Lets the user use the &quot;file&quot; search operator. The &quot;file&quot; search operator is deprecated.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>web_debug</td>
<td>Lets the user debug Web files.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Windows-specific capabilities**

If you are running Splunk Enterprise on Windows, additional capabilities are provided to facilitate monitoring.

<table>
<thead>
<tr>
<th>Capability name</th>
<th>What it lets you do</th>
</tr>
</thead>
<tbody>
<tr>
<td>edit_modinput_admon</td>
<td>Edit modular inputs in admon.conf.</td>
</tr>
<tr>
<td>edit_modinput_perfmon</td>
<td>Edit modular inputs in perfmon.conf.</td>
</tr>
</tbody>
</table>
## Capability name | What it lets you do
---|---
edit_modinput_winhostmon | Add and edit inputs for monitoring Windows host data.
edit_modinput_winnetmon | Add and edit inputs for monitoring Windows network data.
edit_modinput_winprintmon | Required to add and edit inputs for monitoring Windows printer data.
edit_win_admon | (Deprecated)
edit_win_eventlogs | Edit windows eventlogs.
edit_win_perfmon | (Deprecated)
edit_win_regmon | (Deprecated)
edit_win_wmiconf | Edit wmi.conf.
list_pdfserver | View PDF server files.
list_win_localavailablelogs | List all local Windows event logs.
srchTimeWin | Set search time limits.
write_pdfserver | Write to PDF server files.

### Add and edit users

To create a new user:

1. From the main menu, go to **Settings > Access Controls**.
2. Click **New User**.
3. In the **Name** field, provide a user name. This is usually the short name for the user.
4. In the **Full Name** field, provide the full name of your user.
5. In the **Email Address** field, provide the user's email address.
6. In the **Set password** field, create a password.
7. Confirm the user's new password in the **Confirm Password** field.
8. Select the user's time zone in the **Time Zone** field.
9. In the **Default App** field, select the app that the user will land in by default. The default is "Home". "Search" is a common default app as well.
10. In **Assign to Roles**, you can select any roles that you want for your user.
11. Click **Create a role for user** if you want to user's new assignments to be created as a role assigned specifically to this user.
12. Check **Require password change on first login** to force your user to immediately change their password.

### Create and manage roles with Splunk Web

You can assign roles to users that determine the level of access that those users have to the Splunk platform and the tasks that they can perform. The platform comes with a set of default roles, and you can also create your own custom roles that you can tailor to the needs of your organization.

Roles can contain one or more capabilities that provide access to specific parts of the Splunk platform. A user that has a role assigned to them receives all of the capabilities that are associated with the role. Roles can inherit capabilities from other roles.
Manage role inheritance, searched indexes, restrictions, and available search resources

When you add and edit roles, you can modify the following role properties:

- You can manage role inheritance. See "Specify role inheritance" in this topic.
- You can manage the indexes that a role has available to it as well as which indexes the Splunk platform searches by default. See "Specify searchable indexes for a role" in this topic.
- You can apply a search filter to further limit search results. You can either specify the filter manually or use the search filter generator - a wizard that lets you build and populate the filter by using indexed fields and values found in those indexes. See "Specify search restrictions for a role" in this topic.
- You can control resource usage on the platform by limiting disk space usage for search artifacts, the number of searches that the role as a whole can run, and the number of searches that users who hold the role can run individually. See "Specify default app and search limits for a role" in this topic.

While you can have any role inherit from any other role, custom roles that inherit from the admin or power users roles do not automatically inherit administrator-level access to the instance.

- For more information about roles and how capabilities and permissions are inherited, see About configuring role-based user access.
- For information about granting management access to custom roles, see Add access controls to custom roles.
- For more information about role inheritance, see Role inheritance in the About configuring role-based user access topic.
- For more information about how capabilities work, as well as the full list of capabilities, see About defining roles with capabilities.

Add or edit a role

Create or edit roles for your Splunk platform instance on the Roles page in Settings.

1. Click Settings > Roles.
2. Click New Role to create a new role, or click an existing role to edit it.
3. Enter a name for your role.
4. Make adjustments to role settings by editing configurations in any of the tabs in this dialog box.
5. After you have made the configuration changes that you want, click Save to save the role.

The only required element of a role is its name. You do not have to complete any of the following tabs to save a role.

Specify role inheritance

Use the 1. Inheritance tab to add or change the inheritance of existing roles.

1. Click 1. Inheritance to display the contents of the Inheritance tab.
2. (Optional) In the Role Name text box, type in characters to display roles whose names contain those characters.
3. (Optional) Click the All column header to select from a menu of display options for roles: "Show selected", "Show unselected", or "Show all".
4. (Optional) Click the checkbox next to an existing role from which you want this role to inherit. You can click multiple checkboxes, or select all existing roles by clicking the checkbox in the column header.
Specify role capabilities

Use the **2. Capabilities** tab to add or change the capabilities that this role holds.

1. Click **2. Capabilities** to display the contents of the Capabilities tab.
2. (Optional) In the **Capability Name** field, type in a string to display capability names that contain the string.
3. (Optional) Click the **All** column header to select from a menu of display options for capabilities: “Show native”, “Show inherited”, “Show selected”, “Show unselected”, or “Show all”.
4. Click the checkbox next to the capabilities that you want to assign to this role.
5. Click **Save**.

Capabilities that have been inherited from other roles appear as grayed out and selected. You cannot deselect capabilities that come with inherited roles.

Specify searchable indexes for a role

Use the **3. Indexes** tab to choose the indexes that the role can search, and which ones it should search by default.

You can specify both event and metric indexes. You can also specify wildcards that match more than one index. If a user with the role runs a metrics search without a specified index, the search includes results from the default metrics indexes that you assign to the role. You must select at least one index with data here if you want to be able to use the **SPL Search Filter generator** in the **4. Restrictions** tab.

Wildcards let you specify all indexes that match the text you enter. For example, if you specify a wildcard of "index_us*", it captures all existing indexes that begin with index_us. Wildcards that you create appear in the Indexes table in alphabetical order, as selected and default indexes.

You can create multiple wildcards, but they only apply to the current role. You cannot transfer wildcards to other roles; instead you must explicitly create the same wildcard by editing the roles and adding the wildcards there. To delete a wildcard from a role, confirm that the wildcard is neither a selected nor a default index, and save the role.

1. Click **3. Indexes** to display the contents of the Indexes tab.
2. (Optional) In the **Wildcards** section, enter a string that contains the * character and specifies the group of indexes you want to search, then click **Create**.

You can repeat this action to add more wildcards. If a wildcard already exists, Splunk Web advises you.

3. (Optional) In the **Index Name** field, type in a string to display index names that begin with that string.
4. (Optional) Click the **All** column header to select from a menu of display options for indexes: “Show native”, “Show inherited”, “Show selected”, “Show unselected”, or “Show all”.
5. Click the **Included** checkbox for an index to include search results from that index for this role.
6. Click the **Default** checkbox for an index to include search results from that index when a user that holds this role does not specify an index in their search.

Indexes from inherited roles appear as grayed out and selected. You cannot deselect indexes that come with inherited roles.

Specify search restrictions for a role

Use the **4. Restrictions** tab to limit the scope of search results that return when users with the role run searches. The search filter combines with the base search that users with the role run, based on several factors. The search job returns only the results that arise from the combined search.
For more information on valid syntax to use with the search filter, see "SPL search filter syntax" later in this topic.

1. Click 4. Restrictions to display the contents of the Restrictions tab.
2. In the SPL Search filter field, type in a valid SPL string that combines with any base search that a user with this role runs.
3. (Optional) Use the Search filter SPL generator to create a search filter.
   1. In the Indexed fields and values time range drop down list, choose a time range to search for indexed fields and their associated values.

For these controls to work, you must have selected at least one index with data in the Indexes tab. Changing the default time of 60 seconds can increase the amount of time it takes to populate the Indexed Fields and Values text boxes, but might be necessary to retrieve a comprehensive list of indexed fields.
2. In the "Indexed fields" text box, do one of the following:
   1. Click on the text box to display a drop-down list box that contains the most common indexed fields that were found, based on the indexes you have selected in the 3. Indexes tab and the time that you specified in the "Indexed fields and values time range" setting. The \texttt{walklex} search command populates this field.
      1. Enter the name of an indexed field.

If you select an indexed field that is already present in the SPL search filter, Splunk Web displays a message about possible SPL collisions. Review the filter to confirm that there are no unintended conflicts.

3. In the "Values" text box, do one of the following:
   1. Click on the text box to display a drop-down list box that shows the top 250 indexed field values that were found, in lexical order, based on the fields you selected in the "Indexed fields" text box.
      1. Enter a custom field value directly. You can also use wildcards.

4. Use the Concatenation option drop-down list box to determine how the SPL generator adds SPL text that it generates to any existing text in the SPL search filter.
   1. Choose "AND" to add the generated SPL prepended with the \texttt{AND} keyword.
   2. Choose "OR" to add the generated SPL prepended with the \texttt{OR} keyword.
   3. Choose "NOT" to add the generated SPL prepended with the \texttt{NOT} keyword.

If the search filter does not have any text in it, the "Concatenation option" drop-down list box is disabled.
5. Review the SPL that the SPL generator proposes adding to the SPL search filter.
6. If you are satisfied with the SPL that has been generated, click Add to SPL search filter. The SPL generator updates the SPL search filter text box with the generated text. If there is already text in the filter text box, the SPL generator appends the generated text. Depending on the concatenation option you chose, the SPL generator adds the text after the "AND", "OR", or "NOT" keyword.
7. (Optional) If you do not like the SPL that you generated with the SPL generator, you can remove the text that you added by clicking Reset.
8. (Optional) If you want to see how the search filter can affect search results before you apply it, click Preview search filter results. This action opens a new Search page that shows the results of a search with the current search filter.

The search preview results are an example of what a user with this role might see. Several factors can alter the actual results from what the preview shows.

The preview makes the assumption that the user holds only this role. While it includes results from inherited indexes, it does not include any search filters that might exist in inherited roles.

If you have configured the Splunk platform instance so that search filters for a role eliminate, rather than select results, actual results might be the opposite of what you see in the preview. The \texttt{srchFilterSelecting} setting in authorize.conf controls whether search filters select or eliminate results,
and is true by default. A false value tells search filters to eliminate results.

**Specify default app and search-related limits for a role**

In the 5. Resources tab, you can control the default app that a user with this role sees when they log into the Splunk platform. You can also set user- and role-based limits to concurrent searches, role-based limits to search time ranges, and limits to the amount of disk space that a person with a given role can take up with their search jobs at a given time.

You can also control various search job characteristics and limits.

1. (Optional) In the Default app dropdown, select the default Splunk app that appears when a user that holds this role logs in.
2. (Optional) In the Role search job limit section, enter the maximum number of standard searches that this role can run concurrently in the Standard search job limit text box.

To remove search limits, you can enter 0 in this and other search limit text boxes.

3. (Optional) Enter the maximum number of real-time searches that a user with this role can run concurrently in the Real-time search job limit text box.

4. (Optional) In the User search job limit section, enter the maximum number of standard and real-time searches that a user can run concurrently in the Standard search job limit and Real-time search limit text boxes.

If enable_cumulative_quota is set to true in limits.conf, role search job limits override user search job limits. For example, say you set a Role search job limit of 45 standard jobs for the Admin role, and you set a User search job limit of 10 standard search jobs. Under this condition, if you have five Admin users, they will only be able to run 45 standard search jobs concurrently as a group, not 50 standard search jobs.

5. (Optional) In the Role search time window limit section, select a standard search maximum time range for this role. Click the drop-down list box to choose a value:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
<th>Can inherited roles override this setting?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unset</td>
<td>Historical searches run by this role do not have a time range limit.</td>
<td>Yes</td>
</tr>
<tr>
<td>Infinite</td>
<td>Historical searches run by this role do not have a time range limit.</td>
<td>No</td>
</tr>
<tr>
<td>Custom time</td>
<td>Exposes a text box where you can define a maximum time range in seconds for historical searches run by this role.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The Splunk platform applies custom time range limits backwards from the latest time specified for a search.

If a user has multiple roles with custom time range limits, or has roles that inherit from roles with custom time range limits, the Splunk platform applies the least restrictive search time range limits to the role. For example, say you have a user named Blue who has role A with a custom time of 30s, role B with a custom time of 60s, and role C with a custom time of 3600s. Blue would get the maximum search time range of 3600s, or 1 hour.

This setting does not apply to real-time searches.

6. (Optional) In the Disk space limit section, enter the amount of space that search jobs run by a person with this role can take up on disk at a given time in the Standard search limit text box.

**Save changes to role configurations**

You must save changes to role configurations (including search time restrictions) and restart the Splunk platform before those changes can take effect. If you do not restart, the instance cannot enforce your configurations and restrictions.

- To save all of the changes you have made and close the dialog box, click Save.
• If you do not want to save the changes, click Cancel.

If you click Cancel, you lose any unsaved changes that you have made since you opened the Roles dialog box.

For more information about restarting the Splunk platform, see Start and stop Splunk Enterprise in the Admin Manual.

**SPL search filter syntax**

The SPL search filter field in the 4. Restrictions tab accepts any of the following search terms:

- source::
- host::
- index::
- sourcetype::
- eventtype= or eventtype::
- The keywords AND, OR, or NOT
- Search fields

You can enter SPL manually into the SPL search filter text box, or use the SPL generator to create SPL for the search filter based on fields and field values that you have indexed.

You can use wildcards. Use OR to allow multiple terms, or AND to make the filter more restrictive.

**Caveats to using the SPL search filter**

The search terms cannot include any of the following:

- Saved searches
- Time operators
- Regular expressions
- The mstats, msearch, and mcatalog search commands, when you use them in conjunction with the key::value syntax
- Any fields or modifiers that you can override from the Splunk Web search bar

**Usage of search filter syntax with event and metrics data**

For event data, when you specify search term filters, use the key::value syntax, rather than key=value, where possible, to restrict search terms to indexed fields. If you specify the key=value syntax as part of a filter, the search filter dialog box warns you that usage of the - operator can result in poor search performance for users who hold the role. Also, it is not secure to use the operator because filters with the operator can be bypassed by user knowledge objects.

If you attempt to add an indexed field that already exists in the current search filter, the page warns you that the indexed field already exists and to ensure that you have no unintended SPL conflicts in the search filter.

For search filters with metrics data, use the key=value to specify search restrictions to metrics fields. This is because the key::value syntax does not work for searches over metrics data. In this case, you can safely disregard syntax warnings about the - operator that the search filter dialog box presents.
Add and edit roles with authorize.conf

You can add or modify roles by editing authorize.conf. Users are assigned to roles that determine their level of access and the tasks that they can perform. For more information about roles and capabilities, read About role-based user access.

Never edit or delete roles in $SPLUNK_HOME/etc/system/default/authorize.conf. This could break your admin capabilities. Instead edit the local version at $SPLUNK_HOME/etc/system/local/, or your own custom application directory in $SPLUNK_HOME/etc/apps/.

You must reload authentication or restart Splunk Enterprise after making changes to authorize.conf. Otherwise, your new roles will not appear in the Role list. See "Refresh the authentication scheme" in this topic for instructions on how to refresh authentication without restarting the Splunk platform.

For more information, see

- authorize.conf
- About configuration files in the Admin Manual.

Note: Distributed search configurations have slightly different authorization needs. When you use search head clustering, you must make sure that the search heads and the search peers all use the same set of authorize.conf file(s). To make sure your authorization is properly set up for search pooling, see How authorization works in distributed searches.

Add roles

Here’s the syntax for adding roles through $SPLUNK_HOME/etc/system/local/authorize.conf:

```
[role_<roleName>]
<setting> = <value>
<setting> = <value>
...
```

The <roleName> in the stanza header is the name you want to give your role. For example: security, compliance, ninja.

Role names must use lowercase characters only. They cannot contain spaces, colons, semicolons, or forward slashes.

You can include these settings in the role stanza:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Definition</th>
<th>Default</th>
<th>For more information</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;capability&gt; = enabled</td>
<td>You can add any number of capabilities to a role. To add a capability to a role, just set that capability to “enabled”.</td>
<td>disabled</td>
<td>See About defining roles with capabilities.</td>
</tr>
<tr>
<td>importRoles = &lt;role&gt;;&lt;role&gt;;...</td>
<td>When set, the current role inherits the capabilities from &lt;role&gt;. Members assigned to multiple roles inherit properties from the role with the broadest permissions. If you enter multiple roles, separate them with semicolons.</td>
<td>No roles set.</td>
<td>See Role inheritance.</td>
</tr>
<tr>
<td>srchFilter = &lt;search_string&gt;</td>
<td>This setting lets you define detailed data access controls. Users with this role will have their searches filtered by this expression.</td>
<td>No filters are set.</td>
<td>See Search filter format.</td>
</tr>
<tr>
<td>srchTimeWin = &lt;string&gt;</td>
<td>The maximum time span in seconds allowed for a search executed by a user in this role.</td>
<td>Not set. Search times</td>
<td></td>
</tr>
<tr>
<td>Setting</td>
<td>Definition</td>
<td>Default</td>
<td>For more information</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------</td>
<td>----------------------</td>
</tr>
<tr>
<td>srchDiskQuota = &lt;int&gt;</td>
<td>The maximum amount of disk space (MB) that can be used by search jobs performed by a user assigned to this role.</td>
<td>100</td>
<td>are not limited.</td>
</tr>
<tr>
<td>cumulativeSrchJobsQuota = &lt;number&gt;</td>
<td>The maximum number of concurrently running historical searches that all members of this role can have. For this setting to apply, you must also set enable_cumulative_quota = true in limits.conf. When a user belongs to multiple roles, the user uses searches from the roles with the largest cumulative search quota first. When the quota for that role is completely used up, roles with lower quotas are used.</td>
<td>Not set.</td>
<td></td>
</tr>
<tr>
<td>cumulativeRTSrchJobsQuota = &lt;number&gt;</td>
<td>The maximum number of concurrently running real-time searches that all members of this role can have. For this setting to apply, you must also set enable_cumulative_quota = true in limits.conf. If the user belongs to multiple roles, the user uses searches from the roles with the largest cumulative search quota first. When the quota for that role is completely used up, roles with lower quotas are used.</td>
<td>Not set.</td>
<td></td>
</tr>
<tr>
<td>srchJobsQuota = &lt;int&gt;</td>
<td>The maximum number of concurrently running searches a member of this role can have.</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>rtSrchJobsQuota = &lt;number&gt;</td>
<td>The maximum number of concurrently running real-time searches a member of this role can have.</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>srchIndexesDefault = &lt;string&gt;</td>
<td>A semicolon-delimited list of default indexes to search when no index is specified. The list can include both event indexes and metric indexes. When the user runs an event search that does not specify an index, the search runs over the default event indexes. When the user runs a metrics search that does not specify an index, the search runs over the default metrics indexes. You can wildcard your entries. However, the wildcard '<em>' does not match internal indexes. To match internal indexes, start with '<em>'. All internal indexes are represented by '</em></em>'.</td>
<td>Not set.</td>
<td></td>
</tr>
<tr>
<td>srchIndexesAllowed = &lt;string&gt;</td>
<td>A semicolon-delimited list of indexes this role is allowed to search. The list can include both event and metrics indexes. You can wildcard your entries. However the wildcard '<em>' will not match internal indexes. To match internal indexes, start with '<em>'. All internal indexes are represented by '</em></em>'.</td>
<td>Not set.</td>
<td></td>
</tr>
</tbody>
</table>
Search filter format

The `srchFilter` field can include any of the following search terms:

- `source=`
- `host=` and host tags
- `index=` and index names
- `eventtype=` and event type tags
- `sourcetype=`
- `search fields`
- `wildcards`
  - use `OR` to use multiple terms, or `AND` to make searches more restrictive.

The search terms cannot include:

- `saved searches`
- `time operators`
- `regular expressions`
- `any fields or modifiers Splunk Web can overwrite`

Example of creating a role in `authorize.conf`

This example creates the role "ninja", which inherits capabilities from the default "user" role. ninja has almost the same capabilities as the default "power" role, except it cannot schedule searches. In addition:

- 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 occupy up to 500MB total space on disk for all its jobs.

```
[role_ninja]
rtsearch = enabled
importRoles = user
srchFilter = host=foo
srchIndexesAllowed = *
srchIndexesDefault = mail;main
srchJobsQuota = 8
rtSrchJobsQuota = 8
srchDiskQuota = 500
```

Refresh the authentication scheme

After you make changes to `authentication.conf`, you must refresh the authentication scheme to have the changes take effect. You can do this with either Splunk Web or the CLI. Refreshing the authentication scheme does not log users off of the system.

A restart of the Splunk platform also reloads authentication, but it also logs users off of the system, and there is no requirement to restart the platform to refresh authentication.
Refresh the authentication scheme using Splunk Web

1. From the system bar, click **Settings > Authentication Methods**.
2. Click **Reload authentication configuration**. This refreshes the authentication caches but does not boot current users from the system.

Refresh the authentication scheme using the CLI

1. On the Splunk platform instance where you want to reload authentication, open a shell prompt or PowerShell window.
2. Switch to the `$SPLUNK_HOME/bin` directory.
3. Use the CLI command `./splunk reload auth`:

```
./splunk reload auth
```

The authentication scheme cache resets immediately.

Setting access to manager consoles and apps

The `local.meta` file is handy for allowing you to grant and restrict access to certain parts of your Splunk instance. For example, you can:

- Restrict users in custom roles to a specific app
- Give users in custom roles the ability to access admin level features

Granting admin roles to users

Some management abilities that belong to the Admin role are unique to that specific label. These abilities are not automatically inherited from the Admin role when you configure a role in Splunk Web or `authorize.conf`.

For example, say you want to create a custom role that inherits all of the Admin abilities but has limited access to your search jobs. To do this, you would create a new role called “SpecialAdmin” and set it to inherit all of the capabilities of an Admin as described in About defining roles with capabilities then set your search limits About configuring role-based user access.

Restricting access to specific apps

The `local.meta` file can also be used to restrict access.

For example, say you want to allow a user access to only one dashboard view. To accomplish this, you could create an app for that view and assign the user’s role to that app. You should use `local.meta` to permit the role to view that app.

How to add and remove access via local.meta files

You can give or restrict access by editing the `local.meta` file to add the new role wherever you want it.

1. Locate the `local.meta` file. If you are editing access for the main search page (ie, the manager controls), look in `$SPLUNK_HOME/etc/system/metadata/`. If you want to edit access to a particular app, look in `$SPLUNK_HOME/etc/apps/<app_name>/metadata/`. If the directory for the desired location does not contain the file, you can copy the default version `default.meta` and rename it.
Note: Do NOT edit the default.meta file directly, you may need the default values in that file at a future time.

2. In the local.meta file, add the name of the new role to the stanza that corresponds with the desired access.

<table>
<thead>
<tr>
<th>Default stanza</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>[manager/accesscontrols] access = read : [ * ], write : [ admin, power ]</td>
<td>Allow all users to read this app's contents, or access functions in the Splunk Manager page, depending on the directory you are in. Unless overridden by other metadata, allows only admin and power users to share objects into this app.</td>
</tr>
<tr>
<td>[views] [manager/accesscontrols] access = read : [ * ], write : [ admin ]</td>
<td>Determines the access controls for the Manager page access.</td>
</tr>
</tbody>
</table>

3. When you have made all of your changes, restart Splunk Enterprise.

Examples

Example 1: A new role called "usermanager" only inherits capabilities from a user and has no searches or indexes inherited. The intent is to create a role that has no access to data and is solely used to create and manage user accounts.

To create this role you would edit the following stanza:

```plaintext
[manager/accesscontrols]
access = read : [ admin ], write : [ admin ]
```

To include the following:

```plaintext
[manager/accesscontrols]
access = read : [ admin, usermanager ], write : [ admin, usermanager ]
```

You have just given "usermanager" the ability to see and edit stuff in the "Access controls" pages in Manager.

Example 2: To enable the role "userview," to access but not edit the pages, only add the role to the read value:

```plaintext
[manager/accesscontrols]
access = read : [ admin, userview, usermanager ], write : [ admin, usermanager ]
```

You can also grant access to read the manager pages to EVERY role using the wildcard:

```plaintext
[manager/accesscontrols]
access = read : [ * ], write : [ admin ]
```

Example 3: You want to have a subset of users who can only read sales data that you specify. To accomplish this you can create an app for the dashboard and then create a new role "salesusers."

In the local.meta file in your app directory (remember that you can create one from the default.meta file), you then edit the following stanza:

```plaintext
[viewstates]
access = read : [ * ], write : [ * ]
```

to read:

```plaintext
[viewstates]
access = read : [ salesusers ], write : [ admin ]
```
Find existing users and roles

To locate the existing user or role in Splunk Web:

1. In the main menu click **System > Access Controls**.
2. Click Users or Roles to choose which entity you want to search.

By default, all fields are searched for the specified string. To search a particular field, specify the name of the field. Note that Splunk Search supports wildcards. For example:

- To search only email addresses:
  
  "email=<{email address or address fragment}>:

- To search only the "Full name" field:
  
  "realname=<{name or name fragment}>.

- To search for users in a given role:
  
  "roles=":

Delete all user accounts

Remove all the user data (user accounts) from your Splunk Enterprise installation by typing 

`. /splunk clean` followed by

the `userdata` argument. This deletes all the user accounts and you are required to create new admin user credentials.

Removing user data is irreversible. If you accidentally delete user data, you must recreate the accounts manually. Additionally, you must satisfy any password requirements that are in place when you recreate the accounts.

To remove all of the user accounts in the system:

`. /splunk clean userdata`

To remove the user accounts in the system and skip the confirmation prompt:

`. /splunk clean userdata -f`

To recreate the default admin account:

In Splunk Enterprise 7.1.0 and later, the default admin account is no longer automatically recreated on startup after

running 

`. /splunk clean userdata` or 

`. /splunk clean all`

To recreate the admin account, you can create a

`$SPLUNK_HOME/etc/system/local/user-seed.conf` file with the following

information before restarting Splunk Software.

```plaintext
[user_info]
USERNAME = admin
PASSWORD = <your new password>
```
Secure access for Splunk knowledge objects

As you use Splunk Enterprise, you create a variety of knowledge objects such as event types, tags, lookups, field extractions, workflow actions, and saved searches. Splunk Web lets you restrict and expand access to knowledge objects within your Splunk implementation. You can use it to:

- Make an object available to users of all apps.
- Make an object available to users of a particular app.
- Restrict object access by user role.
- Disable or delete objects.
- Allow users to share or delete objects they do not own.

For more information about securing your knowledge objects, see Manage knowledge object permissions and Disable or delete knowledge objects in the Knowledge Manager Manual.

Use Access Control Lists

To help secure your Splunk configuration, use the Splunk Enterprise Access Control Lists (ACLs) to limit the IP addresses that can access various parts of your networks.

To configure ACLs, you edit server.conf and inputs.conf to specify the IP addresses that will be accepted or rejected for various communications.

How to set up ACLs

The addresses are separated by commas or spaces. You can provide the addresses in the following formats:

- A single IPv4 or IPv6 address. For example: 10.1.2.3, fe80::4a3.
- A CIDR block of addresses. For example: 10/8, fe80::1234/32.
- A DNS name, possibly with an * used as a wildcard, for example: myhost.example.com, *.splunk.com.
- A single * which matches anything (this is the default value).

To add addresses that you wish to include, you add the addresses in one of the formats described below. To exclude an address you prefix the address with '!'.

Rules are applied in order, and the first one to match is used. For example, !10.1/16, * will allow connections from everywhere except the 10.1.*.* network.

Where to set up ACLs

You can secure IP addresses for the following connections by editing the [Accept from] value:

- To instruct a node to only accept replicated data from other nodes with specific IPs, edit the httpServer stanza in server.conf.
  If you set this attribute, you must make sure that you include the IP addresses of all other peers in the cluster. For more information about clusters, see "About clusters and index replication" For more information about editing server.conf, see server.conf.

- To restrict TCP communications to specific IP addresses, edit the tcp stanza in inputs.conf. Be careful, as this
will overwrite the output values in `server.conf` if the information conflicts.

- To restrict TCP communications that use SSL to specific IP addresses, edit the `tcp-ssl stanza` in `inputs.conf`.

- To restrict your indexer to accept data only from forwarders with specific IP addresses, edit the `splunktcp stanza` in `inputs.conf`. This prevents someone from spoofing your forwarders and possibly corrupting your data.

- If your forwarder to indexer communications are secured with SSL, edit the `splunktcp-ssl stanza` in `inputs.conf` to restrict your indexer to only accept data from forwarders with specific IP addresses.

- To restrict UDP communications to specific IP addresses, edit the `UDP stanza` in `inputs.conf`.

For more information about editing `inputs.conf`, see `inputs.conf`
Native Splunk Enterprise authentication

Set up Splunk authentication

Splunk authentication allows you to easily set up users within your system. Splunk authentication always takes precedence over any external authentication systems. This is the order in which users are authenticated:

1. Splunk authentication.

2. LDAP or Scripted authentication (if enabled). For more information, see “Set up user authentication with LDAP” and “Set up user authentication with external systems.”

Note: LDAP and scripted authentication cannot be used together.

You can create new users and assign those users to roles with a role-based access control system in two ways:

- Use Splunk Web to create users and assign roles. For more information see "Configure users with Splunk Web."
- Use the CLI to create users and then assign them to roles with Splunk Web. For more information see "Configure users with the CLI."

Important naming guidelines when creating users and roles

Usernames stored in native authentication cannot contain spaces, colons, or forward slashes. Names are case-insensitive, for example: "Jacque", "jacque", "JacQue" are all the same to Splunk authentication.

Role names must use lowercase characters only. They cannot contain spaces, colons, or forward slashes.

Configure users with Splunk Web

To configure users and roles In Splunk Web:

1. Navigate to Settings > Users and Authentication > Access controls.
2. Click Users.
3. Click New or select an existing user to edit.
4. Specify or change the information for the user. You can specify the user's:
   - full name.
   - email address.
   - time zone. This allows users to view events and other information in their own time zone.
   - default app. This overrides the default app inherited from the user's role.
   - password.
5. Assign the user to an existing role or roles and click Save.

You can also create a role specifically for a user, defining exactly what access that user has to Splunk Enterprise. You can then assign the user to that role. For information about roles, read “About role-based user access.”

For information about managing user settings, see the Splunk Enterprise Administration Guide.
Configure users with the CLI

In the CLI, use the `add user` command. Here are some examples:

- To add a new administrator user with the password "changeme2":
  ```bash
  ./splunk add user admin2 -password changeme2 -role admin -auth admin:changeme
  ```

- To change an existing user's password to "fflanda":
  ```bash
  ./splunk edit user admin -password fflanda -role admin -auth admin:changeme
  ```

**Important:** Passwords with special characters that would be interpreted by the shell (for example '!' or '!') must be either escaped or single-quoted. For example:

```bash
./splunk edit user admin -password 'fflanda$' -role admin -auth admin:changeme
```

**Add a user to a role with Splunk Web**

You can add a user to a default role or to a custom role you create yourself. For more information, see "About role-based user access."

To add a user or users to a role with Splunk Web:

1. Click **Settings > Access Control > Access Controls** in the main menu.
2. Click **Users**.
3. Edit an existing user or create a new one.
4. Choose which role to map to from the **Role** list. Any custom roles you have created in `authorize.conf` will be listed here.
Authentication with tokens

Set up authentication with tokens

Authentication tokens, also known as JSON Web Tokens (JWT), are a method for authenticating Splunk platform users into the Splunk platform. Tokens let you provide access to environments without having to provide the standard types of credentials. Instead of providing a username and password, you provide the token. As a Splunk platform user, you can use tokens to make calls to Representational State Transfer (REST) endpoints and use the Splunk CLI on Splunk platform instances. Tokens are credentials, so you must closely guard them, and not share them with anyone who does not explicitly need access to Splunk platform services.

Authentication tokens are different from other types of token that you can configure in Splunk Enterprise on forwarders and indexers for authenticated communication between those components, or HTTP Event Collector, though their function is similar.

As a Splunk platform administrator, you can control a number of authentication token properties:

- Who owns the token. A token is associated with a username on the instance
- Token audience. A label that indicates the token's purpose
- How long a token lasts
- When its validity begins. It does not have to become valid immediately when you create it
- Whether or not it is enabled, at any time

You can also delete the token if you no longer want the user to have access through the token. If you disable or remove a token, users of that token lose access to the instance unless they have standard credentials such as the username and password.

Supported Splunk deployment types and authentication schemes for tokens

You can create and assign tokens to various user types that can access a Splunk platform instance, based on the type of authentication system that the instance uses. At this time, tokens are supported in the following deployment types and authentication schemes:

Supported deployment types

- Single-instance, on search heads only
- Search head cluster nodes

Unsupported deployment types

- Indexers
- Indexer cluster nodes
- Universal forwarders
- Splunk Cloud instances of any kind

Supported authentication schemes

- Native authentication
- Lightweight Directory Access Protocol (LDAP) authentication
• Single Sign-On (SSO) schemes that use Security Assertion Markup Language (SAML). These schemes must either support Attribute Query Requests (AQR) or provide information through scripted authentication extensions.

Unsupported authentication schemes

• Proxy SSO

Prerequisites for activating tokens

Before activating token authentication, confirm that you have performed the following tasks:

• Enable Transport Layer Security (TLS)/SSL on your Splunk platform instance. See About securing Splunk Enterprise with SSL for details.
• Confirm that you have enabled app key value store (KV Store). By default, KV store is enabled on search heads. See About app key value store in the Admin Manual for more information.

Prerequisites for creating and configuring tokens

Before you create and configure tokens, confirm that you have performed the following tasks:

• Logged in to the Splunk platform as a user that is either an administrator level or holds a role with at least one of the following Splunk platform capabilities:
  ♦ edit_tokens_settings, which turns token authentication on or off
  ♦ edit_tokens_all, which lets you create, view, and manage tokens for any user on the instance
  ♦ edit_tokens_own, which lets you create, view, and manage tokens for yourself
• Completed all of the prerequisites for activating authentication tokens
• Enabled token authentication

Information on the "list_*" capabilities for tokens

If you have been assigned a role that includes only the following capabilities, you can view tokens, but cannot create them, or enable and disable token authentication:

• The list_tokens_all capability lets you see all tokens on the instance.
• The list_tokens_own capability lets you see your own tokens.

The edit_tokens_* capabilities include the ability to view tokens, as provided by the list_tokens_* capabilities.

How authentication tokens work

When you create a token, the Splunk platform picks an ID for the token. The token ID references the token information, which the platform stores securely in a KV store collection.

When you use tokens, you must provide the token with each request. If you use the Splunk CLI, you must supply the token as part of the command or use an environment variable that the CLI picks up. If the Splunk platform uses its native authentication scheme, it does not cache authentication details.

When you present a token as part of a REST request or CLI command, the Splunk platform authentication system does the following:

• It checks to see if token authentication is enabled.
- It processes the token that it receives by validating its signature.
- It checks whether or not the token has expired, or if its validity has not yet started.
- It confirms that the token is available and has not been deleted.
- It checks whether or not the token is enabled.
- It confirms that the user that is associated with the token is authorized to use it.

If all checks pass, it authenticates the user and lets the submitted operation complete.

**Enable, create, manage, and use tokens**

See the following topics in this chapter for additional information on how to work with authentication tokens:

- Enable or disable token authentication
- Create authentication tokens
- Manage and delete authentication tokens
- Use authentication tokens

**Enable or disable token authentication**

You can enable token authentication at any time if your Splunk platform account has the appropriate permissions. Token authentication is off by default on the Splunk platform.

You can also disable token authentication at any time if you have enabled it and have the appropriate permissions. If token authentication is disabled, token users cannot authenticate into the instance, even if you have previously defined valid tokens.

Tokens retain their individual validity status regardless of whether token authentication is on or off, and when you re-enable token authentication after disabling it, holders of valid tokens can use them again.

**Prerequisites for enabling or disabling token authentication**

Before you can enable token authentication, you must satisfy the following requirements:

- The Splunk platform instance where you want to enable token authentication must not operate in legacy mode, where Splunk Web operates as a separate process. If the Splunk platform is in legacy mode, token authentication does not run. See Start and Stop Splunk Enterprise in the *Admin Manual* for information on legacy mode.

- The account that you use to log into the Splunk platform must hold a role that has the `edit_tokens_settings` Splunk platform capability before you can turn token authentication on or off.

**Enable token authentication for a Splunk platform instance**

On Splunk Cloud instances, you can enable token authentication by using Splunk Web. On Splunk Enterprise instances, you can enable token authentication by using Splunk Web, editing configuration files, or making a call to a Representational State Transfer (REST) endpoint.

At this time, you cannot use the Splunk CLI to enable or disable token authentication.
Enable token authentication using Splunk Web

When token authentication is off, the following message displays on the “Tokens” page in Splunk Web:

**Token authentication is currently disabled**

To enable token authentication, click Enable Token Authentication.

Perform this procedure on the instance where you want to enable token authentication.

1. Log in to the Splunk platform instance as an administrator user, or a user that can manage tokens settings.
   
   You cannot use a token to log in to Splunk Web. You must provide a valid user name and password.

2. After you log in successfully, in the system bar, select **Settings > Tokens**.
3. Click **Enable Token Authentication**. The Splunk platform instance enables token authentication immediately, and there is no need to restart the instance.

Enable token authentication using configuration files

Perform this procedure on the Splunk Enterprise instance where you want to enable token authentication.

1. Open a shell prompt or PowerShell window.
2. Change to the `$SPLUNK_HOME/etc/system/local` directory.
3. Use a text editor to open the `authorize.conf` file for editing.
4. In the `authorize.conf` file, add the following lines of text:

   ```
   [tokens_auth]
   disabled = false
   ```

5. Save the `authorize.conf` file and close it.
6. Restart the Splunk platform.

Set a default relative token expiration time using configuration files

Optionally, to set a default relative time expiration for any tokens on the Splunk Enterprise instance, use this procedure. Expiration times that you specify in the token creation dialog override the default setting. You cannot perform this operation in Splunk Web, and you cannot set an expiration time in the past.

1. Open a shell prompt or PowerShell window.
2. Change to the `$SPLUNK_HOME/etc/system/local` directory.
3. Use a text editor to open the `authorize.conf` file for editing.
4. In the `tokens_auth` stanza, add the following line of text, substituting `<relative time>` with a string that represents an amount of time from the time that you create a token:

   ```
   expiration=<relative time>
   ```

   For example, if you want to specify a default expiration time of 5 days for a token after you create it, set `<relative time>` to `+5d`.

5. Save the file and close it.
6. Restart the Splunk platform.

See Time modifiers in the *Search Reference* manual for more information on time modifier syntax.
Enable token authentication using REST

The curl command does not come standard on Windows PowerShell. Instead, you can use the Invoke_RestMethod PowerShell cmdlet on PowerShell versions 3.0 and higher.

1. Open a shell prompt.
2. Run the following command

   curl -k -u <splunk_username>:<password> -X POST https://<servername>:<port>/services/admin/token-auth/tokens_auth -d disabled=false

Splunk Enterprise enables token authentication immediately, and there is no need to restart the instance.

Disable token authentication on a Splunk platform instance

On Splunk Cloud instances, you can disable token authentication by using Splunk Web. On Splunk Enterprise instances, you can disable token authentication by using Splunk Web, editing configuration files, or making a call to a REST endpoint.

Disable token authentication using Splunk Web

Perform this procedure on the instance where you want to disable token authentication.

1. Log in to the Splunk platform instance as a user that can edit token settings.
   You cannot use a token to log in to Splunk Web. You must provide a valid user name and password.

2. After you log in, in the system bar, select Settings > Tokens.
3. Click Disable Token Authentication. The instance disables token authentication immediately, and there is no need to restart the instance.

Disable token authentication using configuration files

Perform this procedure on the Splunk Enterprise instance where you want to disable token authentication.

1. Open a shell prompt or PowerShell window.
2. Change to the $SPLUNK_HOME/etc/system/local directory.
3. Use a text editor to open the authorize.conf file.
4. In the authorize.conf file, edit the following lines of text:

   [tokens_auth]
   disabled = true

5. Save the authorize.conf file and close it.
6. Restart Splunk Enterprise.

Disable token authentication using REST

The curl command does not come standard on Windows PowerShell. Instead, you can use the Invoke_RestMethod PowerShell cmdlet.

1. Open a shell prompt.
2. Run the following command

   curl -k -u <splunk_username>:<password> -X POST https://<servername>:<port>/services/admin/token-auth/tokens_auth -d disabled=false

Splunk Enterprise enables token authentication immediately, and there is no need to restart the instance.
curl -k -u <splunk_username>:<password> -X POST https://<servername>:<port>/services/admin/token-auth/tokens_auth -d disabled=true

The instance disables token authentication immediately, and there is no need to restart the instance.

Create, use, manage, and delete tokens

After you enable token authentication, you can do the following with authentication tokens:

• Create tokens. See Create authentication tokens.
• Manage or delete tokens. See Manage or delete authentication tokens.
• Use tokens to authenticate. See Use authentication tokens.

If you disable token authentication, any tokens that are on the instance become inaccessible immediately, and you must enable token authentication again to restore access to tokens that are valid.

Create authentication tokens

Authentication tokens let users of Splunk platform environments access Representational State Transfer (REST) endpoint resources or use the Splunk CLI in those environments. You can create tokens in Splunk Web or use an API call to a REST endpoint on the instance where the tokens are to reside.

At this time, you cannot use the CLI to create authentication tokens.

Prerequisites for token creation

• You must enable token authentication. See Enable or disable token authentication.
• Your Splunk platform account must satisfy at least one of the following criteria before you can create authentication tokens:
  ♦ If you want to create tokens for yourself, your account must hold a role that has the edit_tokens_own capability.
  ♦ If you want to create tokens for any user on the instance, your account must hold a role that has the edit_tokens_all capability. See About defining roles with capabilities for additional information on Splunk platform capabilities.
• You must be prepared to save or share the token immediately after you create it. You only have one opportunity to do so, and the full token cannot be recalled after you close the “New token” dialog box.

Supported user types for token creation

You can only create tokens for users that exist on the Splunk platform instance where you create the token. The users that exist on the instance depend on the authentication scheme that the instances uses:

• Native Splunk authentication: Where user accounts exist only on that specific instance
• Authentication through a Lightweight Directory Access Protocol (LDAP) server
• Authentication through a single-sign-on (SSO) scheme that uses Security Assertion Markup Language (SAML)

The Splunk platform confirms that the user you entered exists and raises an error message if the user does not exist.
**Considerations for creating tokens on instances that use the LDAP authentication scheme**

When you create a token on an instance that uses the LDAP authentication scheme, the LDAP server treats that creation as a login for LDAP caching purposes. The Splunk platform connects to the LDAP server to validate the user and any associated LDAP groups.

**Considerations for creating tokens on instances that use the SAML authentication scheme**

You might need to perform additional tasks before you can create tokens for users who use the SAML authentication scheme to access Splunk platform resources. This need depends on the identity provider (IdP) that the Splunk platform instance connects to with SAML.

- If the SAML IdP supports Attribute Query Requests (AQRs), then no additional work is required, and tokens that you create should be immediately usable by users who access a Splunk platform instance that uses the SAML scheme.
- If the SAML IdP does not support AQRs, then you must configure authentication extensions that interface with the IdP. See Configure SAML authentication extensions for additional information and instructions.

**Save or share the token immediately after you create it**

When you create an authentication token, for security purposes, you only have one opportunity to see the entire token. After you specify a user and audience for the token and click "Create", the token appears as a string of text in the "New Token" dialog box. You must copy this token and paste it into another document such as a text file before closing the "New Token" dialog box.

Token users must have the full token to authenticate without credentials. If you close the Create Token dialog box before saving the full token somewhere, then you must create a new token, as you cannot recover the one that you previously created.

**Configure token expiry and "Not Before" settings**

When you create a new token, you can set whether or not the token expires, and whether or not it is valid before a certain time. Both of these choices are optional, which means that you can configure a token to last forever and be available for use immediately.

If you want an authentication token to expire, you must set an expiry date and time for it. You can set an absolute time, such as "Friday, February 1, 2019 at 10:30", or you can set a relative time, which is a certain period of time from the current time, for example, three days from now, a week from now, or two months from now. Expiration times can never be in the past.

If you do not set an expiration time, then the Splunk platform uses the default global expiration time, which is "never". You can change this default. See Set a default relative token expiration time using configuration files.

If you do not want the token to be valid immediately, you can set a "Not Before" time for it. This means that even though you create the token now, it cannot be used until the Not Before time has passed. For example, if today is Friday, January 25, 2019, and you do not want the token to be used until the following Friday, you can set an absolute time of Friday, February 1, 2019 at 00:00, or a relative time of $+7d$ (7 days from now). The "Not before" time can never be in the past, nor can it be after the expiration time.
Relative time versus absolute time

When you specify expiration and "Not before" times, you can specify an absolute time or a relative time. You can specify either type of time format for either type of time.

You indicate the absolute time by specifying a date and time, including year, month, day, hour, minute, and second. In Splunk Web, you can use the date picker to chose the appropriate date and time, or you can type it in. If you type it in, you must specify it in the following format:

YYY-MM-DDTHH:MM:SS[+HH:MM]

The T between DD and HH is the actual letter T. The +HH:MM represents the time zone that you want to use, and is optional.

You indicate the relative time by specifying a string that represents an amount of time beyond the current time. In general, specify the following format:

+<number>[s|h|m|d]@[s|h|m|d][+<number>[s|h|m|d]]

The letters s, m, h, and d are identifiers that represent seconds, minutes, hours, and days, respectively. You can also use the following words as identifiers:

- seconds, secs, sec
- minutes, mins, min
- hours, hrs, hr
- days, day

If you want to round down to a certain time, you can include the @ modifier in the time string. For example, if you want a token to expire 10 days from now at the beginning of the day, you can specify +10d@. This string reads as "10 days from now, on the day."

You can also concatenate different identifiers. For example, to have a token expire 15 days and 5 hours, on the hour, from now, you can specify +15d+5h@h.

Use Splunk Web to create authentication tokens

1. In the system bar, click Settings > Tokens.

   If you see a message that says you have not enabled token authentication, see Enable token authorization for instructions on how to enable token authorization.

3. In the "New Token" dialog, enter the Splunk platform user that you want to create the token for in the User field.
4. Enter a short description of the token purpose in the Audience field.
5. (Optional) In the Expiration drop down list, select one of Absolute Time or Relative Time. This selection determines what to enter in the text field below the drop down list.
   1. If you selected Absolute Time, then two text fields appear under the drop down.
      1. Enter a valid date into the first field. You can also click the field to select a date from a pop-up calendar.
      2. Enter a valid 24-hour time in the second field.
   2. Otherwise, one text field appears under the drop down list.
      1. Enter a string that represents how long after the current time you want the token to remain valid. For example, if you want the token to expire 10 days from now, enter +10d into this field.
6. (Optional) In the Not Before drop-down, select one of Absolute Time or Relative Time.
7. Repeat the step you used for the "Expiration" control. The "Not before" time can neither be in the past, nor can it be later than the "Expiration" time.
8. Click Create. The New Token window updates the Token field to show you the token that has been generated.
9. Select all of the token text in the field. Depending on your operating system and browser, you can click on the "Token" field, then either triple click or press Ctrl-A or Command-A on your keyboard.

Confirm that you have selected all of the token text. There are no further opportunities to see the whole token after you close the window.
10. Copy the text from the Token field.
11. Paste the token into a text file, e-mail, or other form of communication to the person you have authorized to use the token.

Confirm that you share the token only with those who you have authorized to use it. Anybody who has the full token can use it to authenticate.
12. Click Close.
13. Share the token with its authorized user.

Use REST to create authentication tokens

You can also create authentication tokens by making an API call to the services/authorization/tokens REST endpoint on a Splunk platform instance with the cURL command. The cURL command is not available on Windows PowerShell; instead you can use the Invoke-RestMethod command on PowerShell version 3.0 and higher.

If you have an existing valid token, you can use it rather than user credentials to authenticate when creating a new token. The user account that is associated with the token must satisfy the previously-described criteria for token creation.

See "Example API calls" later in this topic for examples on token creation.

1. Open a shell prompt.
2. Generate the token.

```
curl [-k] [-u <username>:<password>|-H "Authorization: Bearer <existing_token>"] -X POST 
https://<servername>:<management_port>/services/authorization/tokens?output_mode=json --data 
name=<token_user> --data audience=<audience> [--data-urlencode expires_on=[<relative_time>|<absolute_time>]] 
[--data-urlencode not_before=[<relative_time>|<absolute_time>]]
```

See the syntax variable table that follows this procedure for a description of each variable in the syntax.
3. In the output that appears, look for the "token": string. The text immediately after this string, up to the next " character, is the token.
4. Share the whole token with the person who is to use it.
5. Close the shell prompt.

**Sample output of token creation**

Following is sample output from a REST call to generate a token.

```
{"links":{"create":"/services/authorization/tokens/_new"},"origin":"https://mysplunk.com:8089/services 
/authorization/tokens","updated":"2019-03-07T22:55:46+00:00","generator":{"build":"85dd8eb9b9c6","version":"7.3.0"},"entry":{"name":"tokens","id":"https:// 
/mysplunk.com:8089/services/authorization/tokens/tokens"},"updated":"1970-01-01T00:00:00+00:00","links":{"alternate":"/services/authorization/tokens/tokens","list":"/services/authorization 
/tokens/tokens","edit":"/services/authorization/tokens/tokens","remove":"/services/authorization/tokens"
```
Determine the meaning of the variables used in the previous syntax using the following table:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Meaning</th>
<th>Required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>The user that is creating the new token</td>
<td>If using a token: No</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Otherwise: Yes</td>
</tr>
<tr>
<td>password</td>
<td>The password for the user that is creating the new token</td>
<td>If using a token: No</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Otherwise: Yes</td>
</tr>
<tr>
<td>existing_token</td>
<td>A valid, existing token that has already been issued to the user that is creating the new token</td>
<td>If using a token: Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Otherwise: No</td>
</tr>
<tr>
<td>servername</td>
<td>The Splunk platform instance where the token is to reside</td>
<td>Yes</td>
</tr>
<tr>
<td>management_port</td>
<td>The management port of the Splunk instance. Is usually 8089 but can be different depending on how the instance is configured.</td>
<td>Yes</td>
</tr>
<tr>
<td>token_user</td>
<td>The user that is to receive the new token. This user must already exist on the Splunk instance.</td>
<td>Yes</td>
</tr>
<tr>
<td>audience</td>
<td>The purpose for which the token is being created. Splunk Enterprise uses this to let you group tokens that have been assigned to different users.</td>
<td>Yes</td>
</tr>
<tr>
<td>expires_on</td>
<td>An argument that specifies when a token expires. Expired tokens are invalid and authorization requests that contain them are rejected.</td>
<td>No</td>
</tr>
<tr>
<td>not_before</td>
<td>An argument that specifies a future time when the token is to become valid. &quot;Not before&quot; tokens are invalid until the &quot;Not before&quot; time has passed, and authorization requests that occur before that time with these kinds of tokens are rejected.</td>
<td>No</td>
</tr>
<tr>
<td>absolute_time</td>
<td>A string that represents a specific date and time. Must be in the format YYYY-MM-DDTHH:MM:SS[+HH:MM]. Include the time zone offset when applicable. For example: 2019-02-09T07:35:00+07:00</td>
<td>No, uses default expiration time if not included</td>
</tr>
<tr>
<td>relative_time</td>
<td>A string that represents a period of time past the current time. Must be in the format +&lt;number&gt;[a][m][h][d]. For example, +90m means 90 minutes from now, +15d means 15 days from now.</td>
<td>No</td>
</tr>
</tbody>
</table>
**Example API calls for creating tokens**

The following example is of an administrator using their credentials to create a token for user "jdoe" with an expiration of 30 days from the current time:

```bash
curl -k -u admin:Ch#ng3d! -X POST https://splunk1.server.com:8089/services/authorization/tokens?output_mode=json --data name=jdoe --data audience=Users --data-urlencode expires_on=+30d
```

The following example is of an admin using a valid token to create another token for user "sallyjane" for the purpose "Managers" that uses the default token expiration time.

```bash
```

The following example is of an admin creating a token for user "bobd" on February 5, 2019. The admin wants the token to become valid on March 1, 2019 at midnight and expire around 60 days after that. The instance uses a nonstandard management port 44514:

```bash
curl -k -u admin:Ch#ng3d! -X POST https://mysplunk.com:44514/services/authorization/tokens?output_mode=json --data name=bobd --data audience=Accountants --data-urlencode not_before=2019-03-01T00:00:00 --data-urlencode expires_on=+85d
```

The following example is of an admin creating a token for user "steveg" for the purpose "Operations". The token is not to become valid before 10 days, on the day, from the current time, and should expire 45 days, on the day, plus 2 hours, on the hour, after it becomes valid. The instance uses a nonstandard management port 38182:

```bash
curl -k -u admin:Ch#ng3d! -X POST https://mysplunk.com:38182/services/authorization/tokens?output_mode=json --data name=steveg --data audience=Operations --data-urlencode not_before=+10d@d --data-urlencode expires_on=+55d@d+2h@h
```

**Use, manage, and delete tokens**

After you created tokens, you can do the following:

- Manage or delete tokens. See Manage or delete authentication tokens.
- Use tokens to authenticate. See Use authentication tokens.

**Manage or delete authentication tokens**

Before you can manage or delete authentication tokens, you must have enabled token authentication and created at least one token. If you have not enabled token authentication, see Enable token authorization for instructions.

You can manage authentication tokens that you have created in Splunk Web or by using Representational State Transfer (REST) calls. You can view the following information on each token:

- **Token ID**
- **Token issuer** (Issued by, comprised of the Splunk platform user who created the token and the hostname on which the token was created)
- **Token owner** (Username or subject) and audience
- **Token validity ranges including** Not before and expiration times
- **The Identity Provider** (the authentication scheme that was in use when the administrator created the token)
• When the token was last used
• The IP address that last used the token

Owing to security reasons, you cannot do any of the following with tokens:

• Reassign token ownership. A token is assigned to a single user and audience at all times.
• Change a token audience.
• Change the expiration of a token.
• Change the "Not before" validity of a token.
• Renew an expired token. Users of expired tokens lose access immediately.

If you need to change any of these properties of a token, then you must create a new token with the updated settings, share the token with the user, and, optionally, disable or delete the old tokens.

Considerations for managing authentication tokens on instances that use LDAP for authentication

There are some caveats for using and managing authentication tokens on Splunk platform instances that use LDAP to authenticate.

• The LDAP cache controls how long Splunk platform instances that use LDAP retain information from LDAP queries. By default, the LDAP cache never expires, but you can control when it expires by editing a setting in the limits.conf configuration file. See Configure LDAP with configuration files for instructions.
• When you delete a user from an LDAP provider, delete any tokens that are associated with the deleted user as well. Tokens can remain valid until the user entry in the LDAP cache expires.
• While tokens that are associated with a deleted user no longer work for authentication, if you create a new user with the same username, the LDAP provider can re-associate those tokens with the new user, potentially causing unauthorized access.

Manage authentication tokens in Splunk Web

You can perform the following actions on the Tokens page:

• Create new tokens. See Create authentication tokens for the procedure.
• Enable or disable existing tokens. See "Enable or disable authentication tokens" later in this topic.
• Delete existing tokens. See "Delete authentication tokens" later in this topic.

While you can view token IDs, there is no way to view a token in its entirety. Token users require the full token before they can use it. You cannot give the token ID to a user to use as a token if they have forgotten or misplaced the token. You must either provide the entire token, if it is available to you, or create a new one.

View token information

The Tokens page lists information on the tokens that you have created. Each token is represented by its token ID.

It is not possible to view a full token on this page. You can only view a full token immediately after you create it in the "New Token" dialog box, and before you close that dialog box.

1. From the system bar, click Settings > Tokens. The Tokens page appears.
2. (Optional) Use the Search text box to locate a token by one of the following fields:
   • ID

58
3. (Optional) Hover the mouse over a token ID to see a tooltip that shows the entire token ID.
4. (Optional) Select the > button to expand a token entry and show detailed information about a token:
   - Token ID
   - Token issuer and issuing workstation
   - "Not before" validity time
   - The Splunk authentication scheme that this token uses
   - The last IP address that used the token successfully

The instance updates the last seen IP address and time whenever you use a token. There is a period of up to two minutes after use, where usage information is cached, and Splunk Web does not show multiple uses during that period.

**Enable or disable existing tokens**

When you disable a token, users who use the token lose access immediately. You must enable the token again for users to regain access while it is valid.

Tokens that have not reached their "Not Before" validity time remain unusable until that time has passed, regardless of the changes that you make with this procedure.

1. From the system bar, click **Settings > Tokens**. The tokens page appears.
2. (Optional) Use the Search text box to locate a token. The page updates to show only tokens that match the text you entered.
3. Locate the token whose status you want to change.
4. In the Actions column for the token, if a token is enabled, click the Disable link to disable the token.
   1. In the Disable Token dialog box that appears, click Disable.
5. Otherwise, if a token is disabled, click the Enable link to enable the token.
   1. In the Enable Token dialog box that appears, click Enable.
6. Repeat these actions for additional tokens whose status you want to change. You can use the Search text box to update the list of tokens.

**Delete an existing token**

When you delete a token, users who use the token lose access when the cache for the token expires, up to two minutes after token revocation. You must reissue a new token or standard credentials to grant access to the user that had the previous token.

1. From the system bar, click **Settings > Tokens**. The tokens page appears.
2. (Optional) Use the Search text box to locate a token. The page updates to show only tokens that match the text you entered.
3. Locate the token that you want to delete.
4. In the Actions column for the token, click the Delete link to disable the token.
   1. In the Delete Token dialog box that appears, click Delete.
5. Repeat these actions for additional tokens that you want to delete. You can use the Search text box to update the list of tokens.
Manage authentication tokens using REST

You can use either a REST client or the `curl` command-line utility to generate REST requests to your Splunk Enterprise instance. All of the following command examples use `curl`. In addition to using standard credentials to manage tokens, you can also use a valid token to perform these requests.

1. Open a shell prompt.
2. From the prompt, run the appropriate `curl` command, based on how you want to authenticate.
   1. To authenticate with standard credentials, provide them as part of the command: `curl -k -u <username>:<password> ...
   2. To authenticate with a token, provide the token in an authorization header: `curl -k -H "Authorization: Bearer <valid_token> ...
3. Review the output to confirm that the command completed successfully.
4. (Optional) Perform additional requests, depending on the endpoints you are using and the tasks you want to complete.

View all existing tokens

curl -k -u <username>:
$password >X GET https://<server>:
<management_port>/services/authorization/tokens

This command generates the following output:

```xml
<?xml version="1.0" encoding="UTF-8"?>
...
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:s="http://dev.splunk.com/ns/rest"
xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
<title>tokens</title>
<id>https://10.224.61.92:43705/services/authorization/tokens/id
<updated>2019-02-19T22:29:33+00:00</updated>
...
<author>
  <name>Splunk</name>
</author>
...
<entry>
  <title>45a2b05b2cc737e4ce6387092a00b8fcbb7502960dd651a0ab16129161495ad6</title>
  <id>https://10.224.61.92:43705/services/authorization/tokens/45a2b05b2cc737e4ce6387092a00b8fcbb7502960dd651a0ab16129161495ad6</id>
  <updated>1970-01-01T00:00:00+00:00</updated>
  <link href="/services/authorization/tokens/45a2b05b2cc737e4ce6387092a00b8fcbb7502960dd651a0ab16129161495ad6" rel="alternate"/>
  <author>
    <name>system</name>
  </author>
  ...
  <content type="text/xml">
    <s:dict>
      <s:key name="claims">
        <s:dict>
          <s:key name="aud">Tokentown</s:key>
          <s:key name="exp">0</s:key>
          <s:key name="iat">1550614409</s:key>
          <s:key name="iss">admin from so1</s:key>
        </s:dict>
      </s:dict>
    </content>
  </entry>

```
View existing tokens by user

curl -k -u <username>:<password> -X GET https://<server>:<management_port>/services/authorization/tokens?username=<token_user>
View existing tokens by status

```
curl -k -u <username>:<password> -X GET 
https://<server>:<management_port>/services/authorization/tokens?status=<enabled|disabled>
```

View information on a single existing token

```
curl -k -u <username>:<password> -X GET https://<server>:<management_port>/services/authorization/tokens -d 
id=<token_id>
```

Disable an existing, enabled token

```
curl -k -u <username>:<password> -X POST 
https://<server>:<management_port>/services/authorization/tokens/<token_user> -d id=<token_id> -d 
status=disabled
```

Enable an existing, disabled token

```
curl -k -u <username>:<password> -X POST 
https://<server>:<management_port>/services/authorization/tokens/<token_user> -d id=<token_id> -d 
status=enabled
```

Delete an existing token

```
curl -k -u <username>:<password> -X DELETE 
https://<server>:<management_port>/services/authorization/tokens/<token_user> -d id=<token_id>
```

This command generates the following output:

```
<?xml version="1.0" encoding="UTF-8"?>
...
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:s="http://dev.splunk.com/ns/rest"
xmns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
<title>tokens</title>
&id>https://10.224.61.92:43705/services/authorization/tokens</id>
<updated>2019-02-19T23:04:31+00:00</updated>
<generator build="71b3ebc05ef9" version="7.3.0"/>
<author>
    <name>Splunk</name>
</author>
...
<s:messages>
   <s:msg type="INFO">Token(s), removed.</s:msg>
</s:messages>
</feed>
```

Use authentication tokens

If you have been assigned an authentication token, you can access a Splunk platform instance using Representational State Transfer (REST) calls or the CLI. These methods let you access the instance and make requests without having to authenticate with credentials.
You must possess a valid token that has been given to you from an administrator who maintains the instance. The token cannot be expired, as expired tokens are unusable. Administrators can delay when the validity of a token starts, meaning that the token cannot be used until its validity begins.

You cannot use a token on any instance other than the instance where the administrator granted you the token. For example, if you have a token for server1, and need access to server2, you must ask the administrator for server2 to give you access to a token there. There is an exception on search head clusters: if an administrator grants you token-based access to a search head cluster, that token is valid for all search head cluster nodes.

You lose access to the instance through the token if any of the following occurs:

- The token expires. You can ask the administrator to issue a new token to regain access.
- The administrator disables the token. The administrator must reenable the token before you can use it again.
- The administrator deletes the token. You can ask the administrator to issue a new token to regain access.
- The administrator disables token authentication, either temporarily or permanently.
- The administrator deletes your account.
- You lock out or disable your Active Directory account on a Splunk platform instance that uses LDAP to connect to Active Directory.

Confirm that the administrator sends you the full token when they assign it to you. You must have the full token to authenticate with it. If you do not, the Splunk platform rejects those authentication requests as if you had entered incorrect credentials.

### Use an authentication token to make REST calls to an instance

After you receive a valid token for the instance you want to use, you can use the token to make calls to REST endpoints on the instance. Your access is the same as what you have if you logged in with credentials.

The type of web request you make depends on the endpoint you want to access and what you want to do with the endpoint. For more information on REST endpoints and how to use them, see the REST API User Manual and the REST API Reference Manual for information on REST and using REST endpoints.

Understand the ramifications of making REST calls using tokens. Tokens are only a method of authentication, not a safeguard against making potentially dangerous changes to a Splunk platform instance. If you have questions or concerns about using tokens, contact your administrator.

Currently, tokens are available only for making REST calls or using the CLI to access a Splunk Enterprise instance. You cannot use tokens to authenticate using Splunk Web.

Use your REST web client or command line tool to make an HTTP GET, POST, or DELETE request on an endpoint. When you make the request, include the management port of the instance:

```
curl -X GET -H "Authorization: Bearer <token>"  https://<instance host name or IP address>:<management port>/services/authentication/users/<user>
```

If the operation succeeds, the command prints a response in XML that contains the information that the instance generated upon accessing the endpoint.

An authentication failure appears as follows:
Insufficient permissions to access the endpoint appear as follows:

Use the CLI to access a Splunk Enterprise instance with a token

You can run commands with the Splunk CLI and use a token as authentication rather than standard credentials. You can do this with any command that normally requires authentication.

Use the \-token CLI argument to supply the token:

```
/opt/splunk/bin/splunk search '*' -token <token>
```

Examples

The following examples assume that you have properly configured SSL certificates on your Splunk platform instance. If you have not, then the commands will not work. As a temporary measure to test connectivity to SSL, you can use the \-k argument for \curl to bypass certificate review. Do not do this long term, as certificates are a key part of maintaining security in SSL.

**Examples for accessing REST endpoints using tokens**

The following example accesses the /apps/local endpoint to retrieve a list of local Splunk applications on the instance:

```
```

...
The following example accesses the `authorization/roles` endpoint to get information about the roles that have been configured on the instance:

```
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
xmlns:s="http://dev.splunk.com/ns/rest"
xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
><atom:title>roles</atom:title>
<atom:updated>2019-02-13T23:10:41+00:00</atom:updated>
<atom:generator build="7d447048a453" version="7.3.0"/>
<atom:author>
<atom:name>Splunk</atom:name>
</atom:author>
<atom:link href="/services/authorization/roles/_new" rel="create"/>
<opensearch:totalResults>5</opensearch:totalResults>
<opensearch:itemsPerPage>30</opensearch:itemsPerPage>
<opensearch:startIndex>0</opensearch:startIndex>
<s:messages/>
<atom:entry>
<atom:title>admin</atom:title>
...</atom:entry>
...</atom:feed>
```

**Examples for using the CLI with tokens**

The following example searches `splunkd.log` on the local instance for the string `JsonWebToken`:

```
./splunk search "index=_internal source=*splunkd.log* JsonWebToken" -token eyJraWQ...
```

```
09-18-2019 15:48:25.853 -0700 INFO JsonWebToken - Saving tokenId: f61b7aefc75e95cd226234f4fe901ed450fa323648165a91bf20f0a070c5199eb, authType: Splunk, IDP: Splunk to KVStore
```

The following example shows all of the apps that are present on the Splunk platform instance:

```
./splunk display app -token eyJraWQ...
```

```
alert_logevent CONFIGURED ENABLED INVISIBLE
alert_webhook CONFIGURED ENABLED INVISIBLE
appsbrowser CONFIGURED ENABLED INVISIBLE
```
## Troubleshoot token authentication

If a token fails authentication for any reason, Splunk Enterprise writes a message to `splunkd.log` with additional information. As a Splunk administrator, you can read this log file to get information on why authentication with the token failed.

For additional information, you can enable debug logging. Splunk Enterprise writes information about token authentication using the `JsonWebTokenHandler` tag. See Enable debug logging for instructions. After you have enabled debug logging, look for this tag when you review logs for information on problems that occur with token authentication.

### Common problems for token authentication

Following are a list of common problems that can occur with token authentication.

**Splunk instance displays "Token authentication is disabled"**

If you receive this error message, either in Splunk Web or through a REST command, it means that you have not enabled token authentication.

- Confirm that you have completed the requirements for enabling token authentication.
- Enable token authentication.

**cURL command returns "call not properly authenticated"**

This message means that authentication to the Splunk platform instance with the token you presented was not successful.

- Confirm that the token is enabled. If it is not, and it has not yet expired, enable it if you have permission, or contact your administrator.
- Confirm that the token is valid and has not expired. If it has expired, create a new one if you have permission, or contact your administrator. You cannot extend token validity.
- Confirm that the "Not before" validity time for the token has passed. If it hasn't, either wait or create a new token if you have permission.
- Confirm that the token has not been deleted. If it has, create a new one if you have permission.
- Confirm that the account that is associated with the token exists. If it doesn't, create one, then create a new token and assign that user to the token, if you have permission.
- Confirm that you use the full token as it was generated. If you don't have the full token, request or create a new one, if you have permission.
- Confirm that you are using a token on the same Splunk platform instance where it was issued.
• If your Splunk platform instance uses an LDAP server for authentication, confirm that the user exists and is not disabled on LDAP server.
• If your Splunk instance uses an LDAP server for authentication, confirm that the instance can connect to the LDAP server.

_Error received “KV store not ready”_

This message means that app key value store (KV store) has not been enabled. Enable KV store if you have permission, or contact your administrator.
Authentication with LDAP

Set up user authentication with LDAP

Splunk Enterprise supports three types of authentication systems:

• Native Splunk authentication, as described in “Set up user authentication with Splunk's built-in system.”
• Lightweight Directory Access Protocol (LDAP), as described in this topic.
• A scripted authentication API for use with an external authentication system, such as PAM or RADIUS, as described in “Set up user authentication with external systems.”

About configuring LDAP authentication for Splunk Enterprise

Splunk Enterprise allows user and role configuration for LDAP users and groups. You can configure one or many LDAP servers and map users and user groups from your servers to Splunk roles. You can also configure authentication tokens for LDAP users.

For more information about configuring multiple LDAP servers, see “How Splunk works with multiple LDAP servers.”

Before you configure LDAP, take a look at “LDAP prerequisites and considerations.”

After you configure LDAP as an authentication scheme, see Set up authentication with tokens for more information on creating authentication tokens for LDAP users.

How to configure LDAP authentication

These are the main steps to configure Splunk Enterprise to work with LDAP:

1. Configure one or more LDAP strategies (typically, you configure one strategy per LDAP server).
2. Map your LDAP groups to one or more Splunk roles.
3. If you have multiple LDAP servers, specify the connection order of their servers.

You can perform these steps in Splunk Web or by editing the configuration file. See “Configure LDAP with Splunk Web” or “Configure LDAP with the configuration file” for more information.

Authentication precedence

Native Splunk authentication takes precedence over any external schemes. This is the order in which Splunk Enterprise authenticates a user:

1. Splunk Enterprise attempts native authentication first. If the account is expired or otherwise fails, there is no follow up LDAP login attempt.
2. If a local user does not exist, Splunk Enterprise attempts an LDAP login or a scripted authentication, if that is enabled. For more information about scripted authentication, see “Set up user authentication with external systems.”

Answers

Have questions? Visit Splunk Answers and see what questions and answers the Splunk community has around LDAP authentication with Splunk.
Manage Splunk user roles with LDAP

To configure Splunk Enterprise to use LDAP authentication, first create a Splunk strategy for each LDAP server and then map Splunk roles to that server's groups. When a user attempts to log in, Splunk Enterprise queries the server(s) to find the user. It grants the user permissions based on any roles associated with the LDAP groups the user is a member of.

When it comes to changing a user's permissions, you have several options:

- To change the permissions for a group of users, you can remap the LDAP group to a different Splunk role. You can also update the role itself to specify a different set of permissions for it. You do this on Splunk Enterprise.
- To change the permissions for an individual user, you can move the user to an LDAP group mapped to a different Splunk role. You do this on the LDAP server.

Here are some other user management activities:

- To add a user to a Splunk role: First, on Splunk Web, make sure that you've mapped the Splunk role to an LDAP group. Then, on your LDAP server, add the user to that LDAP group.
- To remove a user from a Splunk role: On your LDAP server, remove the user from the corresponding LDAP group.

A user can have membership in several roles. In that case, the user has access to all the capabilities available for any of those roles. For example, if the user is a member of both the docs and eng groups, and docs is mapped to "user" and eng is mapped to "admin", the user obtains all permissions assigned to both the "user" or "admin" roles.

**Note:** Splunk Enterprise checks LDAP membership information when a user attempts to log in. You do not need to reload the authentication configuration when adding or removing users.

LDAP prerequisites and considerations

Before configuring LDAP for authentication with Splunk, make the preparations described in this topic.

**Determine your User and Group Base DN**

Before you map your LDAP settings to Splunk settings, figure out your user and group base DN, or distinguished name. The DN is the location in the directory where authentication information is stored.

If group membership information for users is kept in a separate entry, enter a separate DN identifying the subtree in the directory where the group information is stored. Users and groups will be searched recursively on all the subnodes under this DN. If your LDAP tree does not have group entries, you can set the group base DN to the same as the user base DN to treat users as their own group. This requires further configuration, described later.

If you are unable to get this information, contact your LDAP Administrator for assistance.

**Note:** For best results when integrating Splunk Enterprise with Active Directory, place your Group Base DN in a separate hierarchy than the User Base DN.

**Additional considerations**

When configuring Splunk Enterprise to work with LDAP, note the following:
• Entries in Splunk Web and authentication.conf are case sensitive.
• Any user created locally through Splunk native authentication will have precedence over an LDAP user of the same name. For example, if the LDAP server has a user with a username attribute (for instance, cn or uid) of 'admin' and the default Splunk user of the same name is present, the Splunk user will win. Only the local password will be accepted, and upon login the roles mapped to the local user will be in effect.
• The number of LDAP groups Splunk Web can display for mapping to roles is limited to the number your LDAP server can return in a query. You can use the Search request size limit and Search request time limit settings to configure this.
  ∗ To prevent Splunk from listing unnecessary groups, use the groupBaseFilter. For example:
  
  ```
  groupBaseFilter = ((cn=SplunkAdmins)(cn=SplunkPowerUsers)(cn=Help Desk))
  ```
  ∗ If you must role map more than the maximum number of groups, you can edit authentication.conf directly. In this example, "roleMap_AD" specifies the name of the Splunk strategy. Each attribute/value pair maps a Splunk role to one or more LDAP groups:

  ```
  [roleMap_AD]
  admin = SplunkAdmins1;SplunkAdmins2
  power = SplunkPowerUsers
  user = SplunkUsers
  ```

• Splunk always uses LDAP protocol version 3, aka v3.

Secure LDAP with TLS certificates

Splunk uses OpenLDAP and OpenSSL. You can leverage both tools to secure your LDAP authentication with certificates. For more information on creating and managing certificates, see the OpenSSL documentation.

The following examples are certificate configurations for LDAP. For more information about ways you can configure certificates in LDAP, see the OpenLDAP documentation at http://www.openldap.org/doc/admin24/tls.html:

**LDAP server configuration**

TLSCACertificateFile <filename>: the PEM-format file containing certificates for the CA's that slapd will trust, including the certificate for the CA that signed the server certificate. Multiple certificates can be appended to the file in no particular order.

TLSCertificateKeyFile <filename>: the file that contains the private key that matches the certificate stored in the TLSCertificateFile file.

TLSCipherSuite <cipher-suite-spec>: ciphers will be accepted and the preference order. <cipher-suite-spec> should be a cipher specification for OpenSSL. Use "openssl ciphers -v ALL" for a list of available cipher specifications.

TLSRandFile <filename>: the file to obtain random bits from when /dev/urandom is not available. If the system provides /dev/urandom then this option is not needed, otherwise a source of random data must be configured.

TLSEphemeralDHParamFile <filename>: the file that contains parameters for Diffie-Hellman ephemeral key exchange.

TLSVerifyClient { never | allow | try | demand }: specifies what checks to perform on client certificates in an incoming TLS session, if any. This option is set to never by default, in which case the server never asks the client for a certificate.
LDAP client configuration

This directive specifies the file that contains the client certificate. This is a user-only directive and can only be specified in a user's .ldaprc file.

TLS_KEY <filename> specifies the file that contains the private key that matches the certificate stored in the TLS_CERT file. The same constraints mentioned for TLSCertificateKeyFile apply here. This is also a user-only directive.

TLS_RANDFILE <filename> the same as the server's TLSRandFile option.

TLS_REQCERT { never | allow | try | demand }

Note that if you host two or more LDAP servers, you may not want to use self-signed certificates, since each client will have to be configured to work with each certificate. In such a case it would be easier to create a certificate authority to sign your server certificates.

How Splunk Enterprise works with multiple LDAP servers

Splunk Enterprise can search against multiple LDAP servers when authenticating users. To configure multiple LDAP servers, you set up multiple LDAP "strategies," one for each LDAP server.

After you create your strategies, you can specify the order in which you want Splunk Enterprise to query the strategies when searching for LDAP users. If you do not specify a search order, Splunk Enterprise assigns a default "connection order" based on the order in which the strategies are created.

For more about the steps to configure LDAP strategies, see Configure LDAP with Splunk Web or Configure LDAP with the configuration file for more information.

How connection order works during a search

During authentication, Splunk Enterprise searches based on the strategies you created for your servers in the specified connection order. After Splunk Enterprise locates the user on a server, it stops searching and takes those credentials. If the user also has credentials on a server later in the search order, those credentials are ignored.

For example, assume that you configure and enable three strategies in this order: A, B, C. Splunk Enterprise will search the servers in that same order: A, B, C. If it finds the user on A, it stops looking. Even if the user also exists on B and C, Splunk Enterprise will only use A's credentials for that user. If Splunk Enterprise does not find the user on A, it searches the remaining servers: first B, then C.

If you later disable strategy A, Splunk Enterprise will search the remaining strategies in the order: B, C.

You can change the connection order at any time by editing the strategies' properties in Splunk Web or by changing the order of the strategies in the authSettings attribute, as described in the authentication.conf spec file. For more information about editing this file for LDAP, see Edit authentication.conf.

Important: Any user created locally through Splunk authentication has precedence over an LDAP user of the same name. See About user authentication, for details.
Configure LDAP with Splunk Web

This section describes how to configure LDAP through Splunk Web. If you want to configure LDAP by directly editing authentication.conf, see Configure LDAP with the configuration file.

There are three main steps to configuring LDAP with Splunk Web:

1. Create an LDAP strategy.
2. Map LDAP groups to Splunk roles.
3. Specify the connection order (for multiple LDAP servers only)

Create an LDAP strategy

To create an LDAP strategy:

1. Click Settings > Users and authentication > Access controls.
2. Click Authentication method.
3. Check LDAP.
4. Click Configure Splunk to use LDAP and map groups. This takes you to the LDAP strategies page.
5. Click New. This takes you to the Add new page.
6. Enter an LDAP strategy name for your configuration.
7. Enter the Host name of your LDAP server. Be sure that your Splunk Server can resolve the host name. Note: At this time, IPv6 address formats for Windows are not supported.
8. Enter the Port that Splunk Enterprise will use to connect to your LDAP server.
   - By default LDAP servers listen on TCP port 389.
   - LDAPS (LDAP with SSL) defaults to port 636.
9. To turn on SSL, check SSL enabled.
   - This setting is recommended for security.
   - You must also have SSL enabled on your LDAP server.
10. Enter the Bind DN.
    - This is the distinguished name used to bind to the LDAP server.
    - This is typically, but not necessarily, the administrator. This user needs to have read access to all LDAP user and group entries you want to retrieve.
    - Leave blank if anonymous bind is sufficient.
11. Enter and confirm the Bind DN password for the binding user.
12. Specify the **User base DN**. You can specify multiple user base DN entries by separating them with semicolons.
   - Splunk Enterprise uses this attribute to locate user information.
   - You must set this attribute for authentication to work.

13. Enter the **User base filter** for the object class you want to filter your users on.
   - This is recommended to return only applicable users. For example: (department=IT).
   - Default value is empty, meaning no user entry filtering.

14. Enter the **User name attribute** that contains the user name.
   - The username attribute cannot contain white spaces.
   - In Active Directory, this is typically `sAMAccountName`, but you can also authenticate on other attributes, like `cn`.
   - The value `uid` should work for most other configurations.

15. Enter the **Real name attribute** (common name) of the user.
   - Typical values are `displayName` or `cn` (common name).

16. Enter an **Email attribute**

17. Enter the **Group mapping attribute**.
   - This is the user attribute that group entries use to define their members.
   - The default is `dn` for active directory; set this attribute only if groups are mapped using some other attribute besides user DN.
   - For example, a typical attribute used to map users to groups is `dn`.

18. Enter the **Group base DN**. You can specify multiple group base DN entries by separating them with semicolons.
   - This is the location of the user groups in LDAP.
   - If your LDAP environment does not have group entries, you can treat each user as its own group:
     - Set `groupBaseDN` to the same value as `userBaseDN`. This means you will search for groups in the same place as users.
     - Next, set the `groupMemberAttribute` and `groupMappingAttribute` to the same attribute as `userNameAttribute`. This means the entry, when treated as a group, will use the username value as its only member.
     - For clarity, you should probably also set `groupNameAttribute` to the same value as `userNameAttribute`.

   **Note:** For best results when integrating Active Directory, place your Group Base DN in a separate hierarchy than the User Base DN.

19. Enter the **Static group search filter** for the object class you want to filter your static groups on.
   - This is recommended to return only applicable groups. For example:
     - `(|(objectclass=groupofNames)(objectclass=groupofUniqueNames))`
   - Default value is empty, meaning no static group entry filtering.

20. Enter the **Group name attribute**.
   - This is the group entry attribute whose value stores the group name.
• This is usually cn.

21. Enter the **Static member attribute**.

• This is the group attribute whose values are the group's members.
• This is typically member, uniqueMember, or memberUid.

22. To expand nested groups, check **Nested groups**.

• This controls whether Splunk Enterprise will expand nested groups using the 'memberof' attribute. Only check this if you have nested groups that leverage the 'memberof' attribute to resolve their members. On OpenLDAP, you need to explicitly enable the 'memberof' overlay.

23. Enter the **Dynamic group search filter** to retrieve dynamic groups, if any.

• This must match the object class of your dynamic groups definition to ensure that those groups get returned to Splunk. For example: (objectclass=groupOfURLs)
• Default value is empty, meaning Splunk Enterprise will not look for dynamic group entries during authentication and authorization.

24. Enter the **Dynamic member attribute**.

• This is the group attribute that uses the form of an LDAP search URL (such as ldap://o=Acme, c=US??sub?(objectclass=person) ) to define its members.
• This is typically memberURL.

25. If you check **Advanced settings**, there are several additional options you can set:

• **Enable referrals with anonymous bind only.**
  ♦ This setting is on by default. Turn this off if you have no need for referrals.
  ♦ Splunk can chase referrals with anonymous bind only. You must also have anonymous search enabled on your LDAP server.
  ♦ If you are seeing long LDAP search timeouts (likely in Active Directory) and “Operations error” in splunkd.log for ScopedLDAPConnection, the issues might be related to referrals.

• **Search request size limit**
  ♦ To avoid performance-related issues, you can set the search request size limit. Splunk Enterprise will then request that the LDAP server return the specified maximum number of entries in response to a search request. In a large deployment with millions of users, setting this limit to a high value could result in a long response, depending on the search filter set in the LDAP strategy configuration. If this limit is reached, splunkd.log should contain a size limit exceeded message.
  ♦ You should set the **search request time limit** and **search request size limit** values in conjunction with the splunkweb timeout property, described in Configure user session timeouts. If you have a group that is not showing up in the Splunk console, it was likely excluded due to one of these limits. Tune these properties as needed.
  ♦ To set the request size limit higher than 1000, you must also edit max_users_to_precache in limits.conf to accommodate the number of users you set for your request size limit.

• **Search request time limit**
  ♦ To avoid performance-related issues, you can set the search request time limit. Splunk Enterprise will then request that the LDAP server complete its search within the specified number of seconds. In a large deployment with millions of users, setting this limit to a high value could cause Splunk Web to timeout. If this limit is reached, splunkd.log should contain a time limit exceeded message.
You should set the search request time limit and search request size limit values in conjunction with the splunkweb timeout property, described in Configure user session timeouts. If you have a group that is not showing up in the Splunk console, it was likely excluded due to one of these limits. Tune these properties as needed.

- **Network socket timeout**
  - This property is used to break the loop in the authentication chain when one of the LDAP servers in a multiple strategy configuration is unreachable due to network congestion or otherwise takes too long to respond. After waiting the specified number of seconds, the authentication process will continue with the next available strategy, if any.
  - When an LDAP strategy is first created, Splunk Enterprise validates the LDAP server/port and other parameters. If the LDAP server is down or one of the parameters cannot be validated at that time, the LDAP strategy does not get created.

26. Click Save.

**Map your new LDAP groups to Splunk roles**

Once you have configured Splunk Enterprise to authenticate via your LDAP server, map your LDAP groups to Splunk roles. If you do not use groups, you can map users individually.

**Note:** You can map either users or groups, but not both. If you are using groups, all users must be members of an appropriate group. Groups inherit capabilities from the highest level role they're a member of.

All users are visible in the Users page in Splunk Manager. To assign roles to groups in Splunk Web:

1. From the main menu, select System > Users and Authentication > Access Controls.

2. In the Access Controls page, click Authentication method.

3. Select the LDAP radio button then click Configure Splunk to use LDAP and map groups. This takes you to the LDAP strategies page.

4. Click Map groups in the Actions column for a specific strategy. This takes you to the LDAP Groups page. You can use the search field in the upper right corner of the page to qualify the list of groups; for example, to search for groups containing specific users.

5. Click on a group name. This takes you the mapping page, which includes a list of available roles and a list of LDAP users for that group.

6. To map a role to a group, click the arrow to the left of a role in the “Available Roles” list. This moves the group into the “Selected Roles” list. You can map multiple roles to the group.

7. Click Save. This takes you back to the LDAP Groups page.

8. Repeat the process for each group that you want to assign Splunk roles to.

**Specify the server connection order**

If you have enabled multiple LDAP strategies, you can specify the order in which Splunk Enterprise searches their servers to find a user, as described in How Splunk works with multiple LDAP servers.
By default, Splunk Enterprise searches the servers in the order in which they were enabled. To change the connection (search) order, you need to edit the properties for each strategy individually:

1. From the main menu, select System > Users and Authentication > Access Controls.

2. Click Authentication method.

3. Select the LDAP radio button.

4. Click Configure Splunk to use LDAP and map groups. This takes you to the LDAP strategies page.

5. Click on the strategy whose connection order you want to specify. This takes you to the properties page for that strategy.

6. Edit the Connection order field near the top of the page. This field appears only if multiple strategies are enabled.

Note: The Connection order field does not appear when you initially create the strategy. It only appears when you later edit its properties. Also, the field will be grayed out if the strategy has been disabled.

7. Click Save.

8. Repeat the process for any other enabled strategy whose connection order you want to change.

Map LDAP groups to Splunk roles in Splunk Web

If you have configured Splunk Enterprise to authenticate via your LDAP server, you can map your LDAP groups to Splunk roles. If you do not use groups, you can also map LDAP users individually.

For information about setting up LDAP groups in Splunk Web, see "Configure LDAP with Splunk Web" in this manual.

Note: You can map either users or groups, but not both. If you are using groups, all users you want to access Splunk Enterprise must be members of an appropriate group. Groups inherit capabilities from the highest level role they're a member of.

All users are visible in the Users page in Splunk Manager. To assign roles to groups in Splunk Web:

1. Click Settings in Splunk Web.

2. In the Users and authentication section, click Access controls.

3. Click Authentication method.

4. Select the LDAP radio button.

5. Click Configure Splunk to use LDAP and map groups. This takes you to the LDAP strategies page.

6. Click Map groups in the Actions column for a specific strategy. This takes you to the LDAP Groups page. You can use the search field in the upper right corner of the page to qualify the list of groups; for example, to search for groups containing specific users.
7. Click on a group name. This takes you the mapping page, which includes a list of available roles and a list of LDAP users for that group.

8. To map a role to a group, click the arrow to the left of a role in the “Available Roles” list. This moves the group into the “Selected Roles” list. You can map multiple roles to the group.

9. Click **Save**. This takes you back to the **LDAP Groups** page.

10. Repeat the process for each group that you want to assign Splunk roles to.

Configuring LDAP using configuration files

You can make changes to how Splunk Enterprise authenticates with servers that run the Lightweight Directory Access Protocol (LDAP). The authentication.conf file controls how Splunk Enterprise interacts with LDAP services for authentication.

Edit `authentication.conf` in `$SPLUNK_HOME/etc/system/local/`. For general information on editing configuration files, see [About configuration files](#) in the Admin Manual.

If you prefer to configure LDAP with Splunk Web, see [Configure LDAP with Splunk Web](#).

**How `authentication.conf` works with LDAP and LDAP strategies**

When you change the authentication scheme on the Splunk platform from native to LDAP, you must specify at least one LDAP strategy for the instance to connect to when it performs authentication.

`authentication.conf` represents this as the `authSettings` setting, where you specify at least one strategy, and a group of settings under a stanza for each strategy that you specify in the `authSettings` setting. The stanza names for any LDAP strategies you specify must match the names that you specified in `authSettings`. For example, if you configured `authSettings=ad_ldap`, then there must be a stanza called `ad_ldap` where the Splunk platform can look for settings and values for the `ad_ldap` strategy.

When you configure an LDAP strategy stanza, you must specify a minimum of the following settings and values:

- **host** = <LDAP server name>
- **port** = <LDAP port>
- **groupBaseDN** = <string representation of the group base Distinguished Name (DN), ex: `ou=Groups,dc=example,dc=com`>
- **groupMemberAttribute** = <group entry attribute whose values are the group members, ex: 'uniqueMember'>
- **groupNameAttribute** = <group entry attribute whose value stores the group name, ex: 'cn'>
- **realNameAttribute** = <user entry attribute whose value is their real name, ex: 'displayName' or 'cn'>
- **userBaseDN** = <string representation of user base DN, ex: `ou=People,dc=example,dc=com`>
- **userNameAttribute** = <user entry attribute whose value is the username, ex: 'uid'>

Either you or your LDAP administrator must provide the minimum setting values described here. There are additional settings that you can configure; see the `authentication.conf` specification file for those settings and their descriptions.

On Windows, there is no support for IPv6 address formats for the **host** setting.

For examples of how to create `authentication.conf`, see the `authentication.conf` spec file.
Configure multiple LDAP strategies

The Splunk platform can search across multiple LDAP servers, as described in How Splunk works with multiple LDAP servers. To configure multiple LDAP strategies, set the `authSettings` setting to a comma-separated list of all strategies, in the order in which you want to query the strategies. Then, specify separate stanzas for each strategy.

Set authentication type and configure LDAP strategy names and settings

This is a generic procedure for configuring authentication.conf for LDAP. Depending on your LDAP strategy settings, you might need to specify additional settings and values in the strategy-specific stanzas.

1. Open a shell prompt.
2. Change to the `$SPLUNK_HOME/etc/system/local` directory.
3. (Optional) Create the `authentication.conf` file if it does not already exist.
4. Open the `authentication.conf` file for editing.
5. Add the following lines to the file:

```
[authentication]
authType = LDAP
authSettings = <ldap_strategy1>,<ldap_strategy2>
```

In this example, `<ldap_strategy#>` represents one or more LDAP strategies. You can separate multiple strategies with commas.

6. Configure a stanza for each of the LDAP strategies that you specified in the `authSettings` setting, with the stanza name matching the strategy that you specified.

```
[authentication]
authType = LDAP
authSettings = ldap_strategy1
```

```
[ldap_strategy1]
```

7. Add a minimum of the following lines for each strategy stanza:

```
[ldap_strategy]
host = <LDAP server name>
port = <LDAP port>
groupBaseDN = <string representation of the group base Distinguished Name, ex: 
'ou=Groups,dc=example,dc=com'>
groupMemberAttribute = <group entry attribute whose values are the group members, ex: 
'uniqueMember'>
groupNameAttribute = <group entry attribute whose value stores the group name, ex: 'cn'>
realNameAttribute = <user entry attribute whose value is their real name, ex.: 'displayName' or 
'cn'>
userBaseDN = <string representation of user base Distinguished Name, ex: 
'ou=People,dc=example,dc=com'>
userBaseFilter = <user search filter, ex: (objectclass=*)>
userNameAttribute = <user entry attribute whose value is the username, ex: 'uid'>
```

8. Save the `authentication.conf` file and close it.
9. Restart the Splunk platform.

Set authentication type and configure LDAP strategies with SSL

If you have enabled SSL for your LDAP strategy, you must edit two files: `authentication.conf`, where you set the authentication type to LDAP and configure your LDAP strategy, and `ldap.conf`, where you configure the Splunk platform to use your SSL certificates to connect to your LDAP strategy.
You must also have already set up your SSL certificates on the instance where you want to use the LDAP authentication scheme. See About creating certificates for Splunk for additional information and procedures.

1. Complete steps 1 to 7 of the previous procedure, "Set authentication type and configure LDAP strategy names and settings".
2. For each LDAP strategy that you want to use SSL certificates to connect, add the following line in the strategy stanza:

   [ldap_strategy1]
   ... 
   sslEnabled = 1

3. Save the authentication.conf file and close it.
4. (Optional) Create the ldap.conf file if it does not already exist.
5. Add the following lines to the file:

   TLS_REQCERT demand
   TLS_CACERT <path to your SSL certificate, for example: /opt/splunk/etc/auth/LDAProotcert.crt>
   TLS_CIPHER_SUITE <your cipher suite>

6. Save the ldap.conf file and close it.
7. Restart the Splunk platform.

**Map LDAP groups to Splunk roles**

To map an LDAP strategy group to a Splunk roles, you must configure a roleMap stanza in authentication.conf for that strategy. Each strategy requires its own roleMap stanza. The following example maps LDAP groups in the "ldaphost1" strategy to Splunk roles. For each mapping, the syntax is <Splunk RoleName> = <LDAP group string>.

```
[roleMap_ldaphost1]
admin = SplunkAdmins
itusers = ITAdmins
```

**Map LDAP users to Splunk roles**

If you need to map an LDAP user directly to a Splunk role, in the stanza for the LDAP strategy where you want to do the mapping:

1. Set the value of the groupBaseDN setting to the value of the userBaseDN setting.
2. Set the values for the groupMappingAttribute, groupMemberAttribute, and groupNameAttribute settings to the same value as the userNameAttribute setting.

For example:

```
[authentication]
authType=LDAP
authSettings=supportLDAP

[supportLDAP]
SSLEnabled = 0
bindDN = cn=Directory Manager
bindDNpassword = ########
groupBaseDN = ou=People,dc=splunksupport,dc=com
groupBaseFilter = (objectclass=*)
groupMappingAttribute = uid
groupMemberAttribute = uid
groupNameAttribute = uid
```
host = supportldap.splunksupport.com
port = 389
realNameAttribute = cn
userBaseDN = ou=People,dc=splunksupport,dc=com
userBaseFilter = (objectclass=*)
userNameAttribute = uid

[roleMap_supportLDAP]
admin = rlee;bsmith

Disable LDAP authentication and return to native authentication

If you configure LDAP authentication and decide later to return to using the default Splunk authentication scheme, the fastest way is to rename authentication.conf (for example, by renaming it to authentication.conf.disabled) and restarting the Splunk platform.

Map LDAP groups and users to Splunk roles in the configuration files

Once you’ve set up LDAP authentication and users, you can map your LDAP groups and users to roles in Splunk Web. To set up LDAP for Splunk Enterprise, see Configure LDAP with the configuration file in this manual.

As an alternative to using Splunk Web to map roles, you can directly edit your authentication.conf contained in $SPLUNK_HOME/etc/system/local/. There are further examples at the end of the authentication.conf spec file.

For information on configuration files in general, see About configuration files In the Admin Manual.

Map groups to roles

To map Splunk roles to a strategy’s LDAP groups, you need to set up a roleMap stanza for that strategy. Each strategy requires its own roleMap stanza. This example maps roles for groups in the “ldaphost1” strategy. In your authentication.conf file in $SPLUNK_HOME/etc/system/local/:

[roleMap_ldaphost1]
admin = SplunkAdmins
itusers = ITAdmins

Map users directly to roles

If you need to map users directly to Splunk roles, you can do so by setting the groupBaseDN setting in authentication.conf to the value of userBaseDN.

Also set the following attributes to the same value as userNameAttribute:

- groupMappingAttribute
- groupMemberAttribute
- groupNameAttribute

For example:

[supportLDAP]
SSLEnabled = 0
bindDN - cn=Directory Manager
bindDNpassword = 80
groupBaseDN = ou=People,dc=splunksupport,dc=com
groupBaseFilter = (objectclass=*)
groupMappingAttribute = MyUserID
groupMemberAttribute = MyUserID
groupNameAttribute = MyUserID
host = supportldap.splunksupport.com
port = 389
realNameAttribute = cn
userBaseDN = ou=People,dc=splunksupport,dc=com
userBaseFilter = (objectclass=*)
userNameAttribute = MyUserID

[roleMap_supportLDAP]
admin = rlee;bsmith

Test your LDAP configuration

If you find that Splunk Enterprise is not able to connect to your LDAP server, try these troubleshooting steps:

1. Check $SPLUNK_HOME/var/log/splunk/splunkd.log for any authentication errors. Turn on DEBUG-level logging for AuthenticationManagerLDAP to get more information here. This can be done from the Splunk Web UI - Server Settings/Server Logging.

2. Remove any custom values you've added for userBaseFilter and groupBaseFilter.

3. In the *nix command line, you can use ldapsearch to confirm that the variables you are specifying will return the expected entries:

   ldapsearch -x ?h <ldap_host> ?p <ldap_port> ?D "bind_dn" -w "bind_passwd" -b "user_basedn" "userNameAttribute="

   ldapsearch -x ?h <ldap_host> ?p <ldap_port> ?D "bind_dn" -w "bind_passwd" ?b "group_basedn" "groupNameAttribute="

If these commands return matching entries, then your backend LDAP system is properly configured. Continue to troubleshoot the Splunk LDAP strategy configuration.

Convert to LDAP from Splunk authentication

If you move from Splunk authentication to LDAP, it's important to note that Splunk accounts are not automatically disabled and take precedence over LDAP accounts.

If you have converted from Splunk authentication system to LDAP, you might need to delete Splunk users to ensure that you're using LDAP credentials. This is only necessary if usernames are the same in both systems.

Secure local Splunk accounts

If you have configured Splunk Enterprise to use LDAP authentication, it's important to be aware that all local accounts using Splunk authentication are still present and active, including the "admin" account. You need to consider the security implications of this.

To remove all the current local accounts when enabling LDAP authentication:

- Move the $SPLUNK_HOME/etc/passwd file to passwd.bak.
• Create a blank $SPLUNK_HOME/etc/passwd file.
• Restart Splunk Enterprise.

Keep in mind that local Splunk accounts can still be created when Splunk Enterprise is in LDAP authentication mode. Also, any local Splunk accounts that must remain for backup or disaster-recovery purposes should use a very strong password.

When using LDAP, make sure that your LDAP implementation enforces:

• Strong password requirements for length and complexity.
• A low incorrect attempt threshold for password lockout.

Saved searches

If your LDAP usernames are the same as the names you previously used in the built-in system (but then deleted), saved searches should work without any conversion.

If you have existing saved searches created when your system was using Splunk authentication and you'd like to transfer them to an LDAP user of a different name, edit the metadata:

1. Modify $SPLUNK_HOME/etc/apps/<app_name>/metadata/local.meta and swap the owner = <username> field under each savedsearch permission stanza to the corresponding LDAP username and save your changes.
2. Restart Splunk Enterprise for your changes to take effect.

Best practice for removing an LDAP user

If you remove a user from your LDAP directory, Splunk Enterprise does not automatically remove the corresponding Splunk user. Usually this is not an issue, but if the user has global permissions of any sort, LDAP may generate errors.

To more information about working with LDAP users in Splunk Enterprise, see “Set up user authentication with LDAP” in this manual.

Take the following steps to safely remove a Splunk user:

1. First, back up the $HOME/splunk/etc/users/$userid folder.
2. Search the files under $HOME/splunk/etc/apps/ for the user id string to see if the user owns any searches or objects with global permissions.
3. For any searches or objects that the user owns, change the owner. You change it an admin user or maintenance account, or whatever you prefer.
4. Check splunkd.log on the search head to make sure there are no further LDAP authentication errors.
5. Once you have redirected any object ownership, you can safely remove the $HOME/splunk/etc/users/$userid folder.
Multi-factor authentication

About multifactor authentication with Duo Security

Multifactor authentication allows you to configure a primary and secondary login for your Splunk Enterprise users. Duo Security multifactor authentication secures Splunk Web logins. We recommend that you also secure your users with a firewall.

Splunk Cloud does not support multifactor authentication with Duo Security.

Splunk Enterprise with Duo Security multifactor authentication requires the user to set up a second authentication method and then use that method for future logins:

1. User logs into their Enterprise Splunk Web homepage using their login credentials. This is the primary login.

2. User then sees a second login page: “Duo Authentication”. This is the secondary login.

3. The first time a user logs in, they follow the instructions on the Duo login page to set up their preferred method for accessing their secondary credentials:
   - Login with credentials sent via a push notification on your your smart phone (Duo Security Mobile app required).
   - Login with credentials sent via SMS message sent to your cell phone.
   - Login with credentials sent via a phone call made to your cell phone.
   - Login by entering a one time code generated by the Duo Mobile app.

4. After the initial login and configuration, every time the user arrives at the secondary login, they receive those login credentials using their preferred method.

About setting up Duo Security for multifactor authentication in Splunk

1. Create an account for your Splunk Enterprise configuration on the Duo website. See https://duo.com for more information.

2. Provide Splunk Enterprise with the information from your Duo Security Account. See Configure Splunk to use Duo Security multifactor authentication for more information.

Configure Splunk Enterprise to use Duo Security multifactor authentication

**NOTE:** If you have previously configured Splunk Enterprise to use Duo authentication via https://duo.com/docs/splunk, you must use the task described in this topic to reconfigure multifactor login with Duo Security.

**Overview**

- Use the Duo Security website to create a Duo Security account for Splunk Enterprise. See https://duo.com for more information.
- Configure Splunk Enterprise to use Duo by providing the following information:
  - Your integration key (i.e. DIXXXXXXXXXXXXXXXXXX)
  - Your secret key
When the user logs into Splunk Enterprise and follows the instructions on the Duo login page, they are given secondary login credentials.

**Configure**

1. In the **Menu**, select **Settings > Users and Authentication > Access roles**.

2. Click **Authentication Method**.

3. Under **Multifactor Authentication**, select **Duo Security**.

4. Click the **Configure Duo Security** link.

5. Provide the **Integration Key** from your Duo configuration. You can find this key on your Duo Security configuration page or at **Configuration > Details**.

6. Provide the **Secret Key** from your Duo Security configuration or detail. You can find this key on your Duo Security configuration page or at **Configuration > Details**.

7. Provide the **API Hostname** from your Duo configuration. You can find this key on your Duo Security configuration page or at **Configuration > Details**.

8. **Tell Splunk Enterprise how to authenticate users when Duo Security is unavailable:**

   - **Let users login** Users who have successfully logged into the Splunk Web (i.e., primary authentication) can access Splunk Enterprise even if Duo authentication (i.e., secondary authentication) fails.
   - **Do not let users login** Users who have successfully logged into the Splunk Web (i.e., primary authentication) cannot access Splunk Enterprise if Duo authentication (i.e., secondary authentication) fails.

9. **Provide a time limit, in seconds, for how long authentication is attempted before the connection times out.**

10. **Save your changes. You do not need to reload authentication for multifactor authentication to take effect.**

Once a user logs in, the Duo login page appears, the user is instructed to choose a method to access their secondary login credentials.

**How multifactor authentication works with other forms of authentication**

Note that you cannot use any form of multi-factor authentication with SSO or SAML authentication. Multi-factor authentication works with the following sources of authentication:

- Native authentication
- LDAP
- Scripted authentication

**Configure Duo multifactor authentication for Splunk Enterprise in the configuration file**
In authentication.conf, edit the [2FA stanza name] stanza as follows:

[authentication]
externalTwoFactorAuthVendor = <Duo>
externalTwoFactorAuthSettings = <2FA stanza name>

[<2FA stanza name>]
integrationKey = <Integration Key as provided by Duo>
secretKey = <Secret Key as provided by Duo>
applicationKey = <Manually generated secret key>
apiHostname = <API Hostname as provided by Duo>
failOpen = True|False (Default : False)
timeout = <in seconds>

About multifactor authentication with RSA Authentication Manager

Multifactor authentication allows you to configure a primary and secondary login for your Splunk Enterprise users. You can configure multifactor authentication using RSA Authentication Manager for Splunk Web, REST endpoints, and CLI. Multifactor authentication secures the Splunk Enterprise web (8000) and management (8089) ports. After multifactor authentication is configured, the user enters a passcode to log in. The passcode is a combination of the user’s authentication PIN and the RSA-generated tokencode. For example, if the user's PIN is 1111 and RSA generates a tokencode of 2222, the passcode is 11112222. The tokencode may be generated from an RSA key fob or a mobile/desktop application.

Prerequisites

You need to have configured your RSA Authentication Manager before you attempt to configure RSA authentication on your Splunk Enterprise installation.

You need to have the change_authentication capability to configure multifactor authentication with RSA Authentication Manager.

Caveats

You cannot configure multifactor authentication in the following circumstances:

- REST endpoints authenticate via pass4symmkey.
- You have a configuration where there is a distributed search without index clustering where peers are added to the distsearch.conf file by entering the credentials of an admin user on the indexer. This is a one-time operation that is needed to push the search head's public key to the indexer.

How multifactor authentication works with other forms of authentication

Note that you cannot use any form of multifactor authentication with SSO or SAML authentication. Multifactor authentication works with the following sources of authentication:

- Native authentication
- LDAP
- Scripted authentication
Configure RSA authentication from Splunk Web

1. In the **Menu**, select **Settings > Users and Authentication > Access roles**.

2. Click **Authentication Method**.

3. Under **Multifactor Authentication**, select **RSA Security**.

4. Click the **Configure RSA Security** link.

5. Provide the RSA Auth Manager REST service URL.

6. Provide the Access key.

7. Tell Splunk Enterprise how to authenticate users when RSA Authentication Manager is unavailable:
   - **Let users login** Users who have successfully logged into Splunk Web (i.e., primary authentication) can access Splunk Enterprise even if RSA authentication (i.e., secondary authentication) fails.
   - **Do not let users login** Users who have successfully logged into Splunk Web (i.e., primary authentication) cannot access Splunk Enterprise if RSA authentication (i.e., secondary authentication) fails.

8. Provide an error/diagnostic message. This is the message you display if an error occurs when authenticating with RSA Authentication Manager.

9. Provide a time limit, in seconds, for how long to attempt authentication before the connection times out.

10. Save your changes. You do not need to reload authentication for two-factor authentication to take effect.

Before logging out of the configuration session, perform configuration verification using the **/services/admin/Rsa-MFA-config-verify** endpoint. This prevents you from blocking your ability to log in if you misconfigure authentication settings. If you connect to this endpoint without entering the passcode, this test can serve as a ping to ensure the services are running. Or, you can test the login for a user by including the username and passcode. For example, `curl -k -u admin:changed123 -X POST https://localhost:8089/services/admin/Rsa-MFA-config-verify/rsa-mfa -d username=user1 -d passcode=11112222`.

Configure Splunk Enterprise to use RSA Authentication Manager multifactor authentication via the REST endpoint

You can configure multifactor authentication via REST endpoints.

- To configure multifactor authentication for Splunk Web, you use the **/services/admin/Rsa-MFA** endpoint. To enable CLI and management port, set the parameter **enableMfaAuthRest** to true.
- To verify the authentication, you use the **/services/admin/Rsa-MFA-config-verify/** endpoint.

For details, see the REST API documentation for RSA_Multifactor_Authentication_REST_API_usage_details.
Two factor authentication when connecting to REST endpoints

When multifactor authentication is turned on for the management port, the user will need to pass the passcode encoded in the basic auth header to authenticate the user. The passcode is a combination of the user's authentication PIN and the RSA-generated tokencode. For example, if the user's PIN is 1111 and RSA generates a tokencode of 2222, the passcode is 11112222.

The following command shows the user passing the passcode 11112222 to authenticate:
```
```
You may also need to connect to the `auth/login` REST endpoint using the RSA tokencode to acquire a session key. For example:

```
curl -k https://localhost:8089/services/auth/login -d username=user1 -d password=changed -d passcode=11112222
<response>
<sessionKey>LfyRYqGEvt6aeOcotdBvqlBTDDG6jK674ExS6WqwhBYNnrZRFrZ8XbgP8h4_kbrMN4OJa2BF4dat7PjWb3E6QmmCYuGcXYJN4U1BzCEviqVKN</sessionKey>
<message code=""></message>
</response>
```

You cannot use the SDK to authenticate once the backend RSA multifactor authentication is enabled.

Configure Splunk Enterprise to use RSA Authentication Manager multifactor authentication in the configuration file

You can configure Splunk Enterprise to use multifactor authentication from the `authentication.conf` configuration file.

Configure multifactor authentication

In `authentication.conf`, edit the `[2FA stanza name]` stanza as follows:

```
[authentication]
externalTwoFactorAuthVendor = <RSA>
externalTwoFactorAuthSettings = <MFA stanza name>

[[MFA stanza name]]
authManagerUrl = <HTTPS-based URL of the RSA Authentication Manager REST endpoint>
accessKey = <Agent name created on RSA Authentication Manager>
failOpen = <True|False. True allows login in case the authentication server is unavailable.>
timeout = <Connection timeout in seconds for the outbound HTTPS connection>
messageOnError = <Message that will be shown on the error page if authentication fails>
sslVersions = <Comma-separated list of SSL versions to support for incoming connections. Supported versions are "ssl3", "tls1.0", "tls1.1", and "tls1.2". If this value is not set, Splunk uses default value of sslVersions=tls1.2>
cipherSuite = <Cipher string. Allows Splunk to use the specified cipher string for the HTTP server>
ecdhCurves = <Comma separated list of ec curves. Specify ECDH curves to use for ECDH key negotiation.>
sslVerifyServerCert = <True|False. True enables both the common and alternate names to be matched.>
sslCommonNameToCheck = <List of common names for outbound RSA HTTPS connections>
sslAltNameToCheck = <List of alternate names for outbound RSA HTTPS connections>
sslRootCAPath = <Root path. The path must refer to full path of a PEM format file containing one or more...>
```

87
Before logging out of configuration session, perform config verification using 'services/admin/Rsa-MFA-config-verify' endpoint. First, enter the factor credentials and then enter the RSA passcode that appears on the RSA key fob or the mobile/desktop application which generates the token. This prevents you from blocking your ability to log in if you misconfigure authentication settings.

User experience when logging into a Splunk instance configured with RSA multifactor authentication

Depending on the method of access -- Splunk Web, CLI, or REST endpoint, the user enters RSA credentials using the login steps described below.

User experience when logging into Splunk Web

From Splunk Web, Splunk Enterprise with RSA Authentication Manager multifactor authentication requires the user to log into a secondary page or secondary method for each login.

1. The user logs into their Enterprise Splunk Web homepage using their login credentials. This is the primary login.

2. The user then sees an RSA login page. This is the secondary login.

3. The user enters their passcode. The passcode is a combination of the user's authentication PIN and the RSA-generated tokencode. For example, if the user's PIN is 1111 and RSA generates a tokencode of 2222, the passcode is 11112222.

4. In certain circumstances, the user may be required to enter their next token code. The user should not include their PIN for this page.

Multifactor authentication is logged in the debug log. Check the debug log if users encounter issues logging in.

Splunk does not support New PIN mode. In this case, the user should use an RSA client to configure the new PIN.

User experience when logging in via the CLI

If you have configured multifactor authentication for the management port, the user will need to use their passcode when logging into the CLI. The passcode is a combination of the user's authentication PIN and the RSA-generated tokencode. To authenticate, the user runs the login command with the -auth parameter, and includes the passcode. For example, the following CLI sample shows the user logging in where the passcode is :676z=sl7.

```
./splunk login -auth user1:changed:676z=sl7
```

User experience when connecting to REST endpoints

If multifactor authentication is enabled for REST endpoints, and the user needs to access the UI endpoint login via https://localhost:8089/servicesNS/<user>/<app>. The user is prompted for a username and password. To login, the user will need to enter the password and passcode using the syntax of password:passcode to login.

The following command shows the user passing the passcode 11112222 to authenticate:

```
```
You may also need to connect to the `auth/login` REST endpoint using the RSA tokencode to acquire a session key. For example:

```
curl -k https://localhost:8089/services/auth/login -d username=user1 -d password=changed -d passcode=1111222
```

You cannot use the SDK to authenticate once the backend RSA multifactor authentication is enabled.
Authentication using single sign-on with SAML

Configure single sign-on with SAML

You can configure Splunk software to use SAML authentication for single sign-on (SSO), using information provided by your supported identity provider (IdP).

Splunk software always outputs usernames in lowercase. If your IdP expects Splunk software to preserve uppercase letters in usernames, you can change the username to lowercase in the IdP or configure the IdP to accept the lowercase version of a username.

Note that if the search head is restarted, you must re-enter your credentials to access saved searches.

Prerequisites

- Either:
  - A running version of Splunk software OR
  - A managed deployment of Splunk Cloud. Self-service deployments of Splunk Cloud log in through the Splunk customer portal and cannot independently configure SAML SSO.
- An identity provider configured to provide the role, realName, and mail attributes. The supported identity providers are:
  - Ping Identity
  - Okta
  - Azure AD
  - AD FS
  - OneLogin
  - Optimal
  - CA siteminder
- An admin role (Splunk Enterprise) or sc_admin role (Splunk Cloud) with the change_authentication capability. This permission level lets you enable SAML and edit authentication settings on the Splunk search head.

Other IdPs

Any SAML IdP that is v2 compliant should be configurable, including the following tested IdPs. For assistance with any IdP that is not documented in this chapter, contact Support:

- SecureAuth
- Novell Directory
- G Suite (Formerly Google Apps for Business)

Configure SAML to work with your IdP

SAML does not support encryption, regardless of IdP.

1. Configure SAML SSO with:
   - Ping Identity
   - Okta
   - Azure AD or AD FS
2. Map SAML groups to Splunk Enterprise roles.

**Configure SSO with PingIdentity as your identity provider**

This task describes how to set up SSO for Splunk Enterprise if you have configured PingIdentity as your Identity Provider (IdP). For information about configuring PingIdentity as an IdP, consult your Ping Federate documentation.

Use the following task to configure your Splunk deployment to recognize and use your PingIdentity configuration. Then, map groups of PingIdentity users to Splunk user roles so that those users can log in.

**Prerequisites**

Verify that your system meets all of the requirements. See Configure single sign-on with SAML.

As a best practice, the session timeout in splunk and on the IdP should be the same so that the sessions are invalidated simultaneously on Splunk and the IdP. For more information, see Configure session timeouts with Ping Identity

As a best practice, the session timeout in splunk and on the IdP should be the same so that the sessions are invalidated simultaneously on Splunk and the IdP.

1. In the **Settings** menu, select **Access Controls > Authentication method**.

2. Select **SAML** as your authentication type.

3. Click **Configure Splunk to use SAML**.

4. On the SAML Groups page, click **SAML Configuration**.

5. Download or browse and select your metadata file, or copy and paste your metadata directly into the text window. Refer to your IdP's documentation if you are not sure how to get your metadata file.

6. In **General Settings**, provide the following information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Sign on URL</td>
<td>This field is populated automatically by your selected metadata file. It is the protected endpoint on your IdP to which Splunk sends authentication requests. Your users use this URL for SSO login. To access the login page once SAML is enabled, append the full login URL (/account/login) with loginType=Splunk. Users can also log into their local Splunk account by navigating directly to splunkweb:port/en-US/account/login?loginType=Splunk.</td>
</tr>
<tr>
<td>Single Log Out URL</td>
<td>This field is populated automatically by the metadata file and is the IdP protocol endpoint. If you do not provide this URL, the user will not be logged out.</td>
</tr>
<tr>
<td>IdP certificate path</td>
<td>This value can be a directory or a file, depending on your IdP requirements. If you provide a file, Splunk software uses that file to validate the SAML response. If you provide a directory, Splunk software looks for all the certificates in that directory and tries to validate the SAML response with each one of them. If any validation fails, the response</td>
</tr>
</tbody>
</table>
is considered invalid.

<table>
<thead>
<tr>
<th>IdP certificate chains</th>
<th>If you use a certificate chain, order them as follows:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1. Root</td>
</tr>
<tr>
<td></td>
<td>2. Intermediate</td>
</tr>
<tr>
<td></td>
<td>3. Leaf</td>
</tr>
</tbody>
</table>

| Replicate certificates | Check this to replicate your IdP certificates in a search head cluster. When configuring SAML on a search head cluster, you must use the same certificate for each search head. |

<table>
<thead>
<tr>
<th>Issuer Id</th>
<th>This is the Entity Id of the IdP. See your IdP documentation if you are not sure where to find this information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity ID</td>
<td>This field is the entity ID as configured in the SP connection entry in your IdP.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sign AuthRequest.</th>
<th>Select this option.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sign SAML Response.</td>
<td>Select this option.</td>
</tr>
</tbody>
</table>

7. In **Attribute Query Requests**, optionally provide the following information so you can create scheduled searches later.

<table>
<thead>
<tr>
<th>Attribute Query URL.</th>
<th>(Optional) This is the endpoint on the IdP to which queries over SOAP are sent. The format is as follows: <code>&lt;urn:oasis:names:tc:SAML:2.0:attrname-format:uri&gt;</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sign attribute query request</td>
<td>Verify that this field is selected.</td>
</tr>
<tr>
<td>Sign attribute query response</td>
<td>Verify that this field is selected.</td>
</tr>
<tr>
<td>Username</td>
<td>Enter a user name.</td>
</tr>
</tbody>
</table>

| Provide a password. |                                                                 |

8. In the **Alias** section optionally provide the following aliasing information:

<table>
<thead>
<tr>
<th>Role Alias</th>
<th>Use this field to specify a new attribute name on any IdP and then configure an alias in your Splunk deployment for any of the three attributes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Name Alias</td>
<td>You may skip this field. For ADFS you can use the displayname for the Attribute Alias Real Name.</td>
</tr>
<tr>
<td>Mail Alias</td>
<td>Skip this field.</td>
</tr>
</tbody>
</table>

9. Populate the advanced section only if you need to set up load balancing or change the SAML binding. See **Configure advanced settings for SSO**.

10. Click **Save**.

**Next Step**

Map SAML groups to Splunk Enterprise roles
Configure SSO with Okta as your identity provider

This task describes how to set up SSO for Splunk Enterprise if you have configured Okta as your Identity Provider (IdP). For information about configuring Okta as an IdP, consult your Okta documentation.

Use the following task to configure your Splunk deployment to recognize and use your Okta configuration. Then, map groups from the IdP to Splunk user roles so that those groups can log in.

Prerequisites

Verify that your system meets all of the requirements. See Configure single sign-on with SAML.

1. In the **Settings** menu, select **Access Controls > Authentication method**.

2. Select **SAML** as your authentication type.

3. Click **Configure Splunk to use SAML**.

4. On the SAML Groups page, click **SAML Configuration**.

5. Download or browse and select your metadata file, or copy and paste your metadata directly into the text window. Refer to your Okta documentation if you are not sure how to locate your metadata file.

6. In **General Settings**, provide the following information:

<table>
<thead>
<tr>
<th><strong>Single Sign on URL</strong></th>
<th>This field is populated automatically by your selected metadata file. It is the protected endpoint on your IdP to which Splunk Enterprise sends authentication requests.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>To access the login page once SAML is enabled, append the full login URL (/saml/acs) with loginType=Splunk. Users can also log into their local Splunk account by navigating directly to ? splunkweb:port/en-US/account/login?loginType=Splunk</td>
</tr>
<tr>
<td><strong>Single Log Out URL</strong></td>
<td>This field is populated automatically by the metadata file and is the IdP protocol endpoint. If you do not provide this URL, the user will not be logged out.</td>
</tr>
<tr>
<td><strong>IdP's certificate path</strong></td>
<td>This value can be a directory or a single file, depending on your IdP requirements. If you provide a file, Splunk Enterprise uses that file to validate authenticity of SAML response. If you provide a directory, Splunk Enterprise looks at all the certificates in the directory and tries to validate SAML response with each one of them. If any validation fails, authentication fails.</td>
</tr>
<tr>
<td><strong>IdP certificate chains</strong></td>
<td>If you use a certificate chain, order them as follows:</td>
</tr>
<tr>
<td></td>
<td>1. Root</td>
</tr>
<tr>
<td></td>
<td>2. Intermediate</td>
</tr>
<tr>
<td></td>
<td>3. Leaf</td>
</tr>
</tbody>
</table>
Check this to replicate your IdP certificates in a search head cluster. When configuring SAML on a search head cluster, you must use the same certificate for each search head.

This is the Entity Id of the IdP. See your IdP documentation if you are not sure where to find this information.

This field is the entity ID as configured in the SP connection entry in your IdP.

Select this option.

Select this option.

If "Request Compression" is set, when you log onto Splunk Web on a Search Head, you are diverted to Okta Applications rather than the Search Head.

7. Skip Attribute Query and go to steps 8 and 9.

8. In the Alias section optionally provide the following aliasing information:

In Alias, provide the following information:

<table>
<thead>
<tr>
<th>Role Alias</th>
<th>Use this field to specify a new attribute name on any IdP and then configure an alias in your Splunk deployment for any of the three attributes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Name Alias</td>
<td>You may skip this field. For ADFS you can use the displayname for the Attribute Alias Real Name.</td>
</tr>
<tr>
<td>Mail Alias</td>
<td>Skip this field.</td>
</tr>
</tbody>
</table>

9. Populate the advanced section only if you need to set up load balancing or change the SAML binding. See Configure load balancing or SAML bindings

10. Click Save.

Next Step

Note: An error in configuring SAML could result in users and admins being locked out of Splunk Cloud. Use this link to access the local login and revert to None for authentication if you are locked out:

https://<name>.splunkcloud.com/en-US/account/login?loginType=splunk [replace <name> with your account name]

Map SAML groups to Splunk Enterprise roles

Configure SSO with AzureAD or AD FS as your Identity Provider

This task describes how to set up SSO for Splunk deployments if you have configured AzureAD or ADFS as your Identity Provider (IdP).

Prerequisites

Verify that your system meets all of the requirements. See Configure single sign-on with SAML.

SAML does not support encryption, regardless of IdP.
When configuring your IdP, note the following suggestions when configuring your groups:

- Before you begin, be sure to consult your IdP documentation and ensure that you have met the IdP configuration requirements.
- For AzureAD the reply URL may require `/SAML/acs` on the end.
- For AzureAD you may need to change the `groupMembershipClaims` from `null` to `SecurityGroup`.
- For AD FS for Splunk Cloud, you may need to set the Claim Type as "UPN" when configuring your IdP. The Splunk blog post at https://www.splunk.com/blog/2016/09/14/configuring-microsofts-adfs-splunk-cloud.html provides more information about configuring AD FS for Cloud.

Use the following task to configure your Splunk deployment to recognize and use your AzureAD or AD FS configuration. Then, map groups of AzureAD or AD FS users to Splunk user roles so that those users can log in.

**Configure Splunk Software for SAML**

1. In the **Settings** menu, select **Access Controls > Authentication method**.
2. Select **SAML** as your authentication type.
3. Click **Configure Splunk to use SAML**.
4. On the SAML Groups page, click **SAML Configuration**.
5. Download or browse and select your metadata file, or copy and paste your metadata directly into the text window. Refer to your IdP's documentation if you are not sure how to get your metadata file.
6. In **General Settings**, provide the following information.

<table>
<thead>
<tr>
<th>Single Sign on URL</th>
<th>This field is populated automatically by your selected metadata file. It is the protected endpoint on your IdP to which Splunk Enterprise sends authentication requests. Your users also use this URL for SSO login. To access the login page once SAML is enabled, append the full login URL (<code>/account/login</code>) with <code>loginType=Splunk</code>. Users can also log into their local Splunk account by navigating directly to <code>splunkweb:port/en-US/account/login?loginType=Splunk</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Log Out URL</td>
<td>This field is populated automatically by the metadata file and is the IdP protocol endpoint. If you do not provide this URL, the user will not be logged out.</td>
</tr>
<tr>
<td>IdP certificate path</td>
<td>This value can be a directory or a file, depending on your IdP requirements. If you provide a file, Splunk software uses that file to validate authenticity of SAML response. If you provide a directory, Splunk looks at all the certificates in the directory and tries to validate SAML response with each one of them. If validation fails, authentication fails.</td>
</tr>
<tr>
<td>IdP certificate chains</td>
<td>If you use a certificate chain, order them as follows:</td>
</tr>
<tr>
<td></td>
<td>1. Root</td>
</tr>
<tr>
<td></td>
<td>2. Intermediate</td>
</tr>
<tr>
<td></td>
<td>3. Leaf</td>
</tr>
<tr>
<td>Replicate certificates</td>
<td>Check this to replicate your IdP certificates in a search head cluster. When configuring SAML on a search head cluster, you must use the same certificate for each search head.</td>
</tr>
</tbody>
</table>
7. Skip the **Attribute Query** section and go to steps 8 and 9.

8. In the **Alias** section optionally provide the following aliasing information:

| Role Alias | Use this field to specify a new attribute name on any IdP and then configure an alias in your Splunk deployment for any of the three attributes. |
| Real Name Alias | You may skip this field. For ADFS you can use the displayname for the Attribute Alias Real Name. |
| Mail Alias | Skip this field. |

9. Populate the advanced section only if you need to set up load balancing or change the SAML binding. See Configure load balancing or SAML bindings.

10. Click **Save**.

**Next Step**

**Note:** An error in configuring SAML could result in users and admins being locked out of Splunk Cloud. Use this link to access the local login and revert to None for authentication if you are locked out:

https://<name>.splunkcloud.com/en-US/account/login?loginType=splunk [replace <name> with your account name]

**Note 2:** You may experience an issue with Splunk continuously refreshing/re-authenticating if the Splunk Web session timeout is different than your IDP vendor session timeout (i.e. Splunk web session timeout: 1 hour and IDP vendor session timeout: 9 hours). This issue may cause users to lose work. To correct the issue you should set the Splunk web session timeout to be equal to the IDP vendor session timeout.

**Map SAML groups to Splunk Enterprise roles**

**Configure SSO with OneLogin as your identity provider**

This task describes how to set up SSO for Splunk if you have configured OneLogin as your Identity Provider (IdP). For information about configuring OneLogin as an IdP, consult your OneLogin documentation.

Use the following task to configure Splunk to recognize and use your OneLogin configuration. Then, map groups of OneLogin users to Splunk user roles so that those users can log into Splunk.

**Prerequisites**

Verify that your system meets all of the requirements. See Configure single sign-on with SAML.
1. In the **Settings** menu, select **Access Controls > Authentication method**.

2. Select **SAML** as your authentication type.

3. Click **Configure Splunk to use SAML**.

4. On the SAML Groups page, click **SAML Configuration**.

5. Download or browse and select your metadata file, or copy and paste your metadata directly into the text window. Refer to your IdP's documentation if you are not sure how to get your metadata file.

6. In **General Settings**, provide the following information.

<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Single Sign on URL</strong></td>
<td>This field is populated automatically by your selected metadata file. It is the protected endpoint on your IdP to which Splunk sends authentication requests. To access the login page once SAML is enabled, append the full login URL (/account/login) with loginType=Splunk. Users can also log into their local Splunk account by navigating directly to /account/login?loginType=Splunk.</td>
</tr>
<tr>
<td><strong>Single Log Out URL</strong></td>
<td>OneLogin supports redirect binding for single log out. Set the binding to 'HTTPRedirect'.</td>
</tr>
<tr>
<td><strong>IdP's certificate path</strong></td>
<td>This value can be a directory or a file, depending on your IdP requirements. If you provide a file, Splunk uses that file to validate authenticity of SAML response. If you provide a directory, Splunk looks for all the certificates that are present as children of the directory and tries to validate SAML response with each one of them, if Splunk fails to validate authenticity with all of them, response is not considered authentic.</td>
</tr>
<tr>
<td><strong>IdP certificate chains</strong></td>
<td>If you use a certificate chain, order them as follows: 1. Root 2. Intermediate 3. Leaf</td>
</tr>
<tr>
<td><strong>Replicate certificates</strong></td>
<td>Check this to replicate your IdP certificates in a search head cluster. When configuring SAML on a search head cluster, you must use the same certificate for each search head.</td>
</tr>
<tr>
<td><strong>Issuer Id</strong></td>
<td>This is the Entity Id of the IdP. See your IdP documentation if you are not sure where to find this information.</td>
</tr>
<tr>
<td><strong>Entity ID</strong></td>
<td>This field is the entity ID as configured in the SP connection entry in your IdP.</td>
</tr>
<tr>
<td><strong>Sign AuthRequest</strong></td>
<td>ISet this value to false.</td>
</tr>
<tr>
<td><strong>Sign SAML Response</strong></td>
<td>Set this value to false.</td>
</tr>
</tbody>
</table>

7. Skip the **Attribute Query** section and go to steps 8 and 9.

8. In the **Alias** section optionally provide the following aliasing information:

<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th>Description</th>
</tr>
</thead>
</table>
| **Role Alias** | Use this field to specify a new attribute name on any IdP and then configure an alias in your Splunk deployment for any of the three attributes.  
You may skip this field. For ADFS you can use the displayname for the Attribute Alias Real Name. |
9. Populate the advanced section only if you need to set up load balancing or change the SAML binding. See Configure load balancing or SAML bindings.

10. Click Save.

**Note:** An error in configuring SAML could result in users and admins being locked out of Splunk Cloud. Use this link to access the local login and revert to None for authentication if you are locked out:

https://<name>.splunkcloud.com/en-US/account/login?loginType=splunk [replace <name> with your account name]

**Next Step**

Map SAML groups to Splunk Enterprise roles

---

**Configure SSO with Optimal as your identity provider**

This task describes how to set up SSO for Splunk if you have configured Optimal as your Identity Provider (IdP). For information about configuring Optimal as an IdP, consult your Optimal documentation.

Use the following task to configure Splunk to recognize and use your Optimal configuration. Then, map groups of Optimal users to Splunk user roles so that those users can log into Splunk.

**Prerequisites**

Verify that your system meets all of the requirements. See Configure single sign-on with SAML.

1. In the **Settings** menu, select **Access Controls > Authentication method**.

2. Select SAML as your authentication type.

3. Click **Configure Splunk to use SAML**.

4. On the SAML Groups page, click **SAML Configuration**.

5. Download or browse and select your metadata file, or copy and paste your metadata directly into the text window. Refer to your IdP’s documentation if you are not sure how to get your metadata file.

6. In **General Settings**, provide the following information.

| Single Sign on URL | This field is populated automatically by your selected metadata file. It is the protected endpoint on your IdP to which Splunk sends authentication requests. If you are using Splunk Cloud, open a support ticket to have the Splunk Cloud operations team open the port for communicating with the IdP. Your users use this URL for SSO login. To access the login page once SAML is enabled, append the full login URL (/account/login) with loginType=Splunk. Users can also log into their local Splunk account by navigating directly |

---

98
<table>
<thead>
<tr>
<th>Single Log Out URL</th>
<th>This field is populated automatically by the metadata file and is the IdP protocol endpoint. If you do not provide this URL, the user will not be logged out.</th>
</tr>
</thead>
<tbody>
<tr>
<td>IdP's certificate path</td>
<td>This value can be a directory or a file, depending on your IdP requirements. If you provide a file, Splunk uses that file to validate authenticity of SAML response. If you provide a directory, Splunk looks for all the certificates that are present as children of the directory and tries to validate SAML response with each one of them, if Splunk fails to validate authenticity with all of them, response is not considered authentic.</td>
</tr>
</tbody>
</table>
| IdP certificate chains | If you use a certificate chain, order them as follows:  
1. Root  
2. Intermediate  
3. Leaf |
| Replicate certificates | Check this to replicate your IdP certificates in a search head cluster. When configuring SAML on a search head cluster, you must use the same certificate for each search head. |
| Issuer Id | This is the Entity Id of the IdP. See your IdP documentation if you are not sure where to find this information. |
| Entity ID. | This field is the entity ID as configured in the SP connection entry in your IdP. |
| Sign AuthRequest. | Select this option. |
| Sign SAML Response. | Select this option. |

7. Skip the **Attribute Query** section and go to steps 8 and 9.

8. In the **Alias** section optionally provide the following aliasing information:

| Role Alias | Use this field to specify a new attribute name on any IdP and then configure an alias in your Splunk deployment for any of the three attributes. |
| Real Name Alias | You may skip this field. For ADFS you can use the displayname for the Attribute Alias Real Name. |
| Mail Alias | Skip this field. |

9. Populate the advanced section only if you need to set up load balancing or change the SAML binding. See Configure load balancing or SAML bindings

10. Click **Save**.

Note: An error in configuring SAML could result in users and admins being locked out of Splunk Cloud. Use this link to access the local login and revert to None for authentication if you are locked out:

https://<name>.splunkcloud.com/en-US/account/login?loginType=splunk [replace <name> with your account name]

**Next Step**

Map SAML groups to Splunk Enterprise roles
# Configure SSO in CA Siteminder

This task describes how to set up SSO for Splunk if you have configured CA as your Identity Provider (IdP). For information about configuring CA as an IdP, consult your CA documentation.

Use the following task to configure Splunk to recognize and use your CA configuration. Then, map groups of CA users to Splunk user roles so that those users can log into Splunk.

## Prerequisites

Verify that your system meets all of the requirements. See Configure single sign-on with SAML.

1. In the **Settings** menu, select **Access Controls > Authentication method**.

2. Select **SAML** as your authentication type.

3. Click **Configure Splunk to use SAML**.

4. On the SAML Groups page, click **SAML Configuration**.

5. Download or browse and select your metadata file, or copy and paste your metadata directly into the text window. Refer to your IdP's documentation if you are not sure how to get your metadata file.

6. In **General Settings**, provide the following information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Single Sign on URL</strong></td>
<td>This field is populated automatically by your selected metadata file. It is the protected endpoint on your IdP to which Splunk sends authentication requests. To access the login page once SAML is enabled, append the full login URL (/account/login) with <code>?loginType=Splunk</code>. Users can also log into their local Splunk account by navigating directly to <code>?splunkweb:port/en-US/account/login?loginType=Splunk</code></td>
</tr>
<tr>
<td><strong>Single Log Out URL</strong></td>
<td>This field is populated automatically by the metadata file and is the IdP protocol endpoint. If you do not provide this URL, the user will not be logged out.</td>
</tr>
<tr>
<td><strong>IdP's certificate path</strong></td>
<td>This value can be a directory or a file, depending on your IdP requirements. If you provide a file, Splunk uses that file to validate authenticity of SAML response. If you provide a directory, Splunk looks for all the certificates that are present as children of the directory and tries to validate SAML response with each one of them, if Splunk fails to validate authenticity with all of them, response is not considered authentic.</td>
</tr>
<tr>
<td><strong>IdP certificate chains</strong></td>
<td>If you use a certificate chain, order them as follows:</td>
</tr>
<tr>
<td></td>
<td>1. Root</td>
</tr>
<tr>
<td></td>
<td>2. Intermediate</td>
</tr>
<tr>
<td></td>
<td>3. Leaf</td>
</tr>
<tr>
<td><strong>Replicate certificates</strong></td>
<td>Check this to replicate your IdP certificates in a search head cluster. When configuring SAML on a search head cluster, you must use the same certificate for each search head.</td>
</tr>
<tr>
<td><strong>Issuer Id</strong></td>
<td>This is the Entity Id of the IdP. See your IdP documentation if you are not sure where to find this information.</td>
</tr>
<tr>
<td><strong>Entity ID</strong></td>
<td>This field is the entity ID as configured in the SP connection entry in your IdP.</td>
</tr>
</tbody>
</table>
7. Skip the **Attribute Query** and go to steps 8 and 9.

8. In the **Alias** section optionally provide the following aliasing information:

| Role Alias | Use this field to specify a new attribute name on any IdP and then configure an alias in your Splunk deployment for any of the three attributes. |
| Real Name Alias | You may skip this field. For ADFS you can use the displayname for the Attribute Alias Real Name. |
| Mail Alias | Skip this field. |

9. Populate the advanced section only if you need to set up load balancing or change the SAML binding. See Configure load balancing or SAML bindings

10. Click **Save**.

**Note:** An error in configuring SAML could result in users and admins being locked out of Splunk Cloud. Use this link to access the local login and revert to None for authentication if you are locked out:

https://<name>.splunkcloud.com/en-US/account/login?loginType=splunk [replace <name> with your account name]

**Next Step**

Map SAML groups to Splunk Enterprise roles

**Secure SSO with TLS certificates**

Configure the following SSL settings to enable Splunk Enterprise to perform TLS verification between Splunk Instance and the SOAP instance providing **AttributeQuery** service.

Unless noted, values not set default to the setting specified in `server.conf`.

```
[saml-authSettings-key]
sslVersions = <Comma-separated list of SSL versions to support>
sslCommonNameToCheck = <commonName> When populated, and sslVerifyServerCert is "true", splunkd limits most outbound HTTPS connections to hosts which use a cert with this common name.
sslAltNameToCheck = <alternateName1>, <alternateName2>, ...If set, and sslVerifyServerCert' is "true", splunkd can verify certificates with "Subject Alternate Name" that matches any of the is alternate names in this list.
ecdhCurveName = <ECDH curve to use for ECDH key negotiation>
```
serverCert - <Server certificate file> Default certificates, "server.pem" are auto-generated by splunkd upon starting Splunk, you may replace the default cert with your own PEM format file.

sslPassword - <Server certificate password>

cacertFile - <Public key of the signing authority> The default value is cacert.pem

capath - <Path where all these certs are stored>. Default value is $SPLUNK_HOME/etc/auth

sslversonceroot - [ true | false ] If true, distributed search makes a search request to another server in the search cluster.

blacklistedAutoMappedRoles - <comma separated list of roles> Optionally provide a comma-separated list of Splunk roles that you do not want Splunk to auto-map if received in the IDP Response.

blacklistedUsers - <comma separated list of user names> Optionally provide a comma-separated list of user names that Splunk must reject from the IDP response.

nameIdFormat - <string> Optionally, and if supported by IDP, specify the format of the Subject returned in the SAML Assertion.

ssoBinding - <HTTPPost | HTTPRedirect> Optionally specify the binding to use when making a SP-initiated SAML request. The binding must match the one configured on the IDP.

sloBinding - < <HTTPPost | HTTPRedirect> > Optionally specify the binding to use when making a logout request or sending a logout response to complete the logout workflow. The binding must match the one configured on the IDP.

signatureAlgorithm - <RSA-SHA1 | RSA-SHA256> Optionally specify the signature algorithm to use for a SP-initiated SAML request. 'signedAuthnRequest' must be set to true for this setting to take effect. The algorithm applies to both the http post and redirect binding.

inboundSignatureAlgorithm - <RSA-SHA1;RSA-SHA256,...> Optionally provide a semicolon-separated list of signature algorithms that are accepted in SAML responses. This setting affects both HTTP POST and HTTP Redirect binding.

replicatescertificates - <boolean> Optionally specify the IdP certificate files to replicate across search head cluster setup. Search head clustering must also be enabled. If certificate replication is not enabled, IdP certificate files must be replicated manually across SHC or verification of SAML signed assertions fails.

Configuring SAML in a search head cluster

You can configure SAML on a search head that does or does not use a load balancer. For authentication requests to be signed (recommended), you must use the same signing certificate on all search head members in the cluster.

Every search head in the cluster must have the public key of the IdP. Splunk uses this key to verify the signature of the SAML authentication response. When you use SplunkWeb to configure SAML, the public key from metadata is automatically set to replicate to Search.

1. Generate a public/private key pair.
2. Concatenate the generated key pair into one pem file. This file is used for signing authentication requests going out from Splunk. Concatenate in the following order:

- Public key is self signed:
  - Private key
  - Public key
- Public key is signed by a intermediate/rootCA:
  - Private key
  - Public key
  - Issuers of PublicKeys. Should match the order in that the certificate issuers present.
  - root CA.

3. Replicate the new certificate file to the location relative to $SPLUNK_HOME on each search head. Make sure to give the certificate the same name on all search heads. For example:

```
$SPLUNK_HOME/etc/auth/samlRequestSigningCerts/samlRequestSigningCert.pem
```

4. Edit the Splunk metadata: In the `<X509Certificate>` file, swap the public key in the metadata with the public key from the new certificate. Then remove the `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----` tags.

5. Configure your IdP using the Splunk metadata. See the instructions for your IdP.

6. Collect your IdP metadata and use it to configure Splunk. Previous steps created a SAML-related configuration in $SPLUNK_HOME/etc/system/local/authentication.conf.

**Note:** To enable seamless Single Logout, we recommend that you configure search head members to all have same entityID.

7. Add the path to the ClientCert parameter in authentication configuration:

```
clientCert = $SPLUNK_HOME/etc/auth/samlRequestSigningCerts/samlCert.pem
```

8. If the private key you created in step 1. is encrypted and you set up a password for the private key `sslPassword = <password for private key>` then you must repeat steps ABC for all search head members.

9. Reload authentication on all search heads to implement your changes.

10. To validate your configuration, log in to each search head individually to ensure all search heads are using the same key for signing authentication requests and that the IdP is configured with the right cert for verifying signature of the request.

### Configure Ping Identity with leaf or intermediate SSL certificate chains

To configure Ping Identity with leaf or intermediate certificates:

1. Verify or create the following directories in Splunk. You can use the following command:

```
/home/build/build-home/galaxy/openssl/  ***(or which every directory
/*/splunk cmd openssl version ?d* command returns
/opt/splunk/etc/auth/idpCerts/
```

```
/home/build/build-home/galaxy/openssl/  ***(or which every directory
/*/splunk cmd openssl version ?d* command returns
```

```
/opt/splunk/etc/auth/idpCerts/certs/
```
2. Create a link between the two directories:

\texttt{ln -s /opt/splunk/etc/auth/idpCerts/ /home/build/build-home/galaxy/openssl/}

3. Place your certificate chain in

\texttt{/opt/splunk/etc/auth/idpCerts/} and ensure that they are owned by splunk:splunk:

\texttt{rw-rw-r-. 1 splunk splunk 1635 Nov 1 16:33 aaa_intermediate.pem}
\texttt{rw-rw-r-. 1 splunk splunk 1261 Nov 1 16:33 aaa_root.pem}

4. Once your root, intermediate, and leaf certificate files are in place, create x.509 hash links from the certificates directory to the certificates in the idpCerts directory. You can use the following command:

\texttt{ln -s /opt/splunk/etc/auth/idpCerts/aaa_intermediate.pem `openssl x509 -hash -noout -in /opt/splunk/etc/auth/idpCerts/aaa_intermediate.pem`.0}
\texttt{ln -s /opt/splunk/etc/auth/idpCerts/aaa_root.pem `openssl x509 -hash -noout -in /opt/splunk/etc/auth/idpCerts/aaa_root.pem`.0}

5. In the authentication.conf file, point the idpCertPath attribute to the intermediate filename (make sure you are pointing to the certificate file, not the symlink). You can use the following entry:

\texttt{idpCertPath = /opt/splunk/etc/auth/idpCerts/aaa_intermediate.pem}


7. Select the following fields:

- Sign AuthnRequest = checked
- Sign SAML response = checked

8. Save your changes in Splunk Web.

9. In your Ping Identity configuration, set "Include Certificate in KeyInfo" to "True".

9. Save your changes.

Configure SAML SSO for other IdPs

Any identity provider that is compliant with version 2.0 of the Security Assertion Markup Language (SAML) should be configurable with SAML on the Splunk platform. For information about supported and tested IdPs, see How SAML SSO works.

The following topic provides general instructions for configuring an IdP to work with Splunk Enterprise. There might be exceptions or differences in the process of configuring an IdP to work with Splunk Enterprise, depending upon the IdP you use. For assistance with configuration procedures that are not documented in this chapter, contact Splunk Support.

When you configure the Splunk platform to use your SAML authentication system, you can authorize groups on your IdP to log in by mapping them to Splunk user roles.
SAML does not support encryption.

**Prerequisites**

Verify that your system meets all of the requirements. See Configure single sign-on with SAML.

1. In the **Settings** menu, select **Access Controls > Authentication method**.
2. Select **SAML** as your authentication type.
3. Click **Configure Splunk to use SAML**.
4. On the SAML Groups page, click **SAML Configuration**.
5. Browse and select your metadata file, or copy and paste your metadata directly into the text window. Refer to your IdP's documentation if you are not sure how to get your metadata file.
6. In **General Settings**, provide the following information.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Sign on URL.</td>
<td>This field is populated automatically by your selected metadata file. It is the protected endpoint on your IdP to which Splunk sends authentication requests. Your users use this URL for SSO login.</td>
</tr>
<tr>
<td></td>
<td>To access the login page once SAML is enabled, append the full login URL (/account/login) with loginType=Splunk. Users can also log into their local Splunk account by navigating directly to splunkweb:port/en-US/account/login?loginType=Splunk</td>
</tr>
<tr>
<td>Single Log Out URL.</td>
<td>This field is populated automatically by the metadata file and is the IdP protocol endpoint. If you do not provide this URL, the user will not be logged out.</td>
</tr>
<tr>
<td>IdP's certificate path</td>
<td>This value can be a directory or a file, depending on your IdP requirements. If you provide a file, Splunk uses that file to validate authenticity of SAML response. If you provide a directory, Splunk looks for all the certificates that are present as children of the directory and tries to validate SAML response with each one of them, if Splunk fails to validate authenticity with all of them, response is not considered authentic. When configuring SAML on a search head cluster, make sure you configure the same certificate for each search head.</td>
</tr>
<tr>
<td>Entity ID.</td>
<td>This field is the entity ID as configured in the SP connection entry in your IdP.</td>
</tr>
<tr>
<td>Sign AuthRequest.</td>
<td>Select this option.</td>
</tr>
<tr>
<td>Sign SAML Response.</td>
<td>Select this option.</td>
</tr>
</tbody>
</table>

7. If you use PingIdentity as your IdP, in **Attribute Query**, provide the following information so you can create scheduled searches later. These fields are not required for creating scheduled searches with Okta, Azure AD, or AD FS.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Query URL.</td>
<td>This field is the endpoint on the IdP to which queries over SOAP are sent. The format is as follows: <a href="">urn:oasis:names:tc:SAML:2.0:attrname-format:uri</a></td>
</tr>
<tr>
<td>Sign attribute query request</td>
<td>Verify that this field is selected.</td>
</tr>
<tr>
<td>Sign attribute query response</td>
<td>Verify that this field is selected.</td>
</tr>
</tbody>
</table>

8. In **Advanced settings**, provide the following information.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Alias Role</td>
<td>Use this field to specify a new attribute name on any IdP and then configure an alias on the Splunk side for any of the 3 attributes. If you have configured the AD FS built-in &quot;Role&quot; attribute to be returned and this has the AD group information, specify <a href="http://schemas.microsoft.com/ws/2008/06/identity/claims/role">http://schemas.microsoft.com/ws/2008/06/identity/claims/role</a>. This value tells Splunk</td>
</tr>
</tbody>
</table>

105
the attribute that contains the role information in the SAML response returned.

If you have configured Azure AD, specify

<table>
<thead>
<tr>
<th>Attribute Alias Real Name</th>
<th>Populate this field if you use Azure AD as your IdP. This value tells Splunk Enterprise where to map the real name in the SAML response returned. Enter <a href="http://schemas.microsoft.com/identity/claims/displayname">http://schemas.microsoft.com/identity/claims/displayname</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Alias Mail</td>
<td>Populate this field if you use Azure AD as your IdP. This value maps the alias to the user email addresses in the SAML response returned. Enter <a href="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress">http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress</a>.</td>
</tr>
<tr>
<td>FQDN - Host Name or IP of the load balancer</td>
<td>Set to: <a href="https://sh1.STACKID.splunkcloud.com">https://sh1.STACKID.splunkcloud.com</a>. This setting works for a Splunk deployment with Single Search Head Setup or a Search Head Cluster Setup. You must provide an address if you use load balancing with a search head cluster.</td>
</tr>
<tr>
<td>(Optional) Redirect Port</td>
<td>Provide a redirect port for the load balancer described in the previous field. For Okta set it to &quot;0&quot; (Zero).</td>
</tr>
</tbody>
</table>

9. Click Save.

Next Steps

Map SAML groups to Splunk Enterprise roles

Configure authentication extensions for SAML tokens

When you configure a Splunk platform instance to use a single sign-on scheme that uses the Security Assertion Markup Language (SAML), depending on the identity provider (IdP) that the scheme connects to, you might have to create or configure authentication extensions, or scripts, that interface with the IdP to process user logins and fetch user information.

In general, if the IdP that you use for authentication with SAML does not support Attribute Query Requests (AQRs), then you can configure authentication extensions to ensure that login and user information retrieval works successfully. When you configure authentication extensions, you must provide details about how your IdP performs these actions in the scripts, then configure the authentication system to use the scripts in the authentication.conf configuration file.

- If the SAML IdP that you use supports AQRs, this content does not apply to you. You can configure tokens as you would with any other type of authentication scheme. Proceed to Create authentication tokens for instructions.
- To learn how to configure authentication extensions, see "Configure authentication extensions" later in this topic.
- To learn more about how authentication extensions work, continue reading.

How authentication extensions work with SAML IdPs

When you connect to a SAML IdP that does not support AQRs, there is no way for the auth system to know how to process login and get information about a user natively, as could be done with AQRs.

Splunk has implemented a system where you can configure an authentication extension to perform the login processing and user retrieval functions as a way to interface with an IdP that cannot provide that information natively. In this case, you provide the extension, which is a Python script, configure the Splunk platform to use the extension, supply the
required arguments to the extension, and potentially make changes to the extension so that it works with your IdP.

The authentication extensions support two separate script functions:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getUserInfo</td>
<td>This function lets the Splunk platform retrieve user information from the IdP as needed, and is a drop-in replacement for AQRs. You can configure an authentication extension to use this script function to gather user information after a successful login as if it did support AQRs.</td>
</tr>
<tr>
<td>login</td>
<td>This function lets the Splunk platform perform post-processing on information that the IdP returns after a successful SAML login. You can configure this script function in cases where IdPs return information that needs post-processing. An example of such a case is Microsoft Azure, which returns a link to a list of groups when a user that is a member of more than 150 groups logs in. Without this extension, the SAML response to the login request appears to be empty.</td>
</tr>
</tbody>
</table>

As part of configuring the scripts, you can supply the following arguments and process the following returned objects:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Arguments</th>
<th>Returns</th>
</tr>
</thead>
</table>
| getUserInfo | --username=<username> | --status=success|fail  
 | | | --userInfo=<username>;<realname>;<roles> |
| | --status=success|fail  
 | | | --userInfo=<username>;<realname>;<roles> |
| | Notes:  
 | | • userInfo must specify a semicolon-delimited list.  
 | | • <username> is required.  
 | | • <realname> is optional, but its semicolon is required.  
 | | • <roles> is required. To return multiple roles, use colons to separate the roles.  
| | For example: admin:power  
 | | This example returns just the roles for a user named "testUser":  
 | | • --status=success  
 | | • --userInfo=testUser;Test;admin:power |
| login | --userInfo=<username>;<realname>;<roles> | --status=success|fail  
 | | | --userInfo=<username>;<realname>;<roles> |
| | Note: userInfo contains role information that the IdP returns as a SAML assertion. |

Splunk includes sample authentication extensions that work with various well known IdPs that do not support AQRs, such as Azure and Okta. You can make changes to these extensions so that they work with your specific IdP.

You do not need to implement both extensions in your environment for successful SAML logins. You can configure the extensions that best address your specific authentication use case.

**Technical information on the authentication extensions**

There are two scripts that comprise authentication extensions, both of which exist in the `$SPLUNK_HOME/share/splunk/authScriptSamples` directory:
• commonAuth.py: This script is the base script that retrieves specific arguments from the auth system, as provided by the second, IdP-specific script. It comes with the Splunk Enterprise installation package.
• IdP-specific script: This script takes IdP-specific arguments, such as API key, base URL for accessing the IdP, and so on, and initiates contact with the IdP for the purposes of establishing a login session or retrieving user information. The Splunk Enterprise installation package includes several example scripts with support for the Azure and Okta IdPs. You can modify and use these scripts or write a custom script that works with your specific IdP. In any case, you must always include the commonAuth.py script for authentication to work.

These scripts exist in the $SPLUNK_HOME/etc/auth/scripts directory, and changes to authentication.conf activate the scripts when the Splunk platform authentication system initiates contact with the SAML IdP to log in or retrieve user information.

Several settings in authentication.conf control how this process works:

<table>
<thead>
<tr>
<th>Setting</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>scriptPath</td>
<td>Performs the connection to the IdP during the login or user retrieval process.</td>
</tr>
<tr>
<td>scriptFunctions</td>
<td>Provides the functions that the extension should use when connecting to the IdP to log in and retrieve user information.</td>
</tr>
<tr>
<td>getUserInfoTTL</td>
<td>Determines how long the Splunk platform retains user information that it has retrieved from the IdP.</td>
</tr>
<tr>
<td>scriptTimeout</td>
<td>Determines how long the Splunk platform lets the extension run while logging in or retrieving user information from the IdP.</td>
</tr>
<tr>
<td>scriptSecureArguments</td>
<td>Provides the credentials, tokens, or other items that the extension needs to initiate a connection to the IdP for login or user information retrieval.</td>
</tr>
</tbody>
</table>

For more information on these settings, see authentication.conf in the Admin Manual.

Upon a login attempt on a Splunk platform instance that is configured for SAML, the auth system attempts to log in to the IdP with the authentication extension, using the script functions that you provide. Upon a successful connection to the IdP, it then retrieves user information from the IdP to verify that the user is valid and complete the login process.

**Configure authentication extensions**

Perform this procedure after you configure the authentication system to use SAML as an authentication scheme. You must edit configuration files to make this change, as the ability to enable extensions is not available in Splunk Web.

1. Open a shell prompt.
2. Change to the $SPLUNK_HOME/etc/auth/scripts directory. You might need to create this directory if it does not exist.
3. Copy the commonAuth.py script from the $SPLUNK_HOME/share/splunk/authScriptSamples directory to the $SPLUNK_HOME/etc/auth/scripts directory.
4. Depending on the IdP you use for SAML, do one of the following:
   1. If your IdP is Azure or Okta, copy the sample Azure or Okta script from the $SPLUNK_HOME/share/splunk/authScriptSamples directory to the $SPLUNK_HOME/etc/auth/scripts directory.
   2. If your IdP is neither of these, you can either copy one of the above scripts and modify it or write your own script.
5. Use a text editor to modify the IdP-specific script to include specific information about your IdP. The information you provide here depends specifically on the IdP that you use for SAML.
6. Change to the $SPLUNK_HOME/etc/system/local directory.
7. Using a text editor, create or edit authentication.conf. At a minimum, add the following entries to the file:
Configure advanced settings for SSO

Populate the advanced section to set up load balancing or change the SAML binding. In **Advanced settings**, provide the following information:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name Id Format</td>
<td>Provide the Id format provided by your IdP.</td>
</tr>
<tr>
<td>Fully qualified domain name or IP of the load balancer</td>
<td>Set to: <a href="https://sh1.STACKID.example.com">https://sh1.STACKID.example.com</a>. This setting works for a Splunk deployment with Single Search Head Setup or a Search Head Cluster Setup. You must provide an address if you use load balancing with a search head cluster.</td>
</tr>
<tr>
<td>(Optional) Redirect port - load balancer port</td>
<td>Provide a redirect port for the load-balancer described in the previous field.</td>
</tr>
<tr>
<td>Redirect to URL after logout</td>
<td>URL to which Splunk should direct a user after logout.</td>
</tr>
<tr>
<td>SSO Binding</td>
<td>The protocol binding to use for the login request sent to the IdP.</td>
</tr>
<tr>
<td>SLO Binding</td>
<td>The protocol binding to use for the logout request sent to the IdP.</td>
</tr>
</tbody>
</table>

Map SAML groups to roles

When you configure your Splunk deployment to use your SAML authentication system, you can authorize groups on your SAML server to log in by mapping them to Splunk user roles. You can map multiple groups to a single user role.

**Prerequisites**

**About SAML SSO**

1. In the **Settings** menu, select **Access Controls > Authentication method**.
2. Select **SAML** as your authentication type.
3. Click **Configure Splunk to use SAML**.
4. On the SAML Groups page, click **New Group** or click **Edit** for a group you want to modify.
5. Provide a name for the group.
6. Determine the roles that you want to assign to this group by moving the desired roles from the left column to the right column.
7. Click **Save**.

After you configure SAML SSO and map groups to role, you can distribute the login URL to your users.

**Modify or remove role mappings**

When you configure your Splunk deployment to use your SAML authentication system, you can authorize groups on your SAML server to log in by mapping them to Splunk user roles. You can map multiple groups to a single user role.

This topic describes how to remove roles from existing groups or delete groups entirely. To remove and individual user from a SAML group, consult your IdP documentation.

**Prerequisites**

**About SAML SSO**

1. In the **Settings** menu, select **Access Controls > Authentication method**.
2. Select **SAML** as your authentication type.
3. Click **Configure Splunk to use SAML**.
4. To delete an entire group click **Delete** for the group you wish to remove.
5. On the SAML Groups page, click **Edit** for a group you want to modify.
6. Specify the roles that you want to remove from this group by moving the desired roles from the right column to the left column.
7. Click **Save**.

After you configure SAML SSO and map groups to role, you can distribute the login URL to your users.

**Configure SAML SSO in the configuration files**

This topic explains how to set up SSO for SAML v2 using configuration files:

- Configure **authentication.conf** and **web.conf** in Splunk Enterprise
- Configure your identity provider
- Secure your SAML configuration

**Configure** **authentication.conf**

Configure the following stanza in **authentication.conf**

```plaintext
[authentication]
```
authSettings = saml_settings
authType = SAML

[roleMap_SAML]
admin = Super Admin;
power = Power Admin;
user = <list roles> Admin; Employee;

[saml_settings]
entityId = <entityid>
idpAttributeQueryUrl = <optional path to the Attribute query>
https://your path/idp/attrsvc.ssaml2

idpCertPath = <path to the idp cert in Splunk>
/home/user/splunk/saml-install/etc/auth/ping_idp.crt.>

idpSSOUrl = <path to the sso url>
https://your path/idp/SSO.saml2.

idpSLOUrl = <Logout url> If not specified, this will be treated as a typical sso and the logout button is disabled.
https://your path/idp/SLO.saml2 #

redirectPort=443
attributeQueryTTL = 3600
signAuthnRequest = true
signedAssertion = true

attributeQueryRequestSigned = <Set to true if using optional idpAttributeQuerySSL>
attributeQueryResponseSigned = <Set to true if using optional idpAttributeQuerySSL>

attributeQuerySoapPassword = <your password>
attributeQuerySoapUsername = <your username>

To configure single sign-on with Azure AD or ADFS, add the following additional attributes:

nameIDFormat = (optional) Specify the format of the subject that is returned in the SAML response. AzureAD returns a string to identify the subject and this attribute lets you optionally specify a different format (we recommend email address). This can be useful for auditing and saved searches. To specify email address as the format, use:
urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress

role = Populate this field if you use Azure AD for SSO or ADFS. This value tells Splunk Enterprise the attribute that supplies role information in the SAML response returned. For Azure AD, use:
http://schemas.microsoft.com/ws/2008/06/identity/claims/groups

mail = This value maps the alias to the user email addresses in the SAML response returned. For Azure AD, use:
http://schemas.microsoft.com/identity/claims/displayname
realName = This tells Splunk Enterprise where to map the real name in the SAML response returned. For Azure AD use: http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress.

Use blacklists to improve security

Splunk supports auto-mapped roles by default, so if Splunk roles are returned in an assertion, Splunk uses them. To turn off auto-mapping for roles, add the list of roles to the `blacklistedAutoMappedRoles` setting in `authentication.conf`:

```
blacklistedAutoMappedRoles = <Comma separated list of splunk roles that should be blacklisted from being auto-mapped by Splunk from the IDP Response>
```

To prevent blacklisted users from logging in, add the users to `blacklistedUsers` attribute in `authentication.conf`. If your Splunk instance is configured to use `defaultRolesIfMissing`, and role information is missing in the assertion, Splunk uses the `defaultRolesIfMissing` setting to complete that information.

```
blacklistedUsers = <comma separated list of user names from the IDP response to be blacklisted by the Splunk Platform>
```

Do not configure `defaultRolesIfMissing` with the "Admin" role. The Admin role is temporarily used to send group information in the SAML assertion until the IdP is configured.

**Supported blacklist role and group formats**

User and group names are not case-sensitive manner. Names are converted to lower case and stored. If the IdP expects the username in the NameId in the exact case that it was sent to Splunk, this might result in an error.

Splunk supports groups information in the following formats:

**DN format**

```
<saml:AttributeStatement>
  <saml:Attribute Name="role" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
    <saml:AttributeValue xsi:type="xs:string">
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        CN=Employee,OU=SAML Test,DC=qa,DC=ad2r2,DC=com</saml:AttributeValue>
    </saml:AttributeValue>
    <saml:AttributeValue xsi:type="xs:string">
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        CN=User Admin,OU=SAML Test,DC=qa,DC=adr2,DC=com</saml:AttributeValue>
  </saml:Attribute>
</saml:AttributeStatement>
```

**Strings format**

```
<saml2:AttributeStatement>
  <saml2:Attribute Name="role" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
  </saml2:Attribute>
</saml2:AttributeStatement>
```
Semicolon separated lists

<
<saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
Name="role">
  <saml:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">
    role_splunktest;role_admin
</saml:AttributeValue>
</saml:Attribute>

Splunk Roles

<
<saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
Name="role">
  <saml:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">
    admin
</saml:AttributeValue>
</saml:Attribute>

Configure web.conf and optionally add a failure redirect address

Add the following values to your settings stanza in web.conf

[settings]
appServerPorts = 7065 <make sure this attribute is enabled>
<br this is your custom user redirect for failed logins>

Configure certificate replication for search head clusters

Splunk software uses certificate replication to allow SAML single sign-on in a search head cluster environment. If a search head cluster is set up and certificate replication is turned on, then IdP certificate files are replicated across that search head cluster.

The is enabled by default and if there is not a configured search head cluster, the system ignores the attribute.

In authentication.conf turn on certificate replication:

replicateCertificates = true / false

Configure your identity provider

Now you must configure your IdP to import Splunk software metadata. To import Splunk software metadata on your IdP, make sure that the AuthnRequest signing and AttributeQuery request signing setting is compatible on Splunk software and the IdP:

1. Export the IdP certificate onto a file in your Splunk software instance.

2. Make sure that authentication.conf points to this certificate in the SAML configuration stanzas.

3. Import the Splunk software server certificate (server.pem) into the IdP for signature verification.
Note that you can export Splunk software metadata using the `/saml/spmetadata` endpoint on Splunk Web. You can also access the `SAML-sp-metadata` endpoint on `splunkd`.

**Secure your SAML configuration**

SAML attributequery service supports all of the standard SSL settings for Splunk Enterprise to perform TLS verification between Splunk Instance and SOAP instance providing AttributeQuery service.

In general, TLS encryption only works with an IdP that supports attribute queries. However, the `sslKeysFile` and `sslKeysFilePassword` attributes will work for any IdPs.

For more information about TLS encryption, see Configure SSO with TLS certificates

**SAML SSO best practices**

1. Always enable SSL for Splunk Web.
2. Enable authentication request signing to make sure that all SAML responses, for example AQR, assertions, and logout responses, are signed.
3. Set an Issuer ID in `Authentication.conf`.
4. Use Post binding for SAML responses sent by the IdP to the Splunk platform.
5. For your SAML responses, use a certificate chain instead of self-signed certificates.
6. Use Post and Redirect binding for SAML responses sent to the Splunk platform by the IdP. With redirect binding, the Splunk platform verifies the SAML response against the leaf certificate on disk. The Splunk platform does not perform CRL validation during response verification.
7. Make sure that none of your certificates are expired or revoked.
8. Set blacklisted users to ensure that accounts and users are unable to log in or remain logged in.

   `blacklistedUsers = <Comma-separated list of user names from the response that should be blacklisted by the Splunk platform.>`
9. Set blacklist of untrusted users that are in control of IdP group names. For example, you can limit access by specifying that Splunk roles such as admin and power are added to auto-mapped rules section.

   `blacklistedUsers = <Comma-separated list of user names from the IDP response that should be blacklisted by the Splunk platform.>`
10. The Splunk platform supports auto-mapped roles by default. If Splunk roles are returned in an assertion, the Splunk platform uses them. To turn off auto-mapping for roles, add the list of roles to the `blacklistedAutoMappedRoles` setting in `authentication.conf`.

   `blacklistedAutoMappedRoles = <Comma separated list of Splunk roles from the IDP Response that should be prevented from being auto-mapped by the Splunk platform.>`
11. Do not assign the Admin role to `defaultRolesIfMissing` setting. The Admin role is temporarily used to send group information in the SAML assertion until the IdP is configured.

**Troubleshoot SAML SSO**

Here are some common issues and how to resolve them.
Error message: SAML fails to verify assertions

You see the following error message:

Failed to verify the assertion - The 'Audience' field in the saml response from the IdP does not match the configuration

**Mitigation**

1. The SAML errors are recorded in the splunkd.log on the search head. You can see the complete error message by running a search on that search head:

```
index=_internal sourcetype=splunkd SAML error
```

You should see the following:

```
09-18-2017 14:58:06.939 +0000 ERROR Saml - Failed to verify the assertion - The 'Audience' field in the saml response from the IdP does not match the configuration, Error details=Expected=https://<instance_name>.com, found=https://<wrong_instance_name>.com/
```

2. Modify `authentication.conf` with the `entityId` found in the error message in step 1.

```
[saml]
entityId= https://<instance_name>.com/ (found from ERROR message)
```

3. Reload `authentication.conf` from Splunk Web at **Settings > Access Controls > Authentication Method > Reload Authentication configuration**

Error message: Leaf certificate does not match

You receive the following message:

```
No leaf certificate matched one from the assertion
```

This error occurs when the signature certificate on Splunk does not match the certificate that the IdP uses to sign SAML messages.

**Mitigation**

If your signature verification certificate is a self-signed certificate:

Make sure that the certificate specified in the `idpCertPath` attribute in `authentication.conf` is the same as the certificate the IdP uses to sign SAML messages. You can use OpenSSL to determine the details of the certificate that Splunk uses for signature verification.

For example, the following command:

```
openssl x509 -in etc/auth/idpCerts/idpCert.pem -text -noout | grep 'Serial\|Issuer\|Subject:'+
```

Should produce information similar to this:

```
Serial Number: 1478287046063 (0x15830c635af)
Issuer: C=US, ST=CA, L=San Francisco, O=Splunk, OU=Splunk Service, CN=5165ffdf1a0363c8a5c08062337fb4
Subject: C=US, ST=CA, L=San Francisco, O=Splunk, OU=Splunk Service,
```
If the signature verification certificate is part of a certificate chain

Make sure that the signing certificates match and are consistently named. For example, a simple chain would have three files in the following order:

- the root CA, for example: "cert_1.pem"
- the intermediate certificate, for example: "cert_2.pem"
- the leaf certificate or the signing certificate, for example: "cert_3.pem"

In this example, make sure that the "cert_3.pem" (the leaf) is the same certificate that the IdP uses to sign responses.

If you have multiple chains, or chains with more than one intermediate CA

In most cases, the certificate chain consist of a single root certificate, a single intermediate certificate, and a single signing certificate. However, you may have multiple chains configured, or more than one intermediate CA.

If you have multiple chains configured, structure your certificate chain as follows:

```
$SPLUNK_HOME/etc/auth/idpCerts
idpCertChain_1 idpCertChain_2
$SPLUNK_HOME/etc/auth/idpCerts/idpCertChain_1
cert_1.pem cert_2.pem cert_3.pem
$SPLUNK_HOME/etc/auth/idpCerts/idpCertChain_2
cert_1.pem cert_2.pem cert_3.pem
```

If you have more than one intermediate CA

If you have more than one intermediate CA, structure your certificate chain as follows:

```
$SPLUNK_HOME/etc/auth/idpCerts
idpCertChain_1
$SPLUNK_HOME/etc/auth/idpCerts/idpCertChain_1
cert_1.pem cert_2.pem cert_3.pem cert_4.pem cert_5.pem
```

Error message: Attribute query request error

**Issue: You experience the following message**

ERROR AuthenticationManagerSAML - Requesting user info from ID returned an error. Error in Attribute query request, AttributeQueryTransaction err=Cannot resolve hostname, AttributeQueryTransaction descr=Error resolving: Name or service not known, AttributeQueryTransaction statusCode=502

**Mitigation**

- Make sure that the `cipherSuite` is specified correctly in the SAML stanza. For example:
  - `cipherSuite = TLSv1+MEDIUM:@STRENGTH`
  - `cipherSuite = ALL:!aNULL:!eNULL:!LOW:!EXP:RC4+RSA:+HIGH:+MEDIUM`

- Make sure all SOAP password requirements are met.

- Make sure your SSL settings for SAML are configured correctly in `authentication.conf`. 

116
Issue: You experience the following message:

```
ERROR AuthenticationManagerSAML - Attribute query request failed. Status
code=urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal, Status msg=No attributes found for
requested subject
```

**Mitigation**

- Make sure that the `role`, `mail`, and `realName` attributes are mapped to be returned back as part of `AuthnRequest`
and the Attribute Query Request.

Error message: SAML user missing roles

You experience the following message:

```
ERROR UserManagerPro - user="samluser1" had no roles
```

**Mitigation**

Make sure that `rolemap_SAML` contains the correct role mapping with ";;" at the end of each role name.

User cannot login

User cannot log in after successful assertion validation. No valid Splunk role is found in the local mapping or in the
assertion.

**Mitigation**

- Make sure that `rolemap_SAML` stanza contains proper mapping between roles returned from IdP and the
appropriate Splunk role.

- Make sure there are no spaces between, before, or after each role defined in `authentication.conf`. For example:

```
user = User;Employee
```

User cannot access SAML login page

Authentication is configured as SAML and the settings appear to be correct, but the login screen shows the page for
Splunk authentication instead.

**Mitigation**

- Make sure that in `web.conf`, `appServerPorts` is set to a valid port and not '0'.

- Make sure `web.conf` does not contain a value for `trustedIP`.

Error message: Failed to validate SAML logout response

When you log out of Splunk Enterprise or Splunk Cloud, you see the following error message:

```
Failed to validate SAML logout response received from IdP
```
Mitigation

This might be caused by case-sensitive IdPs that expect Splunk software to preserve uppercase letters in usernames. You can change the username to lowercase in the IdP or configure the IdP to accept the lowercase version of a username.

Cannot authenticate users for CLI commands

Unable to authenticate SSO users for CLI commands

Mitigation

You can add the SAML users as native Splunk users.

API and CLI commands cannot be performed by users that are defined only in SAML. This is because the user password is never sent in the SAML assertion.
Authentication using Proxy SSO

About ProxySSO

ProxySSO is an authentication method that lets you configure Single-Sign On (SSO) for Splunk instances through a reverse proxy server. A user logged in using SSO can seamlessly access Splunk Web.

With ProxySSO Single-Sign On, user identity and group information can be passed in HTTP headers to Splunk Enterprise. Splunk Enterprise uses this information to authenticate users and authorize them by mapping groups to appropriate Splunk Enterprise roles.

ProxySSO authentication:

- Combines authentication and authorization into one step for the user, streamlining the login process.
- Reduces configuration steps. No need to configure complex LDAP strategies within Splunk Enterprise.
- Reduces the back and forth messages between Splunk Enterprise and authentication services, making authentication more efficient.
- The external authentication service is not restricted to LDAP as long as the proxy server can pass the required information.

ProxySSO cannot be configured through Splunk Web. Instead you must use the REST API or modify configuration files as described in Configure ProxySSO.

Splunk Cloud does not support ProxySSO.

Prerequisites

To set up ProxySSO, you should already have the following configured:

- A Proxy Server configured to send required HTTP headers.
- A working Splunk Enterprise configuration.

For more information about how to configure these items and set up ProxySSO, see Configure ProxySSO.

How it works

1. The proxy server authenticates against the configured authentication service and creates an HTTP request.
2. Splunk Enterprise receives HTTP headers from the trusted reverse proxy server.
3. Splunk Enterprise checks trustedIP (which is configured in web.conf) for a receiving request from the proxy.

After a successful login, a session cookie is created and the user can seamlessly access Splunk Web.

Configure ProxySSO

Before you configure Splunk Enterprise, configure your proxy server so that it acts as a proxy for Splunk Web, prompts user for credentials, and passes user identity and groups to Splunk Web through HTTP headers:
AuthType Basic
AuthBasicProvider ldap

ProxyPass / http://mysplunkhost:8000/
ProxyPassReverse / http://mysplunkhost:8000/

AuthLDAPURL "ldap://<ldap-server>:<ldap-port>/OU=IT Department,DC=com?sn,sAMAccountName?"

RequestHeader set Remote_User %{AUTHENTICATE_sn}e
RequestHeader set Remote_Groups %{AUTHENTICATE_sAMAccountName}e

Configure Splunk Enterprise

1. Configure web.conf

[settings]
SSOMode = strict
tradedIP = 10.1.1.2
remoteUser = Remote_User
remoteGroups = Remote_Groups
remoteGroupsQuoted = true
allowSsoWithoutChangingServerConf = 1

2. Restart Splunk.

3. In authentication.conf configure the [authentication] stanza:

[authentication]
authType = ProxySSO
authSettings = my_proxy

4. Map groups to Splunk roles in the roleMap_proxySSO stanza.

[roleMap_proxySSO]
admin = IT operational admin
splunk-system-role = IT sub-admin

defaultRoleIfMissing is assigned:

[my_proxy]
defaultRoleIfMissing = user

5. Configure the [my_proxy] stanza for additional settings. If a group mapping is not found, the role configured in defaultRoleIfMissing is assigned:

Troubleshoot Proxy SSO

You can view the HTTP request headers that proxy server sends to Splunk Web on the below endpoint after you set enableWebDebug=true in web.conf under settings stanza:


This endpoint will help to verify some of the common configuration or setup errors:
• Incoming request IP matches the configured value of trustedIP
• Ensure header attribute names set on proxy server are same as those configured on Splunk
• Make sure group entries are sent and parsed correctly. Especially, when remoteGroupsQuoted = true is set. You can see how groups are parsed by adding category.UiAuth=DEBUG in etc/log.cfg under splunkd stanza.

Once this is verified, check the following configuration:

• Groups parsed have mapping in roleMap_proxySSO
• In some cases, user cannot login because either the user or their roles are blacklisted. Check blacklisted objects under stanza named after value of authSettings

These kind of login events are logged in var/log/splunkd.log along with reason for failure.
Authentication using single sign-on with reverse proxy

About Single Sign-On using reverse proxy

Splunk Single Sign-on (SSO) lets you use a reverse proxy to handle Splunk authentication, meaning that once the user has logged into their proxy, they can seamlessly access Splunk Web (and presumably any other applications configured to your proxy).

The reverse proxy implementation of Splunk Enterprise SSO supports logging into Splunk Enterprise only through Splunk Web. Since the implementation relies on cookies to save authentication information, SSO cannot be used for CLI authentication to Splunk Enterprise. Invoking https://localhost:8089 (or the assigned management port) still requires independent authentication.

For more information about how to configure these items and set up SSO, see Configure Single Sign-On

How it works

Splunk Enterprise administrators and users invoke Splunk Web via a proxy URL that is deployed with Splunk Web. The proxy authenticates the incoming request against your authentication system. Upon successful authentication the proxy sets a request header with the authenticated identity’s attribute and sends this information to Splunk Enterprise.

Splunk Enterprise accepts the incoming HTTP request from the proxy, and if Splunk Enterprise recognizes the user contained in the header, the user bypasses the login page and is automatically authorized.

For successful single sign-on, all requests from the proxy to Splunk Web must include this authenticated header. If the header is not included in a request, then the user is returned to the login page or an error page, depending on your configuration. Splunk software uses this authenticated header for the duration of the browser session.

How Splunk software processes the proxy request

When the proxy server makes a request to Splunk Web, Splunk Web looks to the trustedIP value in web.conf to verify that the proxy’s IP is on the trusted IP list.

If the IP is not trusted, the request is rejected and the sign-on attempt fails. If the IP address is trusted, then Splunk Web queries for the identity in the request header and sends splunkd an authorization request containing that header information.

Upon receiving the authorization request from Splunk Web, splunkd verifies whether the incoming IP address of the client (usually Splunk Web) matches the value of the trustedIP property of the server.conf file.

If the IP addresses are not in the trustedIP list the request is rejected and the sign-on attempt fails. The user is either returned to a login page or shown an error page, depending upon your SSMode configuration in web.conf. For more on this attribute and other configuration information, see Configure Splunk Single Sign-On.

If the IP is trusted, then splunkd uses the information contained in the request header and conducts the authorization process.
How Splunk software authorizes the user

Splunk software first checks to see if the given identity and role matches any of your Splunk users. If no match is found, Splunk software looks to see if there are any LDAP matches. (For information about how Splunk software authenticates users, see Set up user authentication with LDAP in this manual.)

If no match is found and the user contained in the header cannot be authorized, then the browser redirects to an error page.

If a match is found, Splunk software authorizes the user and checks to see if an existing session is present. If a session already exists, Splunk software uses that session identifier and creates the necessary cookies to allow the user access to Splunk Web. If a session does not exist, then Splunk software creates a new session as well as the necessary cookies for Splunk Web authorization.

After the cookies are created, Splunk Web resumes its normal flow. Any subsequent access to Splunk Enterprise through the proxy URL does not require re-authorization as long as the request header contains the trusted identity and until the user closes the browser session.

Configure Single Sign-On with reverse proxy

Before you configure reverse proxy-based SSO with Splunk Enterprise, make sure you have the following:

- A Proxy Server (Splunk Enterprise supports IIS or Apache) configured as a reverse proxy to authenticate to external systems.
- An LDAP Server or other external authentication system provisioned with appropriate groups and users for your proxy to authenticate against.
- A working Splunk Enterprise configuration that is either configured to use the same external authentication system as your proxy (usually LDAP) or that has native Splunk Enterprise users that match the user and group IDs contained in your external authentication system.

Configuring SSO with reverse proxy requires the following steps:

1. Edit the properties on your proxy server to authenticate against your external authentication system.
2. Edit the Splunk Enterprise server.conf file.
3. Edit the Splunk Enterprise web.conf file.

Note: For optimal security, any HTTP header-based solutions should be implemented over a TLS/SSL enabled deployment.

Configure server.conf

Edit the trustedIP in the general settings stanza to add the IP address that will make secure authentication requests to splunkd. This is typically Splunk Web and therefore the localhost. You can only enter one IP address per splunkd instance.

trustedIP=127.0.0.1

If no IP addresses are provided in the trustedIP list, Splunk SSO is disabled by default.
Configure web.conf

To enable SSO, configure the following in the [settings] stanza in web.conf ($SPLUNK_HOME/etc/system/local):

- SSOMode = strict
- trustedIP = 127.0.0.1,10.3.1.61,10.1.8.81
- remoteUser = Remote-User
- tools.proxy.on = False

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Default</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSOMode</td>
<td>no</td>
<td>The SSOMode attribute determines whether the Splunk Web SSO operates in strict or permissive mode. Strict mode restricts authentication to identities that match the IP addresses listed in trustedIP property. If the IP attempting to connect does not match any IP address, an error page appears to the user. Strict mode is recommended for SSO. Permissive mode also restricts authentication to requests from IPs found in the trustedIP list. In permissive mode, if the IP attempting to connect does not match any IP address, a login page is displayed to allow the user to re-authenticate.</td>
</tr>
<tr>
<td>trustedIP</td>
<td>n/a</td>
<td>Set this to the IP address of the authenticating proxy or proxies. Specify a single address or a comma-separated list of addresses; IP ranges and netmask notation are not supported.</td>
</tr>
<tr>
<td>remoteUser</td>
<td>REMOTE_USER</td>
<td>The remoteUser attribute determines the authenticated identity's attribute that is passed by the proxy server via the HTTP request header. This value defaults to REMOTE_USER but any LDAP attribute can be passed in this request header as long as the proxy sets this attribute properly after authentication. When you configure your remoteUser attribute, you must also configure the RequestHeader property in your proxy configuration to pass the identity's attribute to Splunk software. This process is described in &quot;About Splunk Single Sign-On&quot;. The default Splunk header used is REMOTE_USER, but if your proxy uses a different header, you can change the name of the header here.</td>
</tr>
<tr>
<td>tools.proxy.on</td>
<td>false</td>
<td>For apache 1.x proxy this value should be set to True. For later versions this value should be set to False.</td>
</tr>
</tbody>
</table>

If you host Splunk Web behind a proxy that does not place Splunk Web at the proxy's root, you may also need to configure the root_endpoint setting in $SPLUNK_HOME/etc/system/local/web.conf.

For example if your proxy hosts Splunk Web at "yourhost.com:9000/splunk", root_endpoint should be set to /splunk.

For example:

```
root_endpoint=/lzone
```

In the above example, Splunk Web is accessed via http://splunk.example.com:8000/lzone instead of http://splunk.example.com:8000/.

You would next make it visible to the proxy by mapping it in httpd.conf:

```
ProxyPass /lzone http://splunkweb.splunk.com:8000/lzone
ProxyPassReverse /lzone http://splunkweb.splunk.com:8000/lzone
```
Session management

Since there is no simple log out for a session and Splunk Enterprise will preserve a session as long as the correct header information is contained in the proxy header, you should set your proxy’s session timeout value with this in mind.

If you need to end a session before the timeout has occurred, you can use the REST end point along with the session identifier to destroy the session:

curl -s -uadmin:changeme   -k -X DELETE
https://localhost:8089/services/authentication/httpauth-tokens/990cb3e61414376554a39e390471fff0

Troubleshoot reverse-proxy SSO

Splunk Web provides an interface that allows you to analyze the environment and the run-time data to help you debug your deployment. This page can be accessed via the proxy or the direct URL. The request headers will not be available if you do not access this page through the proxy server.

+Splunk recommends that this setting is disabled after troubleshooting is complete.

This URL is located at:

http://YourSplunkServer:8000/debug/sso

Important: This debug page is not available by default. In order to make the page available, two steps must be completed. First, the role that is accessing this end point must have the web_debug capability, which the admin role has by default. Second, in web.conf, the setting enableWebDebug=true must be configured. You should immediately disable this setting after you have finished troubleshooting.

Consider the following when using the troubleshooting page to analyze your deployment:

• Compare the IP provided as the Splunk trusted IP with that of the Host IP. The values must be the same (they should be the IP of your proxy). If they are not the same in the troubleshooting page, you must edit the trustedIP value in server.conf.

• Check the value for Incoming request IP received by splunkweb to make sure that it displays your client's IP address. If the IP does not match that of your client, you must:
  ♦ Edit web.conf to correct this.
  ♦ Make sure that tools.proxy.on is set to true.

• Make sure that your proxy is providing a header. Check the Authorization field under Other HTTP Headers. If there is no value present, check the http.conf file in your proxy to make sure that the remote header attribute value is properly set. Splunk software is configured to accept the remote header value of REMOTE_USER, which is the default for most proxies. If your proxy's remote header is different, and you wish to keep that value, you can edit the remote header value in web.conf to change the header that Splunk software will accept. See Configure SSO for more information.

• Make sure that Splunk Web is creating a cookie to send to splunkd. Check the Cookie field under Other HTTP headers to make sure that a cookie is set. If a cookie is not set, then check your web.conf file to make sure your file is properly configured. Configure SSO for more information.
Scripted authentication

Set up user authentication with external systems

Your options for user authentication are:

- Splunk authentication
- LDAP
- Single sign-on
- A scripted authentication API for use with an external authentication system, such as PAM or RADIUS, described in this section.

Important: Splunk authentication takes precedence over any external systems.

This is the order in which Splunk software authenticates a user for LDAP:

1. Splunk authentication or SSO.
2. LDAP or scripted authentication (if enabled). For more information about LDAP, see "Set up user authentication with LDAP".

How scripted authentication works

In scripted authentication, a user-generated Python script serves as the middleman between the Splunk server and an external authentication system such as PAM or RADIUS.

The API consists of a few functions that handle communications between Splunk software and the authentication system. You need to create a script with handlers that implement those functions.

To integrate your authentication system with Splunk Enterprise, make sure the authentication system is running and then do the following:

1. Create a Python authentication script. See "Create the authentication script" for the procedure.
2. Enable your script by editing authentication.conf to specify scripted authentication and associated settings. See "Edit authentication.conf" for the procedure.

Examples

Splunk provides several example authentication scripts and associated configuration files, including one set for RADIUS and another for PAM. There is also a simple script called dumbScripted.py, which focuses on the interaction between the script and Splunk deployments.

You can use an example script and configuration file as the starting point for creating your own script. You must modify them for your environment.
You can find these examples in $SPLUNK_HOME/share/splunk/authScriptSamples/. That directory also contains a README file with information on the examples, as well as additional information on setting up the connection between Splunk Enterprise and external systems.

Important: These scripts are provided as examples that you can modify or extend as needed. They are not supported and there is no guarantee that they will fully meet your authentication and security needs.

Create the authentication script

To integrate your authentication system with your Splunk deployment, make sure the authentication system is running and then do the following:

1. Create a Python authentication script. See "Create a Python script" in this topic for the procedure.

2. Test the new script. See "Test the script" in this topic for the procedure.

3. Enable your script by editing authentication.conf to specify scripted authentication and associated settings. See "Edit authentication.conf" for the procedure.

Create a Python script

You must create a Python script that implements these authentication functions:

- userLogin
- getUserInfo
- getUsers

The Splunk server will call these functions as necessary, either to authenticate user login or to obtain information on a user's roles.

The script can optionally also include a handler for this function:

- getSearchFilter

This table summarizes the authentication functions, their arguments, and their return values:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Argument string</th>
<th>Return value string</th>
</tr>
</thead>
<tbody>
<tr>
<td>userLogin</td>
<td>Login with user credentials.</td>
<td>--username=&lt;username&gt;</td>
<td>fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--password=&lt;password&gt;</td>
<td>(safely passed over stdout)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(values passed one per line over stdin)</td>
<td></td>
</tr>
<tr>
<td>getUserInfo</td>
<td>Return a user's information, including name and role(s).</td>
<td>--username=&lt;username&gt;</td>
<td>--status=succes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--userInfo=&lt;userId&gt;;&lt;username&gt;;&lt;realname&gt;;&lt;roles&gt;</td>
<td></td>
</tr>
</tbody>
</table>

Note the following:

- **userInfo** must specify a semicolon-delimited list.
### getUsers

**Description**
Return information for all Splunk users.

**Argument string**
none

**Return value string**

- `<userId>` is deprecated; you should return just the associated semicolon.
- `<username>` is required.
- `<realname>` is optional, but its semicolon is required.
- `<roles>` is required. To return multiple roles, use colons to separate the roles.
  
  For example: `admin:power`
  
  This example returns just the roles for a user named "docsplunk":
  
  ```
  --status=succes
  --userInfo=;docsplunk;;admin:power
  ```

**Note the following:**

- See `getUserInfo` for information on the syntax to use to return each user's information.
- Separate each user's information with a space.
- `<roles>` is required. To return multiple roles, use colons to separate the roles.
  
  For example: `admin:power`

---

### getSearchFilter

**Description**
Optional. Returns the filters applied specifically to this user, along with those applied to the user's roles. The filters are OR'd together.

**Argument string**

- `--username=<username>`

**Return value string**

- `--status=succes|fail`
- `--userInfo=<userId>;<username>;<realname>;<roles>`
- `--userInfo=<userId>;<username>;<realname>;<roles>`
- `--userInfo=<userId>;<username>;<realname>;<roles>`
  
  ...  

**Note:** User-based search filters are optional and not recommended. A better approach is to assign search filters to roles and then assign users to the appropriate roles.

For more information, see "Use the getSearchFilter function to filter at search time"

---

See the example scripts for detailed information on how to implement these functions.

### Test the script

Since the communication between your Splunk deployment and the script occurs via `stdin` and `stdout`, you can test the script interactively in your command shell. Be sure to send one argument per line and end each function call with an EOF (Ctrl-D).

Test each function individually, using this pattern:

```
> python [script] [function name]
[pass arguments here, one per line]
[send eof, with Ctrl-D]
[output appears here, check that it's correct]
>
```

The following example shows a debugging session that does some simple testing of a fictional script called "example.py", with two users "alice" and "bob". "alice" is a member of the "admin" and "super" roles, and "bob" is a member of the "user"
> python example.py userLogin
--username=alice
--password=correctpassword
<send an EOF>
--status=success
> python example.py userLogin
--username=bob
--password=wrongpassword
<send an EOF>
--status=fail
> python example.py getUsers
<no arguments for this function, send an EOF>
--status=success --userInfo=bob;bob;bob;user --userInfo=alice;alice;alice;admin:super
> python example.py getUserInfo
--username=bob
<send an EOF>
--status=success --userInfo=bob;bob;bob;user
> python example.py getUserInfo
--username=userdoesnotexist
<send an EOF>
--status=fail
>
**Important:** This is just an example of how to go about testing a script. It does not attempt to perform exhaustive debugging of any real script.

---

**Edit authentication.conf**

To integrate your authentication system with your Splunk deployment, make sure the authentication system is running and then do the following:

1. Create and test a Python authentication script. See "Create the authentication script" for the procedure.

2. Edit authentication.conf to enable your authentication script. See "Enable your script" in this topic.

3. Edit authentication.conf to set your cache duration. See "Set cache durations" in this topic.

**Enable your script**

Once you create a Python script to implement authentication, you update the `authentication.conf` in $SPLUNK_HOME/etc/system/local/ to enable your script. You can also copy and edit a sample `authentication.conf` from $SPLUNK_HOME/share/splunk/authScriptSamples/.

Specify **Scripted** as your authentication type under the [authentication] stanza heading:

```
[authentication]
authType = Scripted
authSettings = script
```

Set script variables under the [script] stanza heading. For example:

```
[script]
```
Set cache durations

To significantly speed authentication performance when using scripted authentication, enable Splunk authentication caching. You do so by adding the optional [cacheTiming] stanza. Each script function (except getSearchFilter) has a settable cacheTiming attribute, which turns on caching for that function and specifies its cache duration. For example, to specify the cache timing for the getUserInfo function, use the getUserInfoTTL attribute. Caching for a function occurs only if its associated attribute is specified.

The cacheTiming settings specify the frequency at which Splunk software calls your script to communicate with the external authentication system. You can specify time in seconds (s), minutes (m), hours (h), days (d), etc. Typically, you'll limit the cache frequency to seconds or minutes. If a unit is not specified, the value defaults to seconds. So, a value of "5" is equivalent to "5s".

This example shows typical values for the caches:

```
[cacheTiming]
userLoginTTL    = 10s
getUserInfoTTL  = 1m
getUsersTTL     = 2m
```

You'll want to set userLoginTTL to a low value, since this determines how long user login/password validity is cached.

To refresh all caches immediately, use the CLI command reload auth:

```
./splunk reload auth
```

**Note:** This command does not boot current users off the system.

You can also refresh caches in Splunk Web:

1. In the System menu, under Users and authentication select Access controls.
2. Click Authentication method.
3. Click Reload authentication configuration to refresh the caches.

Each specified function, except getUsers, has a separate cache for each user. So, if you have 10 users logged on and you've specified the getUserInfoTTL attribute, the getUserInfo function will have 10 user-based caches. The getUsers function encompasses all users, so it has a single, global cache.

Use PAM authentication

You can configure Splunk Enterprise to use PAM authentication by following the steps in the example directory's README, which is located at $SPLUNK_HOME/share/splunk/authScriptSamples/.

If you are still unable to authenticate, then edit /etc/pam.d/pamauth and add this line:

```
auth sufficient pam_unix.so
```
Use the getSearchFilter function to filter at search time

This function is optional and can be used to implement a user-based filter at search time. When `getSearchFilter` is enabled, Splunk software calls it every time a search is run. A user-based search filter supplements any filters specified for that user’s role(s). The returned filter(s) will be applied to each search, along with any configured at the role level. Caching of the filter does not occur with this function.

**Note:** User-based search filters are optional and not recommended. A better approach is to assign search filters to roles and then assign users to the appropriate roles.

To enable `getSearchFilter` function, set the `scriptSearchFilters` parameter in `authentication.conf`:

```plaintext
[script]
scriptPath = $SPLUNK_HOME/bin/python $SPLUNK_HOME/bin/<scriptname.py>
scriptSearchFilters = 1
```

**Note:** In previous releases, `getSearchFilter` could also be used to implement search filters for users who had been authenticated by Splunk software. Starting with 4.2, `getSearchFilter` is called only for users who have been authenticated by scripted authorization.

In addition, if a call to `getSearchFilter` fails, Splunk Enterprise will cancel the user’s search and return an error message, to ensure that users cannot view results from unauthorized searches.
Securing Splunk Enterprise communications with SSL

About securing Splunk Enterprise with SSL

This section describes the types of Splunk configurations that you might want to secure with SSL.

About the default certificates

Splunk software ships with, and is configured to use, a set of default certificates. These certificates discourage casual snoopers but could still leave you vulnerable, because the root certificate is the same in every Splunk download and anyone with the same root certificate can authenticate.

The default certificates are generated and configured at startup and can be found in $SPLUNK_HOME/etc/auth/. They are set to expire three years after they are generated and new certificates must be created and configured at that time.

- For information about the default certificate for Splunk Web, see “Turn on encryption (https) with Splunk Web.” or “Turn on encryption (https) using web.conf.”
- For information about SSL for forwarding with the default certificate, see “Configure Splunk forwarding to use the default certificate.”

Ways you can secure Splunk Enterprise

You can apply encryption and/or authentication using your own certificates for:

- Communications between the browser and Splunk Web
- Communication from Splunk forwarders to indexers
- Other types of communication, such as communications between Splunk instances over the management port

The table below describes the most common scenarios and the default SSL settings:

<table>
<thead>
<tr>
<th>Type of exchange</th>
<th>Node A function</th>
<th>Node B function</th>
<th>Encryption</th>
<th>Certificate Authentication</th>
<th>Common Name checking</th>
<th>Type of data exchanged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Browser to Splunk Web</td>
<td>Browser</td>
<td>Splunk Web</td>
<td>NOT enabled by default</td>
<td>dictated by client (browser)</td>
<td>dictated by client (browser)</td>
<td>configuration and search data</td>
</tr>
<tr>
<td>Splunk Web to search head</td>
<td>Splunk Web</td>
<td>splunkd as a search head</td>
<td>enabled by default</td>
<td>NOT enabled by default</td>
<td>NOT enabled by default</td>
<td>configuration and search data</td>
</tr>
<tr>
<td>Forwarding</td>
<td>splunkd as a forwarder</td>
<td>splunkd as an indexer</td>
<td>NOT enabled by default</td>
<td>NOT enabled by default</td>
<td>NOT enabled by default</td>
<td>data to be indexed</td>
</tr>
<tr>
<td>Deployment server to deployment clients</td>
<td>splunkd as a deployment client</td>
<td>splunkd as deployment server</td>
<td>enabled by default</td>
<td>NOT enabled by default</td>
<td>NOT enabled by default</td>
<td>configuration data</td>
</tr>
<tr>
<td>Distributed search</td>
<td>splunkd as search peer</td>
<td>splunkd as a search head</td>
<td>Enabled by default</td>
<td>NOT enabled by default</td>
<td>NOT enabled by default</td>
<td>search data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Node A function</th>
<th>Node B function</th>
<th>Encryption</th>
<th>Certificate Authentication</th>
<th>Common Name checking</th>
<th>Type of data exchanged</th>
</tr>
</thead>
</table>

Not enabled by default |                |                |            |                           |                      | cluster data           |
Communications between the browser and Splunk Web

Browser to Splunk Web data most commonly consists of search requests and returned data.

Data encryption (HTTPS) can be easily turned on using Splunk Web, or by editing the configuration files. Keep in mind that encryption with the default certificate protects against casual listening but is not fully secure.

For better security, replace the default certificates with certificates signed by a trusted CA. We strongly recommend using CA certs rather than signing your own in this case. Unless you have the ability to add your CA to the certificate stores in every browser that will access Splunk Web, a self-signed certificate is considered untrusted by users' browsers. For more information, see "About securing Splunk Web."

Splunk forwarders to indexers

Data sent from forwarders to indexers is the data that your indexers use for searches and reports. Depending upon your organization and the nature and format of the data being transmitted and Splunk configuration, this data may or may not be readable or sensitive.

Securing sensitive raw data helps to avoid snooping and man-in-the-middle attacks.

You can turn on SSL encryption using the default certificate to provide encryption and compression. However, communication using the default certificate does not provide secure authentication, as the certificate password is supplied with every installation of Splunk software. The default certificates are set to expire three years after initial startup, and forwarder to indexer communications will fail at this point.

For better security, require certificate authentication using a self- or CA-signed certificate. A certificate signed by a known and mutually trusted Certificate Authority is considered more secure by outside parties than a certificate you sign yourself. For more information about using certificates with Splunk forwarders and indexers, see "About securing data from forwarders."

Other SSL communications

Other Splunk communications happen between different instances of Splunk software over the management port, usually but not always in a distributed environment. An example of this is configuration data sent by a deployment server to
clients. This type of SSL encryption is enabled by default. For most configurations this is adequate and is the recommended security method. However, if you do need to secure your communications with SSL authentication, we've provided some guidelines to help you in "About securing Splunk to Splunk communication" in this manual.

To learn about more ways to use TLS certificates, see the following topics:

- Secure LDAP with TLS certificates
- Secure SSO with TLS certificates

Getting your certificates

If you are experienced with SSL certificates, you can create them as you normally would and go straight to configuring your Splunk instances to use them.

If you need help getting your certificates together, we've provided very simple examples using OpenSSL commands. (OpenSSL ships with Splunk software)

- How to self-sign certificates
- How to get third-party certificates
- How to self-sign certificates for Splunk Web
- How to get third-party certificates for Splunk Web

What to do when you have your certificates

The following topics provide more information about configuring Splunk software to use your certificates once you have them:

- Secure Splunk Web with your own certificate
- Configure Splunk forwarding to use your own certificates
- About securing inter-splunk communication

About using SSL tools on Windows and Linux

This manual describes how to configure Splunk deployments to use default, self-signed, or Certificate Authority signed certificates. For those who may not have certificates, we also provide simple examples for generating the certificates and keys using the command line and a version of OpenSSL that is packaged with Splunk software.

Using the OpenSSL command-line examples

This manual provides a few basic examples for creating certificates using the Splunk version of OpenSSL in the command line. In order to perform these tasks you must have root administrator permissions. If you are working on a remote or virtual machine, you may have to take an extra step to ensure that you are able to perform all tasks:

- When working on a Windows platform, you may need to open the command line as the administrator: In the Start Menu, right click the .exe application and select run as administrator.
- When working on a *nix platform, you might need to use sudo to log in as the root administrator.
For more information about the differences between Windows and *nix, see the Administration Guide.

**About SSL tools**

Splunk software ships with a recent version of OpenSSL at $SPLUNK_HOME/splunk/lib. For 6.0, Splunk supports OpenSSL with FIPS 140-2 enabled.

A variety of other SSL tools are available for purchase and download that you can use to create and set up certificates. If you do choose to use OpenSSL for certificate configuration, we strongly recommend that you use the version that ships with Splunk to avoid compatibility issues. To make sure that you are using the version provided with Splunk software, set your environment to the version in $SPLUNK_HOME/splunk/lib or $SPLUNK_HOME\splunk\bin for Windows:

The following is an example of the library path for *nix:

```shell
export LD_LIBRARY_PATH=$SPLUNK_HOME/splunk/lib
```

The following is an example of the path for Windows (using the command prompt):

```shell
set PATH = %PATH%;%SPLUNK_HOME%;\bin
```

**Configure allowed and restricted SSL versions**

Splunk Enterprise version 6.2 and later provides the `sslVersions` keyword to restrict older versions of protocols. SSLv3 is shipped out of box to support easy upgrades but should be disabled as soon as upgrades are complete. By default, Splunk Enterprise allows communications on SSLv3 and all subsequent versions.

When Splunk Enterprise is secured with FIPS, SSLv2 and SSLv3 are always disabled regardless of any additional configuration.

**CAUTION:** To avoid the v3 “POODLE” vulnerability, remove SSLv3 as upgrades are applied to your environment.

**Configure `web.conf`**

1. In `web.conf`, update the `sslVersions` attribute to list or limit the versions (separated by commas) you want to permit. By defaults this attribute is set to `*,-sslv2`, which is any version newer than SSLv2 (not recommended). For 6.2 the allowed SSL versions are:

   - SSLv2 (not recommended)
   - SSLv3 (not recommended)
   - TLS1.0 (not recommended)
   - TLS1.1
   - TLS1.2

For example:

```plaintext
sslVersions = tls1.1, tls1.2
```

**Syntax options**

To select all supported versions use `*`:

```plaintext
sslVersions = *
```

To include all versions tls1.0 or newer use "tls":

```plaintext
sslVersions = tls
```
To restrict a particular version prefix it with "-" :

sslVersions = *, -ssl3

Note: When Splunk Enterprise is configured in FIPS mode, SSLv2 and SSLv3 are always disabled regardless of this configuration.

2. In inputs.conf, update the sslVersions attribute to list or limit the versions (separated by commas) you want Splunk Enterprise to support.

sslVersions = ssl2, tls1.1, tls1.2

You can use "*" to select all supported versions:

sslVersions = *

Simply use "tls" to include all versions tls1.1 or newer:

sslVersions = tls

The prefix a version with "-" to restrict a particular version:

sslVersions = *, -ssl3

3. Configure forwarders to be compatible with your indexer. Changing or limiting the SSL versions (and restricting SSLv3) can create compatibility issues with forwarders, particularly those that run earlier versions of Splunk Enterprise. For forwarders running 6.2 you can mitigate compatibility issues by also updating each forwarder’s inputs.conf and web.conf settings in addition to your indexer.

Update any forwarders to 6.2 to be consistent with your indexer and the SSL settings (For purposes of backward compatibility, 6.0 can support up to tls1.1.)

Configure server.conf

Configure your server.conf file to accept connections with clients. In other words, you would configure web.conf by editing the sslVersions attribute so that it is the same as your the version configured in your server.conf file for your client(s).

For example:

[sslConfig]
sslVersions = tls1.1, tls1.2

Caveats to configuring splunkd with server.conf

If you have configured Splunk Enterprise deployment clients with SSL, confirm that you also configure Splunk Enterprise deployment servers to listen over SSL. If you do not, then deployment clients cannot connect to those servers, even if you have correctly configured certificates and TLS properly. To ensure that deployment clients can connect to deployment servers, review the server.conf configuration file on the deployment servers and confirm that it has at least the following setting:

[sslConfig]
enableSplunkdSSL = true
About creating certificates for Splunk

There are multiple ways you can create signed certificates, depending upon your organizations policies, your platform, and the tools that you are using. If you have already generated these certificates and key, or if you are experienced in generating certificates, you can skip this task and go directly to the configuration topic “Secure Splunk Web with your own certificate” in this manual.

Since self-signed certificates are signed by your organization, they are not contained in browser certificate stores. As a result, web browsers consider self-signed certificates “untrusted”. This produces a warning page to users and may even prevent access for the user.

Things to know about your certificates

Make sure you are using a compatible version of OpenSSL

Make sure that you are using the version of OpenSSL provided with Splunk by setting your environment to the version in $SPLUNK_HOME/lib in *nix or $SPLUNK_HOME/bin in Windows.

Decide between self-signed or third-party certificates

Self-signed certificates are best used for browser to Splunk Web communication that happens within an organization or between known entities where you can add your own CA to all browser stores that will contact Splunk Web. For any other scenario, CA-signed certificates are recommended. See "Get certificates signed by a third party for Splunk Web" for more information.

Remove your password from browser certificates

When you create a new private key for Splunk Web, you must generate a new private key and remove the password. We recommend that you generate a new private key especially for browser to Splunk Web encryption so that you do not remove the password from the keys you use elsewhere.

1. Generate a new private key:
2. When prompted, create a password.
3. Remove the password from your key. (Splunk Web does not currently support password-protected private keys.)

In *nix:

```
$SPLUNK_HOME/bin/splunk cmd openssl rsa -in mySplunkWebPrivateKey.key -out mySplunkWebPrivateKey.key
```

In Windows:

```
$SPLUNK_HOME\bin\splunk cmd openssl rsa -in mySplunkWebPrivateKey.key -out mySplunkWebPrivateKey.key
```

You can make sure your password is gone by issuing the following command:

In *nix:

```
```
In Windows:
$SPLUNK_HOME\bin\splunk cmd openssl rsa -in mySplunkWebPrivateKey.key -text

You should be able to read the contents of your certificate without providing a password.

Create a single PEM file for Splunk

Combine your server certificate and public certificates, in that order, into a single PEM file.

Set up certificate chains for Splunk

To use multiple certificates, append the intermediate certificate to the end of the server’s certificate file in the following order:

[ server certificate]
[ intermediate certificate]
[ root certificate (if required) ]

So for example, a certificate chain might look like this:

-----BEGIN CERTIFICATE-----
... (certificate for your server)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the intermediate certificate)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the root certificate for the CA)...
-----END CERTIFICATE-----

About cipher suites and TLS encryption

As of version 6.6, Splunk provides the following default cipher suites and TLS encryption. If you are upgrading from a previous version, you must update your existing certificates to be compatible with later versions.

inputs.conf

```
sslVersions = tls1.2
ecdhCurves = prime256v1, secp384r1, secp521r1
```

This configuration does not support Splunk 5.x. To add support for Splunk 5.x:

1. Set `sslVersions = tls`

2. Add the following ciphers to the end of the existing `cipherSuite`:

DHE-RSA-AES256-SHA:AES256-SHA:AES128-SHA:
AES256-SHA:AES128-SHA
outputs.conf

sslVersions = tls1.2
ecdhCurves = prime256v1, secp384r1, secp521r1

This configuration does not support Splunk 5.x. To add support for Splunk 5.x:

1. Set sslVersions to tls
2. Add the following ciphers to the end of the existing cipherSuite:

DHE-RSA-AES256-SHA:AES256-SHA:DHE-RSA-AES128-SHA:AES128-SHA:
AES256-SHA:AES128-SHA

server.conf

sslVersions = tls1.2
sslVersionsForClient = tls1.2
ecdhCurves = prime256v1, secp384r1, secp521r1

This configuration does not support Splunk 5.x. To add support for Splunk 5.x:

1. Set sslVersions = tls
2. Set sslVersionsForClient = tls
3. Append AES256-SHA to the existing cipherSuite.

applicationsManagement

sslVersions = tls1.2
ecdhCurves = prime256v1, secp384r1, secp521r1

This configuration does not support Splunk 5.x. To add support for Splunk 5.x:

1. Set sslVersions = tls
2. Add the following ciphers to the end of the existing cipherSuite:

DHE-RSA-AES256-SHA:AES256-SHA:DHE-RSA-AES128-SHA:AES128-SHA:
AES256-SHA:AES128-SHA

140
web.conf

sslVersions = tls1.2
ecdhCurves = prime256v1, secp384r1, secp521r1

This configuration does not support Windows Vista. To add support for Windows Vista:

1. Set sslVersions = tls

2. Add the following ciphers to the existing cipherSuite:

ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA:
ECDHE-RSA-AES128-SHA

ldap.conf

TLS_PROTOCOL_MIN: 3.1 for TLSv1.0, 3.2 for TLSv1.1, 3.3 for TLSv1.2.
TLS_PROTOCOL_MIN 3.3

This configuration does not support Windows Server 2008 R2. To add support for Windows Server 2008 R2:

1. Set TLS_PROTOCOL_MIN = TLS1.0/SSL3.1

2. Add the following ciphers to the existing TLS_CIPHER_SUITE:

ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA:
ECDHE-RSA-AES128-SHA

To enable TLS 1.2 support on Windows Server 2008 R2:

1. Add key:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Server
2. In the TLS 1.2\Server key, create the following:

DWORD (32-bit) Value ? DisabledByDefault; set to 0
DWORD (32-bit) Value ? Enabled; set to 1

How to prepare your signed certificates for Splunk authentication

Once you have your certificates, you must combine the server certificate and your keys into a single file that Splunk software can use.
If you do not have your certificates and need help getting them, we provide some basic examples using OpenSSL in the following topics:

- How to self-sign certificates.
- How to get certificates signed by a third party.

**Note:** Make sure your certificates and public key are in x509 format and that your private key is in RSA format.

**Create a single PEM file**

Combine your server certificate and public certificate, in that order, into a single PEM file.

For the examples here, we are using the file names described in "How to self-sign certificates" and "How to get certificates signed by a third party."

The following is an example for *nix:

```
cat myServerCertificate.pem myServerPrivateKey.key myCACertificate.pem > myNewServerCertificate.pem
```

The following is an example for Windows:

```
>type myServerCertificate.pem myServerPrivateKey.key myCACertificate.pem > myNewServerCertificate.pem
```

Once created, the contents of the file `myNewServerCertificate.pem` should contain, in the following order:

- The server certificate (`myServerCertificate.pem`)
- The private key (`myServerPrivateKey.key`)
- The certificate authority public key (`myCACertificate.pem`)

Here's an example of a properly concatenated certificate:

```
-----BEGIN CERTIFICATE-----
MIICUTCCAbocCQQDsaBgkqhkiG9w0BAQUFADBtMQswCQYDVQQGEwJV
...<Server Certificate>
...
8/PZr3EuXYk1c+N5hgIYys5a/HIn
-----END CERTIFICATE-----

-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,CFCECC7976725DE5
S+S+PcQ012Z1b871N3cBq7r/wuXPNDQ4ugtecCd3iGMV3B/WSOQAxCWzhe9JnI3l
...
<Server Private Key ? Passphrase protected>
...
-----END RSA PRIVATE KEY-----

-----BEGIN CERTIFICATE-----
MIICUTCCAbocCQQDsaBgkqhkiG9w0BAQUFADBtMQswCQYDVQQGEwJV
...
8/PZr3EuXYk1c+N5hgIYys5a/HIn
-----END CERTIFICATE-----
```
How to configure certificate chains

To use multiple certificates, append the intermediate certificate to the end of the server’s certificate file. You can add as many certificates you need to in decreasing order of hierarchy, up to the root.

The certificates should be concatenated in the following order:

[ server certificate]
[ intermediate certificate]
[ root certificate (if required) ]

So for example, a certificate chain might look like this:

-----BEGIN CERTIFICATE-----
... (certificate for your server)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the intermediate certificate)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the root certificate for the CA)...
-----END CERTIFICATE-----

In another example, when using Splunk Forwarder to Indexer Certificates that contain a Private Key, the completed certificate file might look like this:

-----BEGIN CERTIFICATE-----
... (certificate for your server)...
-----END CERTIFICATE-----
-----BEGIN RSA PRIVATE KEY-----
...<Server Private Key ? Passphrase protected>
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
... (certificate for your server)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the intermediate certificate)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the root certificate for the CA)...
-----END CERTIFICATE-----

Next steps

Now that you have the certificates you need, you must configure Splunk software to find and use them:

- See “Configure Splunk forwarding to use your own certificates” to learn more about configuring certificate authentication for forwarding.
- See “About securing inter-Splunk communication” to learn more about configuring certificate authentication for inter-Splunk communications.

Determine your cipher suite

You can select and specify a cipher suite for inter-Splunk, Splunk Web, and Splunk forwarder to indexer communications. You add your cipher suite by appending a line at the end of your server SSL configuration stanza.
The following is an example of how you would updated inputs.conf when configuring forwarder to indexer certificate authentication:

```
[splunktcp-ssl:9998]
[SSL]
password = password
requireClientCert = false
rootCA = $SPLUNK_HOME/etc/auth/cacert.pem
serverCert = $SPLUNK_HOME/etc/auth/server.pem
cipherSuite = AES256-SHA256:DHE-RSA-AES256-SHA256
```

To see which ciphers are available to you:

```
$SPLUNK_HOME/bin/splunk cmd openssl ciphers -v
$SPLUNK_HOME/bin/splunk cmd openssl ciphers -v "TLSv1.2"
$SPLUNK_HOME/bin/splunk cmd openssl ciphers -v "HIGH"
```

Cipher suites are available to you based on your version of OpenSSL. To see which version of OpenSSL you are running:

```
$SPLUNK_HOME/bin/splunk cmd openssl version
```

### Working with multiple intermediate certificates

To use multiple certificates, append the intermediate certificate to the end of the server's certificate file. You can include multiple intermediate certificates as a chain. You can add as many certificates as you need to in decreasing order of hierarchy, up to the root.

The certificates should be concatenated in the following order:

```
[ server certificate]
[ intermediate certificate]
[ intermediate certificate]
[ intermediate certificate]
[ intermediate certificate]
[ root certificate (if required) ]
```

So for example, a certificate chain might look like this:

```
-----BEGIN CERTIFICATE-----
... (certificate for your server)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the intermediate certificate)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the intermediate certificate)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the intermediate certificate)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the intermediate certificate)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the root certificate for the CA)...
-----END CERTIFICATE-----
```
Secure browser to Splunk Web communication

About securing Splunk Web

Information transmitted to Splunk Web mostly consists of search requests and results.

Note that browser to Splunk Web transmission does not always need to be secured. For example, if your users only access Splunk Web from a local browser behind the same firewall as Splunk Web, security may not be a concern. In this case simple encryption using Splunk's default certificates might be adequate.

- For information about the default certificate for Splunk Web, see Turn on encryption (https) with Splunk Web. or Turn on encryption (https) using web.conf.
- For information about SSL for forwarding with the default certificate, see Configure Splunk forwarding to use the default certificate.

To turn on basic encryption, see Turn on encryption (https) with Splunk Web.

On the other hand, if your Splunk configuration lives in a distributed environment where Splunk Web is accessed from browsers outside of firewalls from varied locations, stronger security should be implemented using signed certificates. For information about configuring Splunk Web to use signed certificates, see Secure Splunk Web using your own certificate.

There are several ways you can use signed certificates to improve security for your browser to Splunk Web communications:

- For secured encryption with authentication, you can replace the default certificate with a signed certificate.
  You replace the default certificate provided by Splunk with one that you request from a trusted Certificate Authority. This is the most secure option and recommended if security is a concern.
  For more information about obtaining CA certificates for Splunk deployments, see Get certificates signed by a third-party for Splunk Web.
  Note that you may also use self-signed certificates to secure authentication, however, because they are signed by you rather than a known and trusted Certificate Authority, browsers will not have you as a CA in their certificate store and as a result will not trust you or your certificates. For self-signed certificates to be effective you would need the ability to add your certificate to a the certificate store of every single browser that will access Splunk Web.
  For more information about creating self-signed certificates for Splunk deployments, see Self-sign certificates for Splunk Web.

- When you use a signed certificate, you can further strengthen your SSL configuration by turning on common name checking.
  Common name checking adds an extra layer of security by requiring that the common name provided in the certificates on each communicating instance are a match. You can enable common name checking when setting up your certificate and configure Splunk Enterprise to check for that common name when authenticating.

For more information about configuring Splunk Enterprise to use certificates and learn more about common name checking, see Secure Splunk Web using your own certificate.
**Turn on encryption (https) with Splunk Web**

This topic explains how to use Splunk Web to enable HTTPS for browser to Splunk Web communication. Splunk software can listen on HTTPS or HTTP, but not both.

The simple encryption that can be turned on in Splunk Web uses the default certificate that is provided in the "out of box" installation. Since every installation provides the same default certificate, this method is not highly secure. If security is a priority, change the default certificate and configure authentication for better security. See Secure Splunk Web with your own certificate for information about replacing the default certificates.

To enable HTTPS with Splunk Web:

1. In Splunk Web, select Settings > System > Server settings, and then click General Settings.
2. Under Splunk Web, for Enable SSL (HTTPS) in Splunk Web, select the Yes radio button.

By default, Splunk deployments point to the default certificates when encryption is turned on, so no further action is needed.


You must now prepend "https://" to the URL you use to access Splunk Web.

**Turn on encryption (https) using web.conf**

You can enable HTTPS through the web.conf configuration file. If it is not already present in your local directory, copy the default version of the file from $SPLUNK_HOME/etc/system/default to your local directory $SPLUNK_HOME/etc/system/local/ or your own custom application directory in $SPLUNK_HOME/etc/apps/. For information on configuration files in general, see About configuration files.

The encryption that can be turned on in the task described here is not secure. If security is a priority, change the default certificate and configure authentication for better security. See Secure Splunk Web with your own certificate for information about replacing the default certificates.

To enable HTTPS through web.conf:

1. Set the enableSplunkWebSSL attribute to true:

```
[settings]
httpport = <https port number>
enableSplunkWebSSL = true
```

**Note**: By default, Splunk software points to the default certificates when encryption is turned on.

2. Restart Splunk.

You must now prepend "https://" to the URL you use to access Splunk Web.
Secure Splunk Web with your own certificate

This example assumes that you have already generated self-signed certificates or purchased third-party certificates. If you have not done this and are unsure how to proceed, we've provided some simple examples:

- Self-sign certificates for Splunk Web.
- Get a certificates signed by a third-party for Splunk Web.

Before you begin: make sure your certificate and key are available from your folder. In this example we are using $SPLUNK_HOME/etc/auth/mycerts/:

- $SPLUNK_HOME/etc/auth/mycerts/mySplunkWebCertificate.pem
- $SPLUNK_HOME/etc/auth/mycerts/mySplunkWebPrivateKey.key

Configure Splunk Web to use the key and certificate files

1. In $SPLUNK_HOME/etc/system/local/web.conf (or any other applicable location, if you are using a deployment server), make the following changes to the [settings] stanza:

The following is an example of an edited settings stanza:

```
[settings]
enableSplunkWebSSL = true
privKeyPath = </home/etc/auth/mycerts/mySplunkWebPrivateKey.key >
serverCert = </home/etc/auth/mycerts/mySplunkWebCertificate.pem >
```

2. Restart Splunk:

```
# $SPLUNK_HOME/bin/splunk restart splunkd
```

Troubleshoot your Splunk Web authentication

If you are unable to verify your certificate configuration, you can use the web_service.log in $SPLUNK_HOME/var/log/splunk to view and troubleshoot any errors that occur upon restart.

Look for SSL configuration warnings. For example, if you provide an incorrect path to the server certificate declared in serverCert, Splunk Web fails to start and the following error appears:

```
```

Note: If the private key is provided in privKeyPath is password protected, no error is provided but your browser won't load Splunk Web.

See Self-sign certificates for Splunk Web or Get certificates signed by a third party for Splunk Web for information about removing your password.
Secure Splunk forwarder to indexer communication

About securing data from forwarders

Forwarders send raw data to your indexers. This data can be vulnerable to snooping and corruption. If data is forwarded outside of a closed or co-located network, or if your data is very sensitive you should use SSL certificates to secure your data.

Using the default certificates will discourage casual snoopers but could still leave you vulnerable because the root certificate that ships with Splunk software is the same root certificate in every download, and anyone with the same root certificate can authenticate. The default certificates are generated and configured at startup and can be found in $SPLUNK_HOME/etc/auth/.

Important: If you use the default certificates, keep in mind that they are set to expire three years after they are generated and new certificates must be created and configured at that time using one of the methods described in this manual.

For information about setting up SSL with the default certificate, see Configure Splunk forwarding to use the default certificate.

To ensure that no one can easily snoop on your traffic or send data to your indexers, we recommend that you use new signed certificates that are either self-signed or purchased from a third-party certificate authority. To configure your forwarders and indexers to use certificates, see Configure Splunk forwarding to use your own certificates.

There are several ways you can use self or CA-signed certificates to improve security for your forwarder to indexer:

- **You can replace the default certificates with certificates signed by your own root CA.**
  You replace the default certificate provided by Splunk with one that you generate and sign yourself. For information about generating and self-signing certificates, see How to self-sign certificates.
  
- **You can replace the default certificates with certificates signed by a trusted certificate authority.**
  See How to get certificates signed by a third-party.
  
- **You can further strengthen security by configuring common name checking.**
  Common name checking adds an extra layer of security by requiring that the common name provided in the certificates on each indexer match the common name specified in the configuration file on the forwarder. You can also configure multiple certificates with different common names and distribute them to your indexers. You enable common name checking when setting up your certificate. See Configure Splunk forwarding to use your own certificates for more information.

Configure Splunk forwarding to use the default certificate

The default root certificate that ships with Splunk software is the same root certificate in every download. Splunk software uses these certificates by default and you should not need to configure them unless they have been removed or corrupted.

Default certificates are not considered highly secure. Anyone who has downloaded Splunk Enterprise has server certificates signed by the same root certificate and are able to authenticate to your certificates. To ensure that no one can easily snoop on your traffic or wrongfully send data to your indexers, we recommend that you replace them with signed
Important: The default certificates are set to expire three years after they are generated and new certificates must be created and configured at that time using one of the methods described in this manual.

To configure your forwarders to use certificates signed by your own root CA or a third-party CA see Configure Splunk forwarding to use your own certificates.

In this topic we describe how to:

- Configure the indexer to use the default certificates that ship with Splunk software
- Configure the forwarder to use the default certificates that ship with Splunk software

Set up the indexer to use the default server certificate

1. In $SPLUNK_HOME/etc/system/local/inputs.conf (or the appropriate directory of any app you are using to distribute your forwarding configuration), set up the following stanzas:

```
[splunktcp-ssl:9997]
disabled = 0

[SSL]
serverCert = Absolute path to the certificate. The default certificate is $SPLUNK_HOME/etc/auth/server.pem
sslPassword = password
requireClientCert = false (There is no need validate the default server certificate)
```

2. Your server.conf should also have the following (skip this for Windows configurations):

```
[sslConfig]
sslRootCAPath = /opt/splunk/etc/auth/cacert.pem
```

3. Restart splunkd:

   `$SPLUNK_HOME/bin/splunk restart splunkd`

Configure the forwarder

Set up your forwarder to use the same default certificates as your indexer and configure the forwarder to send data to the configured listening port.

1. Define the following stanzas in $SPLUNK_HOME/etc/system/local/outputs.conf (or in the appropriate directory of any app you are using to distribute your forwarding configuration):

```
[tcpout:group1]
server = 10.1.12.112:9997

clientCert = $SPLUNK_HOME/etc/auth/server.pem
```

149
sslPassword = password

sslVerifyServerCert = false

2. Your forwarder’s server.conf should also have the following:

Linux:

[sslConfig]

sslRootCAPath = /opt/splunkforwarder/etc/auth/cacert.pem

Windows:

cacert.pem

cPath = $SPLUNK_HOME\etc\auth

3. Restart splunkd:

# $SPLUNK_HOME/bin/splunk restart splunkd

Next steps

Next, you should check your connection to make sure your configuration works. See “Validate your configuration” for more information.

Configure Splunk forwarding to use your own certificates

You can send data from your forwarders to your indexers using your own certificates. You can self-sign these certificates, or use a third party to sign them. Using your own certificates to secure Splunk communications involves the following procedures:

- Configuring indexers to use a new signed certificate, as described in this topic.
- Configuring forwarders to use a new signed certificate, as described in this topic.

Prerequisites

Before you can secure communications between Splunk indexers and forwarders, you must procure and prepare the certificates. You must satisfy the following conditions:

- The certificates that you procure are Privacy-Enhanced Mail (.pem) files, and that the format of those files conforms with the x.509 public key certificate standard.
- The certificate key must be in RSA security format.

You can also create multiple certificates that are signed by the same Certificate Authority (CA) with different common names, and distribute those to your indexers for added security. When you give the forwarder the CA public key, the forwarder trusts the CA, verifies the certificate of the CA, and matches one of the SSL common names or alternative names (as configured by either the sslCommonNameToCheck or sslAltNameToCheck settings in the forwarder configuration file.

If you need help on creating and preparing your own certificates, see the following topics for more information:

- About securing data from forwarders
• About securing inter-Splunk communication

**Configure your indexer to use a signed certificates**

1. Copy your server certificate and CA public certificate into an accessible folder on the indexer you want to configure. For example, you can use a destination folder of `$SPLUNK_HOME/etc/auth/mycerts/`

   If you configure `inputs.conf` or `outputs.conf` in an app directory, the indexer does not encrypt the password, and the clear-text value remains in the file. You might want to create different certificates (signed by the same root CA) to use when configuring SSL in app directories.

2. Configure `inputs.conf` on the indexer to use the new server certificate. Add the following stanzas to `$SPLUNK_HOME/etc/system/local/inputs.conf` (or the appropriate directory of any app you are using to distribute your forwarding configuration), stanzas:

   ```plaintext
   [splunktcp-ssl:9997]
   disabled=0
   [SSL]
   serverCert = <Absolute path to the certificate. The default certificate can be found at $SPLUNK_HOME/etc/auth/>
   sslPassword = <password associated with the server certificate, if it exists>
   requireClientCert = "true" if you want your indexer to require authentication from the client (which in this case is the forwarder), "false" otherwise
   sslVersions (Optional) = <string of accepted password SSL versions. Default: the recommended setting of "+,-ssl2", which is anything newer than SSLv2.>
   cipherSuite (Optional) = <cipher suite string. If not set, the indexer uses the default cipher string>
   sslCommonNameToCheck (Optional) = <commonName1>, <commonName2>, ...
   If provided, the indexer checks the common name of the client certificate against this list of names. If there is no match the Splunk instance is not authenticated. You must set the 'requireClientCert' setting to "true" to use this setting.
   sslAltNameToCheck (Optional) = <alternateName1>, <alternateName2>, ...
   If provided, the indexer checks the alternate name of the client certificate against this list of names. If there is no match the Splunk instance is not authenticated. requireClientCert setting must be set to "true" to use this setting.
   ```

   When you edit the file in `$SPLUNK_HOME/etc/system/local/inputs.conf`, the indexer encrypts the password and overwrites the clear-text server certificate password that you provided when you restarted Splunk Enterprise.

3. On indexers that do not run on Windows, configure `server.conf` and add the following:

   ```plaintext
   sslRootCAPath = <Absolute path to the CA certificate, for example, the default value is $SPLUNK_HOME/etc/auth/cacert.pem>
   ```

4. Restart the splunkd process:

   ```bash
   # $SPLUNK_HOME/bin/splunk restart splunkd
   ```
Configure your forwarders to use your certificates

1. Generate a new certificate (for example, client.pem).
2. Copy the new certificate and the CA public certificate myCACertificate.pem into an accessible folder on the forwarders you want to configure. For example, you can use a destination folder of $SPLUNK_HOME/etc/auth/mycerts/.

   If you configure inputs.conf or outputs.conf in an app directory, the forwarder does not encrypt the passwords and the clear-text value remains in the file. You might want to create different certificates (signed by the same root CA) to use when configuring SSL in app directories.

3. Define the [SSL] stanza in $SPLUNK_HOME/etc/system/local/outputs.conf (or in the appropriate directory of any app you use to distribute your forwarding configuration):

   [tcpout:group1]
   server=10.1.1.197:9997
   disabled = 0
   clientCert = <The full path to the client SSL certificate, in PEM format. If this value is provided, the connection will use SSL.>
   useClientSSLCompression = true (Disabling TLS compression can cause bandwidth issues.)
   sslPassword = <password for the client certificate>
   sslCommonNameToCheck (Optional) = <commonName1>, <commonName2>, ...
   sslVerifyServerCert (Optional) = "true" if you want to use SSL common name checking. Default: No common name checking.
   sslAltNameToCheck (Optional) = <alternateName1>, <alternateName2>, ...
   cipherSuite = (Optional) Splunk uses any specified cipher string for the input processors. If not set, Splunk uses the default cipher string provided by OpenSSL.

   When you save the file in $SPLUNK_HOME/etc/system/local/outputs.conf, Splunk Enterprise encrypts and overwrites the clear-text server certificate password on restart.

4. On forwarders that do not run on Windows, configure server.conf and add the following:

   [sslConfig]
   sslRootCAPath = <absolute path to the CA cert, for example, the default value is $SPLUNK_HOME/etc/auth/cacert.pem>

5. Restart the splunkd process.

   $SPLUNK_HOME/bin/splunk restart splunkd

Forward data over SSL to more than one indexer

If you need to forward data securely to multiple indexers, complete the following procedure:

1. On the forwarder where you want to send data to multiple indexers, edit outputs.conf on the forwarder.
2. In the target output group definition stanza for the forwarder, add a host:port entry for each indexer to which you want to send data over SSL. Separate multiple entries with commas.
3. Save the outputs.conf file and close it.
4. Restart the forwarder.
The following example `outputs.conf` file uses the same certificate for the indexer and the forwarders:

```plaintext
[tcpout]
[tcpout:group1]
server = 10.1.12.112:9997,10.1.12.111:9999
disabled = 0
clientCert = $SPLUNK_HOME/etc/auth/client.pem
useClientSSLCompression = true Defaults to the value set in the useClientSSLCompression
attribute set in server.conf.
sslPassword = <password for the client certificate>
sslCommonNameToCheck = indexercn.example.org
sslVerifyServerCert = true
```

**Forward data to multiple indexers using certificates with different common names**

You can create and configure one server certificate for each indexer by configuring `outputs.conf` on the forwarder with one server-specific `[SSLConfig]` stanza for each indexer.

If you have created one server certificate for each indexer and you have set a unique `sslCommonNameToCheck` or `sslAltNameToCheck` in each indexer certificate to be checked by the forwarders, you must configure one `[tcpout-server://host:port]` configuration stanza for each indexer in `outputs.conf`. This is so that you can specify which name to check for each indexer.

**Next steps**

Check your forwarder-indexer configuration to make sure it works. See Validate your configuration.

**Validate your configuration**

To verify your SSL connections in Splunk Web, try the following command:

```
index=_internal source=*metrics.log* group=tcpin_connections |
dedup hostname | table _time hostname version sourceIp destPort ssl
```

You can also `splunkd.log` to validate and troubleshoot your configuration. `splunkd.log` is located on your indexer and forwarder at `$SPLUNK_HOME/var/log/splunk/splunkd.log`.

On the indexer, look for the following or similar messages at the start-up sequence to verify a successful connection:

```
02-06-2011 19:19:01.552 INFO TcpInputProc - using queueSize 1000
02-06-2011 19:19:01.552 INFO TcpInputProc - SSL cipherSuite=ALL:!aNULL:!eNULL:!LOW:!EXP:
RC4+RSA:+HIGH:+MEDIUM
02-06-2011 19:19:01.552 INFO TcpInputProc - supporting SSL v2/v3
02-06-2011 19:19:01.555 INFO TcpInputProc - port 9997 is reserved for splunk 2 splunk (SSL)
02-06-2011 19:19:01.555 INFO TcpInputProc - Port 9997 is compressed
02-06-2011 19:19:01.556 INFO TcpInputProc - Registering metrics callback for: tcpin_connections
```

On the forwarder, look for the following or similar messages at the start-up sequence to verify a successful connection:

```
TcpOutputProc - Retrieving configuration from properties
```

153
TcpOutputProc - Using SSL for server 10.1.12.112:9997, clientCert=/opt/splunk/etc/auth/server.pem

TcpOutputProc - ALL Connections will use SSL with sslCipher=

TcpOutputProc - initializing single connection with retry strategy for 10.1.12.112:9997

Below is how a successful connection might appear in splunkd.log on the indexer:

TcpInputProc - Connection in cooked mode from 10.1.12.111
TcpInputProc - Valid signature found
TcpInputProc - Connection accepted from 10.1.12.111

Below is how a successful connection might appear in splunkd.log on the forwarder:

TcpOutputProc - attempting to connect to 10.1.12.112:9997...
TcpOutputProc - Connected to 10.1.12.112:9997

You can also check metrics.log for something similar to the following:

```shell
index=_internal host=heavy hostname=universal | stats last(connectionType) as connectionType
```

For help troubleshooting your configuration issues, see Troubleshoot your forwarder to indexer configuration in this manual.

**Troubleshoot your forwarder to indexer authentication**

1. Test your certificates:

   `openssl s_client -connect {server}:{port}`

   Port 8000, 8060, 8089, 9998, etc.
   A good certificate will return the following or something similar:

   ```shell
   Verify return code: 0 (ok)
   ```

2. Check `$SPLUNK_HOME/var/log/splunk/splunkd.log` (indexer and forwarder) for errors. On the indexer, check for the messages from the TCP input processor TcpInputProc. On the forwarder, check the messages from the TCP output processor TcpOutputProc.

3. Increase the logging level of the appropriate processors on the indexer and the forwarder in `$SPLUNK_HOME/etc/log.cfg`.


4. Restart Splunk Enterprise for these to take effect and observe the start-up sequence for the pertinent component. Most configuration issues are explicitly revealed by this method.

5. Check the SSL configuration using `btool` as follows:

   On the indexer:

   ```shell
   $SPLUNK_HOME/bin/splunk cmd btool inputs list --debug
   ```
On the forwarder:

$SPLUNK_HOME/bin/splunk cmd btool outputs list --debug

Common problems

- The path to the server certificate file set as the value of serverCert in inputs.conf is wrong, or the file cannot be read. This will generate the following error:

12-16-2010 16:07:30.965 ERROR SSLCommon - Can't read certificate file /opt/splunk/etc/auth/server.pem
errno=33558530 error:02001002:system library:fopen:No such file or directory

- The password to the RSA private key contained in the server certificate file is wrong.

12-07-2010 07:56:45.663 ERROR SSLCommon - Can't read key file /opt/splunk/etc/auth/server.pem
On *nix, you can manually test the password of the RSA key contained in the file with the command:

# openssl rsa -in /opt/splunk/etc/auth/server.pem -text

On Windows, you can manually test the password of the RSA key using the following command:

>openssl.exe rsa -in "c:\Program Files\Splunk\etc\auth\server.pem" -text
Secure distributed environments

About securing inter-Splunk communication

This chapter discusses the following “Splunk to Splunk” types of communications and how you can secure them:

- Securing distributed search heads and peers
- Securing deployment server and clients
- About securing clusters

Configure secure communications between Splunk instances with updated cipher suite and message authentication code

Version 7.2 of Splunk Enterprise, Splunk Light, and the universal forwarder introduces a new cipher suite and message authentication code (MAC), that it uses for data encryption and secure communications between Splunk software instances. The cipher suite and MAC replace the current cipher that Splunk software has used for these types of communications.

The new cipher suite uses a stronger, more secure mechanism for encrypting and decrypting file-based data. It uses a separate MAC to handle node authentication during communications between Splunk instances. Previously, the legacy cipher handled both types of operations.

By default, version 7.2 and higher of Splunk software uses the new cipher suite and MAC for these types of communication. They have also been configured to use the existing legacy cipher for backward compatibility in both data encryption and node authentication with versions lower than 7.2.

This backward compatibility lets you plan upgrades for your Splunk and universal forwarder instances while maintaining secure communications between the instances. Later, after you have upgraded all instances in your deployment to version 7.2 or higher, you can disable the legacy cipher and your Splunk instances continue internal communications using only the new cipher suite and MAC.

The main points for understanding how to use the ciphers are:

- Versions of on-premises Splunk software lower than 7.2 do not have the new cipher suite and MAC, and cannot have either integrated within them.
- You must run version 7.2 or higher to take advantage of the stronger encryption capabilities that the new cipher suite and MAC offer.
- Splunk-to-Splunk communication happens using either the legacy cipher or the new cipher suite/MAC, not both
- A Splunk instance that uses only the legacy cipher cannot communicate at all with a Splunk instance that uses only the new cipher suite and MAC.

Configure cipher usage and decryption with configuration files

By default, versions of Splunk software 7.2 and higher use both the legacy cipher and new cipher suite and MAC for communications between instances, depending on the version of the instance they communicate with:

- For backward compatibility, when 7.2 and higher instances of Splunk software communicate with instances of
Splunk software lower than 7.2, they use only the legacy cipher.
• When 7.2 and higher instances communicate with other 7.2 and higher instances, they use only the new cipher suite and MAC.

You must configure the cipher settings with the server.conf configuration file on instances of Splunk that are version 7.2 and higher only. You cannot configure cipher settings on versions lower than 7.2. You also cannot configure the ciphers in Splunk Web.

After you make configuration changes, you must restart Splunk instances for the changes to take effect.

Configure which ciphers the Splunk instance uses

1. In the $SPLUNK_HOME/etc/system/local directory, create server.conf if it does not already exist. Do not create this file in $SPLUNK_HOME/etc/default as this file gets overwritten whenever you upgrade.
2. Use a text editor to open the file.
3. Create a [node_auth] stanza and add the following block of text to it depending on your specific needs:

<table>
<thead>
<tr>
<th>One or more instances runs lower than 7.2</th>
<th>All instances run 7.2 or higher</th>
</tr>
</thead>
<tbody>
<tr>
<td>[node_auth] signatureVersion = v1,v2</td>
<td>[node_auth] signatureVersion = v2</td>
</tr>
</tbody>
</table>

4. Save the file and close it.
5. Restart the Splunk instance. The instance initiates Splunk-to-Splunk communications using the ciphers that you specified.

Configure legacy cipher decryption options

1. In the $SPLUNK_HOME/etc/system/local directory, create server.conf if it does not already exist. Do not create this file in $SPLUNK_HOME/etc/default as this file gets overwritten whenever you upgrade.
2. Use a text editor to open the file.
3. Under the [general] stanza, add the legacyCiphers setting, based on the versions of Splunk software that run in your Splunk deployment:

<table>
<thead>
<tr>
<th>One or more instances runs lower than 7.2</th>
<th>All instances run 7.2 or higher</th>
</tr>
</thead>
<tbody>
<tr>
<td>legacyCiphers = decryptOnly</td>
<td>legacyCiphers = disabled</td>
</tr>
</tbody>
</table>

4. Save the file and close it.
5. Restart the Splunk instance. The instance decrypts configurations in accordance with the legacyCiphers setting.

Update the splunk.secret key file on instances to use the new cipher

The splunk.secret file controls the encryption of various elements of a Splunk deployment, including SSL and LDAP passwords for configuration files, to ensure secure communications between Splunk instances. In Splunk Enterprise version 7.2.2 and higher, you can rotate the splunk.secret file so that it triggers the re-encryption of file-based passwords with the new cipher that was introduced in Splunk Enterprise version 7.2.

The splunk.secret file can be recreated in one of the following ways:

• By using a CLI command.
• By making an HTTP POST request to a REpresentational State Transfer (REST) endpoint.

The procedure differs slightly based on the type of Splunk Enterprise instance you run. The following section describes the procedure you must run to successfully update the splunk.secret key file.
If you update the splunk.secret key file, any tokens on the Splunk instance become invalid immediately. You cannot undo this change. After you update the key file, issue new authentication tokens to parties who previously had valid tokens and need to regain access.

If the authentication tokens feature is in use, but is temporarily disabled when you rotate the splunk.secret file, then you must manually delete the invalidated tokens after you re-enable the tokens feature.

See Set up authentication with tokens for information on authentication tokens and how to create and manage them.

**Update the key file on a single instance of Splunk Enterprise with the CLI**

- From a shell, command prompt, or PowerShell window, run the following command:

  ```bash
  splunk rotate splunk-secret
  ```

  Splunk Enterprise creates a new splunk.secret file and re-encrypts all secure configurations using the new file, based on the new cipher suite.

**Update the key file on a single instance of Splunk Enterprise using REST**

- Make the following HTTP POST request to the `/services/server/security/splunk-secret/rotate` REST endpoint:

  ```bash
  ```

**Update the key file on a search head cluster (SHC) with the CLI**

Prior to updating the splunk.secret on a search head cluster, confirm the following conditions are true for the cluster:

- The cluster is healthy, meaning that the search head cluster captain and all its peers are up, and that configuration replication occurs regularly. The CLI command fails if it detects that the cluster is not healthy.

You can check configuration status with the `splunk show shcluster-status --verbose` CLI command and view the `last_conf_replication` entry in the output of that command.

- The cluster is not in a rolling restart or upgrade. The CLI command fails if it detects that the cluster is in either of these states.
- All peers in the cluster run the same version of Splunk Enterprise. The CLI command fails if it detects any version mismatches among the peers.

The cluster captain always determines the shared splunk.secret key file for a cluster. If you attempt to rotate the key file as a cluster peer becomes a captain, a situation can occur where the newly-elected captain pushes the original key file to the rest of the cluster peers. Confirming that the cluster is healthy before starting a key file rotation prevents this possibility.

If you run a SHC deployer instance to distribute configurations, that instance does not use the same splunk.secret file as the other SHC members. If you push pre-encrypted configurations to SHC members using the deployer, you must first re-encrypt those configuration with the new shared splunk.secret file before you distribute them with the SHC deployer.

Consider running the key file update process in a maintenance window. When you update the key file on a search head cluster, the cluster peers remove and then re-add themselves to the cluster using the updated key file, and any scheduled searches that the cluster has configured are not available during this period. The amount of time that scheduled searches
are not available depends on the size of the cluster.

```bash
splunk rotate shcluster-splunk-secret
```

The search head cluster captain generates the new key file and re-encrypts its passwords with that file. As peers report in to the captain at standard intervals, the captain directs each peer to remove and re-add itself to the cluster. When the peers reconnect, the captain then sends the new key file to each peer, which then subsequently re-encrypt their passwords with the new key file.

**Update the key file on a search head cluster using REST**

Before attempting to update the key file on a search head cluster, read the information in "Update the key file on a search head cluster (SHC) with the CLI" for important prerequisites to making the update.

```bash
curl -u <splunk username>:<splunk password> https://<splunk server>:<management port>/services/shcluster/captain/control/control/rotate-splunk-secret -X POST
```

**Troubleshoot cipher version mismatches**

Splunk instances that run a version lower than 7.2 cannot communicate securely with instances that run version 7.2 and higher if the higher version has been configured to use the new cipher suite and MAC only. Lower version instances cannot use the new cipher suite at all. A 7.2 or higher version instance must be configured to use at least the legacy cipher to communicate with lower version instances.

If a lower version Splunk instance attempts to connect to a higher version instance that has only been configured to use the new cipher suite and MAC, the instance that makes the connection logs the following error in `$SPLUNK_HOME/var/log/splunk/splunkd.log`:

```error
ERROR HttpClientRequest - Caught exception while parsing HTTP reply: Unexpected character while looking for value: '<'
```

**To fix the problem, do one of the following:**

- If the connecting Splunk instance runs a version lower than 7.2, upgrade the instance to 7.2 or higher, if possible.
- Similarly, if the connecting instance runs version 7.2 or higher and the instance being connected to runs a version lower than 7.2, upgrade the lower versioned instance to 7.2 or higher, if possible.
- If the connecting instance runs a version lower than 7.2 and the instance being connected to runs version 7.2 or higher, edit `server.conf` on the 7.2 or higher instance and confirm that the `signatureVersion` setting is set to `v1,v2` and the `legacyCiphers` setting is set to `decryptOnly`. These are the defaults for 7.2 or higher instances.

**Securing distributed search heads and peers**

**Distributed search** configurations share search information, knowledge objects and app and configuration information over the management port.

Communication between search heads and peers relies on public-key encryption. Upon startup, Splunk software generates a private key and public key on your Splunk installation. When you configure distributed search on the search
head, the public keys are distributed by search heads to peers and those keys are used to secure communication. This default configuration provides built-in encryption as well as data compression that improves performance.

It is possible to swap these generated keys out with your own keys, though the existing keys are generally considered adequate for most configurations.

To configure public-key encryption for distributed search setups, you create your keys and distribute them to your search heads and peers. To learn more about distributing key files to distributed search peers, look in the section on configuring distributed search in the Distributed Search manual: "Distribute the key files".

Secure your deployment server and clients using certificate authentication

You may find certificate authentication necessary in certain distributed configurations, perhaps where sensitive server configuration data is sent to a variety of locations outside your firewall. You can manually configure each indexer to communicate with your Deployment Server:

Note: The deployment server cannot properly push certificates to peers. You must configure each member separately.

1. Create one or more certificates using the same root CA.
2. Distribute the certificates to your deployment server and clients.
3. Edit server.conf to provide the location of your certificates:

   [sslConfig]
   enableSplunkdSSL = true
   sslVersions = Defaults to "*,-ssl2" (anything newer than SSLv2). This is the recommended setting.
   serverCert = The full path to the PEM format server certificate file. Default certificates
   ($SPLUNK_HOME/etc/auth/server.pem) are generated by Splunk at start. To secure Splunk,
   you should replace the default cert with your own PEM file.
   sslPassword = password
   sslRootCAPath = absolute path to the operating system's root CA (Certificate Authority) PEM
   format file containing one or more root CA. Do not configure this attribute on Windows.
4. Edit server.conf to authenticate against your certificates by adding the following attribute to the [sslConfig] stanza in previous step:

   requireClientCert = true

   Important: This requireClientCert is set to "false" by default. If you change it to true to force Splunk to check your
   client's certificates, Splunk Web and the CLI will also be checked for certificates. Your CLI connection will no longer work
   because your CLI is unable to present a certificate as a client.
5. Edit web.conf to present a certificate signed by the same root CA so that Splunk Web can connect to the server.

The following is an example of an edited settings stanza:

   [settings]
   enableSplunkWebSSL = true
privKeyPath = etc/auth/splunkweb/mySplunkWebPrivateKey.key
serverCert = etc/auth/splunkweb/mySplunkWebCertificate.pem
cipherSuite = <your chosen cipher suite (optional)>

Note: Splunk Web does not support passwords, so you must remove the password from the private key. For more information, see “Get certificates signed by a third party for Splunk Web.”

Secure your clusters with pass4SymmKey

Splunk provides a security key to let your search head or indexer clustering nodes authenticate with each other. When you set up an indexer cluster or search head cluster, you assign the same key to each node in the cluster. You use the pass4SymmKey setting in the server.conf file. You can set the key through Splunk Web, the configuration file, or the CLI.

pass4SymmKey controls authentication between Splunk instances and does not manage user access.

Configure pass4SymmKey for search head clustering

Configure pass4SymmKey when you deploy the search head cluster. See Deploy a search head cluster.

For details on configuring pass4SymmKey on a search head cluster, including how to set it post-deployment, see Set a security key for the search head cluster.

Configure pass4SymmKey for indexer clustering

Configure pass4SymmKey when you deploy the indexer cluster, while enabling the master node. See Enable the indexer cluster master node.

For more details on setting pass4SymmKey on an indexer cluster, see Configure the security key.

How apps encrypt pass4SymmKey

When you specify pass4SymmKey in clear-text for an app directory on a Splunk instance (for example: etc/apps/myapp/default/server.conf), the software writes an obfuscated version of the key to the local file (in this example, system/local/server.conf) when you restart the instance. Configuration files in the default directory are generally read-only, and the software writes the information to the local file, which is editable.

Placing a password directly into an app's local directory (for example: etc/apps/myapp/local/server.conf), replaces it with the encrypted version.

When the configuration is listed using curl or a splunkd endpoint, the pass4SymmKey appears encrypted. If the configuration location is read-only, Splunk software likewise writes to local.

Use OpenSSL to generate a random passphrase for pass4SymmKey

You can use the OpenSSL utilities that come with Splunk software to generate a passphrase that you can use with pass4SymmKey.

For the strongest security, select a passphrase that is at least 12 characters long and checks out against a dictionary of known bad passphrases, like abc123, password, qwerty, admin, and so on. The OpenSSL utility that comes with Splunk software lets you randomly generate a passphrase that you can then use to set pass4SymmKey with on all nodes of your Splunk deployment.
1. On a Splunk instance, open a shell prompt.
2. Change to the $SPLUNK_HOME/bin directory.
3. Run the following command to generate a random 12-character passphrase:
   
   splunk cmd openssl rand -base64 9

4. Copy the output of the command to your clipboard.
5. For all machines that you want to use the new passphrase:
   1. Edit $SPLUNK_HOME/etc/system/local/server.conf.
   2. Set pass4SymmKey = <new passphrase that you just generated>
   3. Save $SPLUNK_HOME/etc/system/local/server.conf.
   4. Restart Splunk software.
Audit Splunk Enterprise activity

Use Splunk Enterprise to audit your system activity

Knowing what is happening in your system is vital to keeping it secure. To make the most of your system and keep it secure, we recommend the following best practices:

- Perform a periodic review of Splunk access and audit logs.
- Perform a periodic review of Splunk server audit and security logs.
- Perform a periodic review of all Splunk users and roles.

Audit Splunk activity

With auditing enabled, Splunk logs distinct events to the audit index (index=_audit). Interactions with Splunk such as searches and configuration changes generate audit events.

What's in an audit event?

- Timestamp:
  - date and time of the event.
- User information:
  - the user who generated the event.
  - If the event contains no user information, Splunk sets the user to whoever is currently logged in.
- Additional information:
  - available event details -- what file, success/denial, etc.

Activities that generate audit events

Audit events are generated from:

- all files in Splunk's configuration directory $SPLUNK_HOME/etc/*
  - files are monitored for add/change/delete using the file system change monitor.
- system start and stop.
- users logging in and out.
- adding / removing a new user.
- changing a user's information (password, role, etc).
- execution of any capability in the system.
  - capabilities are listed in authorize.conf

Audit event storage

Splunk stores audit events locally in the audit index (index=_audit). Audit events are logged in the log file: $SPLUNK_HOME/var/log/splunk/audit.log.

If you have configured Splunk as a forwarder in a distributed setting, audit events are forwarded like any other event.
Use audit events to secure Splunk Enterprise

Use Splunk to search your audit log to review and alert on administrative access:

1. Audit user access

index=_audit action=log* action="login attempt"

2. Find the location from which users are accessing Splunk:

index=_internal | eval timestamp=strftime(_time, "%Y-%m-%d %H:%M:%S.%Q") | table timestamp, user, clientip

3. Consider setting up a real time alert on administrative user access:

(index=_audit action=log* action="login attempt") OR (index=_internal) user=admin

See the Alerting Manual for more information about creating alerts.

4. You can also create dashboards that collect and display your chosen searches, see Build Dashboards in Splunk Web in the Dashboards and Visualizations Manual.

Manage data integrity

The Splunk Enterprise data integrity control feature provides a way to verify the integrity of data that is indexed.

When you enable data integrity control for an index, Splunk Enterprise computes hashes (using SHA 256) on every slice of data and stores those hashes so that you can go back later and verify the integrity of your data.

How it works

When you enable data integrity control, Splunk Enterprise computes hashes on every slice of newly indexed raw data and writes it to a l1Hashes file. When the bucket rolls from hot to warm, Splunk Enterprise computes a hash on the contents of the l1Hashes and stores the computed hash in l2Hash. Both hash files are stored in the rawdata directory for that bucket.

Note that data integrity control hashes newly indexed data, data coming from a forwarder should be secured and encrypted with SSL. For more information, see About securing Splunk with SSL.

Check your hashes to validate your data

To check Splunk Enterprise data, run the following CLI command to verify the integrity of an index or bucket:

```bash
./splunk check-integrity -bucketPath [ bucket path ] [ -verbose ]
./splunk check-integrity -index [ index name ] [ -verbose ]
```

Configure data integrity control

To configure Data Integrity Control, edit indexes.conf to enable the enableDataIntegrityControl attribute for each index. The default value for all indexes is false (off).

```
enableDataIntegrityControl=true
```
Data Integrity in clustered environments

In a clustered environment, the cluster master and all the peers must run Splunk Enterprise 6.3 or later to enable accurate index replication.

Optionally modify the size of your data slice

By default, data slices are set to 128kb, which means that a data slice is created and hashed every 128KB. You can optionally edit `index.conf` to specify the size of each slice.

```
rawChunkSizeBytes = 131072
```

Store and secure your data hashes

For optimal security, you can optionally store your hashes outside the system where the data is hosted, such as a different server. To avoid naming conflicts, store your secured hashes in separate directories.

Regenerate hashes

If you lose your hashes for a bucket, Use the following CLI command to re-generate hash files on a bucket or index. This command extracts the hashes embedded in the journal:

```
./splunk generate-hash-files -bucketPath [ bucket path ] [ verbose ]
./splunk generate-hash-files -index [ index name ] [ verbose ]
```
Best practices for Splunk Enterprise security

Safeguards for risky commands

Splunk Enterprise contains built-in search processing language (SPL) safeguards to warn you when you are about to unknowingly run a search that contains commands that might be a security risk. This warning appears when you click a link or type a URL that loads a search that contains risky commands.

The warning does not appear when you create ad hoc searches.

This warning alerts you to the possibility of unauthorized actions by a malicious user. Unauthorized actions include:

- Copying or transferring data (data exfiltration)
- Deleting data
- Overwriting data

A possible scenario when this might occur is when a malicious person creates a search that includes commands that exfiltrate or damage data. The malicious person then sends an unsuspecting user a link to the search. The URL contains a query string (q) and a search identifier (sid), but the sid is expired. The malicious person hopes the user will use the link and the search will run.

Commands that trigger the warning

The commands that trigger this warning are listed here:

- collect
- crawl
- dump
- delete
- input
- outputcsv
- outputlookup
- run
- runshellscript
- script
- sendalert
- sendemail
- tscollect

Actions in the warning dialog box

Instead of running the search immediately, Splunk Enterprise analyzes the search for risky commands. If one or more risky commands are identified, a warning dialog box appears. You have the option to cancel, run, or investigate the search.

Cancel

Closes the warning dialog box. The search does not run and the search is removed from the Search bar. Closing the dialog box, by clicking the Close button (X), is the same as clicking Cancel.

Run

Runs the search.

Investigate
Displays the search in the Search bar so that you can review the SPL. Use this option to copy the syntax of the search. Send a copy of the search, along with any information about the source of the link, to your system administrator.

**Turning off the warning**

Only users with Write permission can edit the `web.conf` file to turn off the warning dialog box.

You can turn off the warning for a specific command, or for all of the risky commands.

**Turn off the warning for a specific command**

1. Copy the `commands.conf` file, which is located in the `$SPLUNK_HOME/etc/system/default` directory.
2. Paste the copy of the file in the `$SPLUNK_HOME/etc/system/local` directory.
3. Locate the command and change the setting from `is_risky = true` to `is_risky = false`.
4. Restart Splunk Enterprise.

**Turn off the warning for all of the commands**

1. Open the `web.conf` file. This file is located in the `$SPLUNK_HOME/etc/system/default/` directory.
2. Change the `enable_risky_command_check` parameter to `false`.
3. Restart Splunk Enterprise.

**See also**

In the *Admin Manual*:

- About configuration files
- `commands.conf` file
- `web.conf` file

**Splunk server tokens**

If a forwarder TCP token is corrupt or rejected, the indexer that receives the token generates error messages in its logs. If you do not locate the bad token, that forwarder tries to use it indefinitely.

To locate the bad forwarder token, increase the logging level of the indexer:

1. Open a shell prompt.
2. Using a text editor, edit `$SPLUNK_HOME/etc/log.cfg` as follows:

```
category.TcpOutputProc=DEBUG
category.TcpInputConfig=DEBUG
category.TcpInputProc=DEBUG
```
3. Save the file and close it.
4. Restart the indexer.

When a token that a forwarder sends matches the token that the indexer receives, the following messages are generated:

Indexer:
Avoid malicious CSV files in searches

If you export your search results as a CSV and then open it in Excel/OpenOffice, any fields that start with an ‘=’ character will be executed.

For example:

1. User runs stats count | eval trick="=1+1".
2. User exports the results as a CSV file.
3. User load the new CSV file in Excel.
4. Field in Excel has a value of 2 when it should have a value of ’=1+1.

To avoid this, you can do one of the following:

- For any cell that starts with the following characters, add a space to the beginning and remove any tab characters (0x09) in the cell.
  - =
  - -
  - “
  - @
  - +
- Append any cell beginning with the previously listed characters with an apostrophe (’).
- Make sure users do not have the “export_results_is_visible” capability (version 6.4 and later only). This capability displays the export results button, without this capability it is not possible to generate CSV files at all.
Appendix A: How to get SSL certificates

How to self-sign certificates

This topic describes one way you can use OpenSSL to self-sign certificates for securing forwarder-to-indexer and Inter-Splunk communication.

If you already possess or know how to generate the needed certificates, you can skip this topic and go directly to the configuration steps, described later in this manual:

- How to prepare your signed certificates for Splunk
- Configure Splunk forwarding to use your own certificates
- About securing inter-Splunk communication

Self-signed certificates are best for data communication that occurs within an organization or between known entities. If you communicate with unknown entities, we recommend CA-signed certificates to secure your data.

Before you begin

In this discussion, $SPLUNK_HOME refers to the Splunk Enterprise installation directory:

- For Windows, Splunk software is installed in C:\Program Files\splunk by default
- For most Unix platforms, the default installation directory is at /opt/splunk
- For Mac OS, it is /Applications/splunk

See the Administration Guide to learn more about working with Windows and *nix.

Create a new directory for your certificates

Create a new directory to work from when creating your certificates. In our example, we are using $SPLUNK_HOME/etc/auth/mycerts:

```
# mkdir $SPLUNK_HOME/etc/auth/mycerts
# cd $SPLUNK_HOME/etc/auth/mycerts
```

This ensures you do not overwrite the Splunk-provided certificates that reside in $SPLUNK_HOME/etc/auth.

Create the root certificate

First you create a root certificate that serves as your root certificate authority. You use this root CA to sign the server certificates that you generate and distribute to your Splunk instances.

Generate a private key for your root certificate

1. Create a key to sign your certificates.

In *nix:

```
$SPLUNK_HOME/bin/splunk cmd openssl genrsa -aes256 -out myCA_PrivateKey.key 2048
```

In Windows:
2. When prompted, create a password for the key.

When the step is completed, the private key myCAPrivateKey.key appears in your directory.

**Generate and sign the certificate**

1. Generate a new Certificate Signing Request (CSR):

   In *nix:
   ```bash
   $SPLUNK_HOME/bin/splunk cmd openssl req -new -key myCAPrivateKey.key -out myCACertificate.csr
   ```
   In Windows:
   ```bash
   $SPLUNK_HOME\bin\splunk cmd openssl req -new -key myCAPrivateKey.key -out myCACertificate.csr
   ```

2. When prompted, enter the password you created for the private key in $SPLUNK_HOME/etc/auth/mycerts/myCAPrivateKey.key.

3. Provide the requested certificate information, including the common name if you plan to use common name checking in your configuration.

A new CSR myCACertificate.csr appears in your directory.

4. Use the CSR myCACertificate.csr to generate the public certificate:

   In *nix:
   ```bash
   $SPLUNK_HOME/bin/splunk cmd openssl x509 -req -in myCACertificate.csr -sha512 -signkey myCAPrivateKey.key -CAcreateserial -out myCACertificate.pem -days 1095
   ```
   In Windows:
   ```bash
   $SPLUNK_HOME\bin\splunk cmd openssl x509 -req -in myCACertificate.csr -sha512 -signkey myCAPrivateKey.key -CAcreateserial -out myCACertificate.pem -days 1095
   ```

5. When prompted, enter the password for the private key myCAPrivateKey.key.

A new file myCACertificate.pem appears in your directory. This is the public CA certificate that you will distribute to your Splunk instances.

**Create the server certificate**

Now that you have created a root certificate to serve as your CA, you must create and sign your server certificate.

**A note about common name checking**

This topic shows you how to create a new private key and server certificate.

You can distribute this server certificate to all forwarders, indexers as well your Splunk instances that communicate on the management port. If you plan to use a different common name for each instance, you simply repeat the process described here to create different certificates (each with a different common name) for your Splunk instances.

For example, if configuring multiple forwarders, you can use the following example to create the certificate myServerCertificate.pem for your indexer, then create another certificate myForwarderCertificate.pem using the same root CA and install that certificate on your forwarder. Note that an indexer will only accept a properly generated and
configured certificate from a forwarder that is signed by the same root CA.

See Configure Splunk forwarding to use your own certificates for more information about configuring your forwarders and indexers.

*Generate a key for your server certificate*

1. Generate a new RSA private key for your server certificate. In this example we are again using AES encryption and a 2048 bit key length:

   In *nix:

   ```bash
   $SPLUNK_HOME/bin/splunk cmd openssl genrsa -aes256 -out myServerPrivateKey.key 2048
   ```

   In Windows:

   ```bash
   $SPLUNK_HOME\bin\splunk cmd openssl genrsa -aes256 -out myServerPrivateKey.key 2048
   ```

2. When prompted, create a new password for your key.

   A new key `myServerPrivateKey.key` is created. You will use this key to encrypt the outgoing data on any Splunk Software instance where you install it as part of the server certificate.

*Generate and sign a new server certificate*

1. Use your new server private key `myServerPrivateKey.key` to generate a CSR for your server certificate.

   In *nix:

   ```bash
   $SPLUNK_HOME/bin/splunk cmd openssl req -new -key
   myServerPrivateKey.key -out myServerCertificate.csr
   ```

   In Windows:

   ```bash
   $SPLUNK_HOME\bin\splunk cmd openssl req -new -key
   myServerPrivateKey.key -out myServerCertificate.csr
   ```

2. When prompted, provide the password to the private key `myServerPrivateKey.key`.

3. Provide the requested information for your certificate, including a Common Name if you plan to configure Splunk Software to authenticate via common-name checking.

   A new CSR `myServerCertificate.csr` appears in your directory.

4. Use the CSR `myServerCertificate.csr` and your CA certificate and private key to generate a server certificate.

   In *nix:

   ```bash
   $SPLUNK_HOME/bin/splunk cmd openssl x509 -req -in
   myServerCertificate.csr -SHA256 -CA myCACertificate.pem -CAkey myCAPrivateKey.key
   -CAcreateserial -out myServerCertificate.pem -days 1095
   ```

   In Windows:

   ```bash
   $SPLUNK_HOME\bin\splunk cmd openssl x509 -req -in
   myServerCertificate.csr -SHA256 -CA myCACertificate.pem -CAkey myCAPrivateKey.key
   -CAcreateserial -out myServerCertificate.pem -days 1095
   ```

5. When prompted, provide the password for the certificate authority private key `myCAPrivateKey.key`. Make sure to sign this with your private key and not the server key you just created.
A new public server certificate `myServerCertificate.pem` appears in your directory.

Next steps

You should now have the following files in the directory you created, which is everything you need to configure indexers, forwarders, and Splunk instances that communicate over the management port:

- `myServerCertificate.pem`
- `myServerPrivateKey.key`
- `myCACertificate.pem`

Now that you have the certificates you need, prepare your server certificate (including appending any intermediate certificates), and then configure Splunk to find and use them:

- See [How to prepare your signed certificates for Splunk](#) to learn how to set up your certificates to work with Splunk.
- See [Configure Splunk forwarding to use your own certificates](#) to learn more about configuring certificate authentication for forwarding.
- See [About securing inter-Splunk communication](#) to learn more about configuring certificate authentication for Splunk to Splunk communications.

**How to get certificates signed by a third-party**

This topic describes one way you can use the version of OpenSSL that ships with Splunk Enterprise to obtain third-party certificates that you can use to secure your forwarder-to-indexer and inter-Splunk communication.

To get certificates that you can use to secure for browser-to-Splunk Web communication, see [Get certificates signed by a third-party for Splunk Web](#).

If you already possess or know how to generate the certificates you can, skip this topic and go directly to the configuration steps, which are described later in this manual:

- [Configure Splunk forwarding to use your own certificates](#)
- [About securing inter-Splunk communication](#)

**Note:** If you plan to use multiple common names in your configurations, you can repeat the steps described here to create a different server certificate using the same root CA for each instance with it’s own common name and then configure your Splunk instances to use them. See [Configure Splunk forwarding to use your own certificates](#) for more information about configuring your forwarders and indexers.

**Before you begin**

In this discussion, `$SPLUNK_HOME` refers to the Splunk Enterprise installation directory. We recommend that you follow this convention, but if you do not, you should replace `$SPLUNK_HOME` with your installation directory when using these examples.

For Windows, you might need to set this variable at the command line or in the Environment tab in the System Properties dialog.

Default home directories depend on your platform:
For Windows, the Splunk Enterprise directory is at C:\Program Files\Splunk by default.
For most *nix platforms, the default installation directory is at /opt/splunk.
For Mac OS, it is /Applications/splunk.

See the Administration Guide to learn more about working with Windows and *nix.

Create a new directory for your certificates

Create a new directory for your new certificates. In our example, we are using $SPLUNK_HOME/etc/auth/mycerts:

```bash
# mkdir $SPLUNK_HOME/etc/auth/mycerts
# cd $SPLUNK_HOME/etc/auth/mycerts
```

When you make a new folder you protect the existing certificates and keys in $SPLUNK_HOME/etc/auth. Working in a new directory protects the default certificates and lets you use them for other Splunk Software components as necessary.

Request your server certificate

Create and sign a Certificate Signing Request (CSR) to send to your Certificate Authority.

**Important:** This example shows you how to create a new private key and request a server certificate. You can distribute this server certificate to all forwarders, indexers as well your Splunk instances that communicate on the management port. If you want to use a different common names for each instance, you simply repeat the process described here to create different certificates (each with a different common name) for your Splunk instances.

For example, when configuring multiple forwarders, you can use the following example to create the certificate myServerCertificate.pem for your indexer, then create another certificate myForwarderCertificate.pem using the same root CA and install that certificate on your forwarder. An indexer will only accept a properly generated and configured certificate from a forwarder that is signed by the same root CA.

See Configure Splunk forwarding to use your own certificates for more information about configuring your forwarders and indexers.

Generate a private key for your server certificate

1. Create a new private key. The following example uses DES3 encryption and a 2048 bit key length. We recommend a key length of 2048 or higher.

   In *nix:
   ```bash
   $SPLUNK_HOME/bin/splunk cmd openssl genrsa -des3 -out myServerPrivateKey.key 2048
   ```
   In Windows:
   ```bash
   $SPLUNK_HOME\bin\splunk cmd openssl genrsa -des3 -out myServerPrivateKey.key 2048
   ```

   2. When prompted, create a password for your key.

   When you are done, a new private key myServerPrivateKey.key is created in your directory. You will use this key to sign your Certificate Signing Request (CSR).

Generate a new Certificate Signing Request (CSR)

1. Use your private key myServerPrivateKey.key to generate a CSR for your server certificate:
In *nix:

```
$SPLUNK_HOME/bin/splunk cmd openssl req -new -key myServerPrivateKey.key -out myServerCertificate.csr
```

In Windows:

```
$SPLUNK_HOME\bin\splunk cmd openssl req -new -key myServerPrivateKey.key -out myServerCertificate.csr
```

2. When prompted, provide the password you created for your private key myServerPrivateKey.key.

3. Provide the requested information for your certificate. To use common-name checking, make sure to provide a Common Name when entering your certificate details.

When you are done, a new CSR myServerCertificate.csr appears in your directory.

### Download and verify the server certificate and public key

1. Send your CSR to your Certificate Authority (CA) to request a new server certificate. The request process varies based on the Certificate Authority you use.

2. Download the new server certificate from your Certificate Authority. For the examples in this manual, let's call this myServerCertificate.pem.

3. Also download your Certificate Authority’s public CA certificate. For the examples in this manual, let's call this myCACertificate.pem.

If your Certificate Authority does not provide you with certificates in PEM format, you must convert them using the OpenSSL command appropriate to your existing file type, consult your OpenSSL documentation for more information about converting different file types.

4. View the contents to make sure it has everything you need:

   - The "Issuer" entry should refer to your CA's information.
   - The "Subject" entry should show the information (country name, organization name, Common Name, etc) that you entered when creating the CSR earlier.

**Note:** For *nix, you can view the contents your certificate using the following command:

```
$SPLUNK_HOME/bin/splunk cmd openssl x509 -in myServerCertificate.pem -text
```

### Next steps

You should now have the following files in the directory you created, which is everything you need to configure indexers, forwarders, and Splunk instances that communicate over the management port:

- myServerCertificate.pem
- myServerPrivateKey.key
- myCACertificate.pem

Now that you have the certificates you need, you must prepare your server certificate (including appending any intermediate certificates), and then configure Splunk software to find and use your certificates:

- See "How to prepare your signed certificates for Splunk" to learn how to set up your certificates to work with Splunk.
• See "Configure Splunk forwarding to use your own certificates" to learn more about configuring certificate authentication for forwarding.
• See "About securing inter-Splunk communication" to learn more about configuring certificate authentication for inter-Splunk communications.

Self-sign certificates for Splunk Web

This topic provides basic examples for creating the self-signed certificates in the command line using the version of OpenSSL included with Splunk software.

There are multiple ways you can create signed certificates, depending upon your organizations policies, your platform, and the tools that you are using. If you have already generated these certificates and key, or if you are experienced in generating certificates, you can skip this task and go directly to the configuration topic Secure Splunk Web with your own certificate in this manual.

Since self-signed certificates are signed by your organization, they are not contained in browser certificate stores. As a result, web browsers consider self-signed certificates "untrusted". This produces a warning page to users and may even prevent access for the user.

Self-signed certificates are best for browser to Splunk Web communication that happens within an organization or between known entities where you can add your own CA to all browser stores that will contact Splunk Web. For any other scenario, CA-signed certificates are recommended. See Get certificates signed by a third party for Splunk Web for more information.

Before you begin

In this discussion, $SPLUNK_HOME refers to the Splunk installation directory.

• For Windows, the default installation directory is C:\Program Files\splunk.
• For most *nix platforms, the default installation directory is /opt/splunk.
• For Mac OS, the default installation directory is /Applications/splunk.

See the Administration Guide to learn more about working with Windows and *nix.

Generate a new root certificate to be your Certificate Authority

1. Create a new directory to host your certificates and keys. For this example we will use $SPLUNK_HOME/etc/auth/mycerts.

2. Generate a new RSA private key. Splunk Web supports 2048 bit keys, but you can specify larger keys if they are supported by your browser.

   $SPLUNK_HOME/bin/splunk cmd openssl genrsa -aes256 -out myCAPrivateKey.key 2048
Note that in Windows you may need to append the location of the openssl.cnf file:

```bash
$SPLUNK_HOME\bin\splunk cmd openssl genrsa -aes256 -out myCAPrivateKey.key 2048
```

Splunk Web supports 2048 bit keys, but you can specify larger keys if they are supported by your browser.

3. **When prompted, create a password.**

The private key `myCAPrivateKey.key` appears in your directory. This is your root certificate private key.

4. **Generate a certificate signing request using the root certificate private key `myCAPrivateKey.key`:**

In *nix:

```bash
$SPLUNK_HOME/bin/splunk cmd openssl req -new -key myCAPrivateKey.key -out myCACertificate.csr
```

In Windows:

```bash
$SPLUNK_HOME\bin\splunk cmd openssl req -new -key myCAPrivateKey.key -out myCACertificate.csr
```

5. **Provide the password to the private key `myCAPrivateKey.key`.**

A new CSR `myCACertificate.csr` appears in your directory.

6. **Use the CSR to generate a new root certificate and sign it with your private key:**

In *nix:

```bash
$SPLUNK_HOME/bin/splunk cmd openssl x509 -req -in myCACertificate.csr
-signkey myCAPrivateKey.key -out myCACertificate.pem -days 3650
```

In Windows:

```bash
>SSLUNK_HOME\bin\splunk cmd openssl x509 -req -in myCACertificate.csr
-signkey myCAPrivateKey.key -out myCACertificate.pem -days 3650
```

7. **When prompted, provide for the password to the private key `myCAPrivateKey.key`.**

A new certificate `myCACertificate.pem` appears in your directory. This is your public certificate.

**Create a new private key for Splunk Web**

1. **Generate a new private key:**

In *nix:

```bash
$SPLUNK_HOME/bin/splunk cmd openssl genrsa -aes256 -out mySplunkWebPrivateKey.key 2048
```

In Windows:

```bash
$SPLUNK_HOME\bin\splunk cmd openssl genrsa -aes256 -out mySplunkWebPrivateKey.key 2048 -config
```

2. **When prompted, create a password.**

A new key, `mySplunkWebPrivateKey.key` appears in your directory.

3. **Remove the password from your key. (Splunk Web does not support password-protected private keys.)**
In *nix:

```bash
$SPLUNK_HOME/bin/splunk cmd openssl rsa -in mySplunkWebPrivateKey.key -out mySplunkWebPrivateKey.key
```

In Windows:

```bash
$SPLUNK_HOME\bin\splunk cmd openssl rsa -in mySplunkWebPrivateKey.key -out mySplunkWebPrivateKey.key
```

You can verify that your password was removed with the following command:

In *nix:

```bash
$SPLUNK_HOME/bin/splunk cmd openssl rsa -in mySplunkWebPrivateKey.key -text
```

In Windows:

```bash
$SPLUNK_HOME\bin\splunk cmd openssl rsa -in mySplunkWebPrivateKey.key -text
```

You should be able to read the contents of your certificate without providing a password.

**Create and sign a server certificate**

1. Create a new certificate signature request using your private key `mySplunkWebPrivateKey.key`:

In *nix:

```bash
$SPLUNK_HOME/bin/splunk cmd openssl req -new -key mySplunkWebPrivateKey.key -out mySplunkWebCert.csr
```

In Windows:

```bash
$SPLUNK_HOME\bin\splunk cmd openssl req -new -key mySplunkWebPrivateKey.key -out mySplunkWebCert.csr
```

The CSR `mySplunkWebCert.csr` appears in your directory.

2. Self-sign the CSR with the root certificate private key `myCAPrivateKey.key`:

In *nix:

```bash
$SPLUNK_HOME/bin/splunk cmd openssl x509 -req -in mySplunkWebCert.csr -CA myCACertificate.pem -CAkey myCAPrivateKey.key -CAcreateserial -out mySplunkWebCert.pem -days 1095
```

In Windows:

```bash
$SPLUNK_HOME\bin\splunk cmd openssl x509 -req -in mySplunkWebCert.csr -CA myCACertificate.pem -CAkey myCAPrivateKey.key -CAcreateserial -out mySplunkWebCert.pem -days 1095
```

3. When prompted, provide the password to the root certificate private key `myCAPrivateKey.key`.

The certificate `mySplunkWebCert.pem` is added to your directory. This is your server certificate.

**Create a single PEM file**

Combine your server certificate and public certificates, in that order, into a single PEM file.
Here’s an example of how to do this in Linux:

```bash
# cat mySplunkWebCert.pem myCACertificate.pem > mySplunkWebCertificate.pem
```

Here’s an example in Windows:

```plaintext
# type mySplunkWebCert.pem myCA.pem > mySplunkWebCertificate.pem
```

**Set up certificate chains**

To use multiple certificates, append the intermediate certificate to the end of the server's certificate file in the following order:

```
[ server certificate]
[ intermediate certificate]
[ root certificate (if required) ]
```

So for example, a certificate chain might look like this:

```
-----BEGIN CERTIFICATE-----
... (certificate for your server)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the intermediate certificate)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the root certificate for the CA)...
-----END CERTIFICATE-----
```

**Next steps**

Now that you have your certificates, you need to distribute them and configure Splunkd and Splunk Web to use them. See Secure Splunk Web with your own certificate in this manual for more information.

**Get certificates signed by a third-party for Splunk Web**

This topic provides basic examples for creating the third-party signed certificates necessary to configure Splunk Web for SSL authentication and encryption.

There are multiple ways you can create these certificates, depending upon your organization's policies, your network structure and the tools that you are using. If you have already generated these certificates and key, or if you are experienced with third-party certificates, you may prefer to skip this step and go directly to the configuration topic in this manual at Secure Splunk Web with your own certificate.

**Before you begin**

In this discussion, `$SPLUNK_HOME` refers to the Splunk installation directory. On Windows, Splunk software is installed at `C:\Program Files\splunk` by default. For most Unix platforms, the default installation directory is at `/opt/splunk`; for Mac OS, it is `/Applications/splunk`. See the Administration Guide to learn more about working with Windows and *nix.

**Create a new private key for Splunk Web**

1. Create a new directory to host your own certificates and keys. In this example we will use `$SPLUNK_HOME/etc/auth/mycerts`.

178
We recommend that you place your new certificates in a different directory than `$SPLUNK_HOME/etc/auth/splunkweb` so that you don't overwrite the existing certificates. This ensures that you can use the certificates that ship with Splunk for other Splunk components as necessary.

2. Generate a new private key. Splunk Web supports 2048-bit keys or larger.

   Linux:

   ```
   $SPLUNK_HOME/bin/splunk cmd openssl genrsa -des3 -out mySplunkWebPrivateKey.key 2048
   ```

   Windows:

   ```
   $SPLUNK_HOME\bin\splunk cmd openssl genrsa -des3 -out mySplunkWebPrivateKey.key 2048
   ```

3. Create a password when prompted to enter the passphrase for the original key.

   A new private key `mySplunkWebPrivateKey.key` is added to your directory. You can use this key to sign your CSR.

4. Remove the password from the private key. Splunk Web does not support private key passwords.

   Linux:

   ```
   $SPLUNK_HOME/bin/splunk cmd openssl rsa -in mySplunkWebPrivateKey.key -out mySplunkWebPrivateKey.key
   ```

   Windows:

   ```
   $SPLUNK_HOME\bin\splunk cmd openssl rsa -in mySplunkWebPrivateKey.key -out mySplunkWebPrivateKey.key
   ```

   You can use the following command to make sure that your password was successfully removed:

   ```
   # openssl rsa -in mySplunkWebPrivateKey.key -text
   ```

   If the password was successfully removed, you can view the certificate contents without providing a password.

**Create a Certificate Authority (CA) request and obtain your server certificate**

1. Create a new certificate signature request using your private key `mySplunkWebPrivateKey.key`:

   In *nix:

   ```
   $SPLUNK_HOME/bin/splunk cmd openssl req -new -key mySplunkWebPrivateKey.key -out mySplunkWebCert.csr
   ```

   In Windows:

   ```
   $SPLUNK_HOME\bin\splunk cmd openssl req -new -key mySplunkWebPrivateKey.key -out mySplunkWebCert.csr
   ```

   **Note for Windows platforms:** If you see an error similar to this:

   ```
   Unable to load config info from c:\build-amd64-5.0.2-20130120-1800\splunk/ssl/openssl.cnf
   ```

   Try typing the following in your command prompt then run the `openssl` command again:

   ```
   set OPENSSL_CONF=c:/Program Files/Splunk/openssl.cnf
   ```

2. Use the CSR `mySplunkWebCert.csr` to request a new signed certificate from your Certificate Authority (CA). The process for requesting a signed certificate varies depending on how your Certificate Authority handles a certificate signature request. Contact your CA for more information.
3. Download the server certificate returned by your Certificate Authority. For this example, let's call it "mySplunkWebCert.pem."

4. Download your Certificate Authority's public CA certificate. For this example, let's call it "myCAcert.pem."

5. Make sure that both the server certificate and the public CA certificate are both in PEM format. If the certificates are not in PEM format, convert them using the openssl command appropriate to your existing file type. Here's an example of a command that you can use for DER formats:

```bash
$SPLUNK_HOME/bin/splunk cmd openssl x509 -in mySplunkWebCert.crt -inform DER -out mySplunkWebCert.pem -outform PEM
$SPLUNK_HOME/bin/splunk cmd openssl x509 -in myCACert.crt -inform DER -out myCACert.pem -outform PEM
```

6. Check both certificates to make sure they have the necessary information and are not password protected.

```bash
$SPLUNK_HOME/bin/splunk cmd openssl x509 -in myCACert.pem -text
$SPLUNK_HOME/bin/splunk cmd openssl x509 -in mySplunkWebCert.pem -text
$SPLUNK_HOME/bin/splunk cmd openssl x509 -in myCACert.pem -text
$SPLUNK_HOME/bin/splunk cmd openssl x509 -in mySplunkWebCert.pem -text
```

The issuer information for mySplunkWebCert.pem should be the subject information for myCACert.pem (unless you are using intermediary certificates).

**Combine your certificate and keys into a single file**

Combine your server certificate and public certificate, in that order, into a single PEM file.

**Set up certificate chains**

To use multiple certificates, append the intermediate certificate to the end of the server's certificate file in the following order:

```
[ server certificate]
[ intermediate certificate]
[ root certificate (if required) ]
```

So for example, a certificate chain might look like this:

```
-----BEGIN CERTIFICATE-----
... (certificate for your server)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the intermediate certificate)...
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... (the root certificate for the CA)...
-----END CERTIFICATE-----
```

Note that the root CA that signed the intermediate certificate and all intermediary certificates must be in the browser certificate stores.

**Next steps**

Configure Splunk's `web.conf` file to find and use your certificates for authentication. See Secure Splunk Web with your own certificate for more information.