



Splunk® Enterprise Getting Data In 9.0.0

Generated: 7/01/2022 11:27 am

Table of Contents

Introduction.....	1
What data can I index?.....	1
Get started with getting data in.....	3
Is my data local or remote?.....	5
Use forwarders to get data into Splunk Cloud Platform.....	6
Use forwarders to get data into Splunk Enterprise.....	9
Use apps and add-ons to get data in.....	10
Other ways to get data in.....	11
How Splunk Enterprise handles your data.....	14
How to get data into your Splunk deployment.....	17
How do you want to add data?.....	17
Upload data.....	18
Monitor data.....	18
Forward data.....	19
Assign the correct source types to your data.....	20
Prepare your data for preview.....	23
Modify event processing.....	24
Modify input settings.....	27
Distribute source type configurations in Splunk Enterprise.....	28
Get data from files and directories.....	30
Monitor files and directories.....	30
Monitor files and directories in Splunk Enterprise with Splunk Web.....	32
Monitor Splunk Enterprise files and directories with the CLI.....	34
Monitor files and directories with inputs.conf.....	36
Specify input paths with wildcards.....	42
Include or exclude specific incoming data.....	46
How the Splunk platform handles log file rotation.....	48
Get data from network sources.....	50
Get data from TCP and UDP ports.....	50
How the Splunk platform handles syslog data over the UDP network protocol.....	56
Send SNMP events to your Splunk deployment.....	60
Get Windows data.....	62
Monitor Windows data with the Splunk platform.....	62
How to get Windows data into your Splunk deployment.....	63
Considerations for deciding how to monitor remote Windows data.....	65
Monitor Active Directory.....	69
Monitor Windows event log data with Splunk Enterprise.....	76
Monitor file system changes on Windows.....	95
Monitor data through Windows Management Instrumentation (WMI).....	99
Monitor Windows Registry data.....	107
Monitor Windows performance.....	112
Monitor Windows data with PowerShell scripts.....	126
Monitor Windows host information.....	130

Table of Contents

Get Windows data	
Monitor Windows printer information.....	134
Monitor Windows network information.....	138
Get data with HTTP Event Collector.....	144
Share HEC Data.....	144
Set up and use HTTP Event Collector in Splunk Web.....	145
Set up and use HTTP Event Collector with configuration files.....	152
Set up and use HTTP Event Collector from the CLI.....	155
Use cURL to manage HTTP Event Collector tokens, events, and services.....	157
About HTTP Event Collector Indexer Acknowledgment.....	161
Scale HTTP Event Collector with distributed deployments.....	166
Format events for HTTP Event Collector.....	168
Automate indexed field extractions with HTTP Event Collector.....	172
Send metrics to a metrics index.....	174
HTTP Event Collector REST API endpoints.....	174
HTTP Event Collector examples.....	175
Troubleshoot HTTP Event Collector.....	178
Get other kinds of data in.....	185
Monitor First In, First Out (FIFO) queues.....	185
Monitor changes to your file system.....	186
Get data from APIs and other remote data interfaces through scripted inputs.....	190
Get data with the Journald input.....	196
Configure event processing.....	201
Overview of event processing.....	201
Configure character set encoding.....	201
Configure event line breaking.....	205
Configure event timestamps.....	210
Configure indexed field extraction.....	210
Anonymize data.....	211
Configure timestamps.....	217
How timestamp assignment works.....	217
Configure timestamp recognition.....	219
Configure timestamp assignment for events with multiple timestamps.....	227
Configure advanced timestamp recognition with datetime.xml.....	228
Specify time zones for timestamps.....	234
Tune timestamp recognition for better indexing performance.....	237
Configure indexed field extraction.....	238
About indexed field extraction.....	238
About default fields (host, source, sourcetype, and more).....	238
Assign default fields dynamically.....	240
Create custom fields at index time.....	241
Extract fields from files with structured data.....	249

Table of Contents

Configure indexed field extraction	
Process events with ingest-time eval.....	258
Reduce lookup overhead with ingest-time lookups.....	260
Configure host values.....	263
About hosts.....	263
Set a default host for a Splunk platform instance.....	264
Set a default host for a file or directory input.....	265
Set host values based on event data.....	269
Change host values after indexing.....	272
Configure source types.....	274
Why source types matter.....	274
Override automatic source type assignment.....	277
Configure rule-based source type recognition.....	280
List of pretrained source types.....	281
Override source types on a per-event basis.....	286
Create source types.....	288
Manage source types.....	289
Rename source types at search time.....	293
Manage event segmentation.....	295
About event segmentation.....	295
Set the segmentation for event data.....	296
Set search-time event segmentation in Splunk Web.....	298
Improve the data input process.....	300
Use a test index to test your inputs.....	300
Use persistent queues to help prevent data loss.....	301
Use ingest actions to improve the data input process.....	303
Troubleshoot the input process.....	310
Resolve data quality issues.....	313

Introduction

What data can I index?

The Splunk platform can **index** any kind of data. In particular, the Splunk platform can index any and all IT streaming, machine, and historical data, such as Microsoft Windows event logs, web server logs, live application logs, network feeds, **metrics**, change monitoring, message queues, archive files, and so on.

Types of data sources in Splunk Cloud Platform

Splunk Cloud Platform provides tools to configure many kinds of data inputs, including those that are specific to particular application needs. Splunk Cloud Platform also provides the tools to configure any arbitrary data input types. In general, you can categorize Splunk Cloud Platform inputs as follows:

- Files and directories
- Network events
- Windows sources
- HTTP Event Collector (HEC)
- Metrics

Files and directories

A lot of data comes directly from files and directories. You can use **universal** and **heavy forwarders** to monitor those files and directories and send them to Splunk Cloud Platform. As a best practice, install universal forwarders on every machine where you want to monitor files and directories and send that data to a heavy forwarder which then sends the data to Splunk Cloud Platform. To monitor files and directories, see [Get data from files and directories](#).

Network events

You might want to collect data from network ports, such as network data from machines that run syslog. To do this in Splunk Cloud Platform, use a heavy or universal forwarder to collect the network data and then send that data to Splunk Cloud Platform. To get data from network ports, see [Get data from TCP and UDP ports](#).

Windows sources

To get data from Windows sources into Splunk Cloud Platform, install the Splunk Add-on for Windows on your universal forwarder. In this scenario, you can use a deployment server to deliver the Splunk Add-on for Windows to the Windows machines you want to monitor. The add-on collects the data and sends it to Splunk Cloud Platform.

For additional information on getting Windows data into Splunk Cloud Platform, see [Get Windows Data into Splunk Cloud Platform](#) in the Splunk Cloud Platform *Admin Manual*.

HTTP Event Collector

In Splunk Cloud Platform, you can use the HTTP Event Collector to get data directly from a source with the HTTP or HTTPS protocols. For more information, see [The HTTP Event Collector endpoint](#).

Metrics

You can also get metrics data from your technology infrastructure, security systems, and business applications. For more information, see [Metrics](#).

Types of data sources in Splunk Enterprise

Because Splunk Enterprise is on-premises, you can either get data into the instance directly or use universal or heavy forwarders to get data in. In general, you can categorize Splunk Enterprise inputs as follows:

- Files and directories
- Network events
- Windows data
- Other sources

Files and directories

You can use the files and directories **monitor** input processor to get data from files and directories. To monitor files and directories, see [Get data from files and directories](#).

Network events

You can index data from any network port, such as remote data from syslog-ng or any other application that transmits over the TCP protocol. It can also index UDP data, but use TCP whenever possible for enhanced reliability.

Splunk Enterprise can also receive and index SNMP events and alerts fired off by remote devices.

To get data from network ports, see [Get data from TCP and UDP ports](#) in this manual.

To get SNMP data, see [Send SNMP events to your Splunk deployment](#) in this manual.

Windows data

The Windows version of Splunk Enterprise accepts a wide range of Windows-specific inputs directly. With Splunk Web, you can configure the following Windows-specific input types:

- Windows Event Log data
- Windows Registry data
- Windows Management Instrumentation (WMI) data
- Active Directory data
- Performance monitoring data

To index and search Windows data on a non-Windows instance of Splunk Enterprise, you must first use a Windows instance to gather the data. See [Considerations for deciding how to monitor remote Windows data](#).

For a more detailed introduction to using Windows data in Splunk Enterprise, see [Monitoring Windows data](#) in this manual.

Other sources

Splunk Enterprise can collect the following data sources directly:

- You can use the HTTP Event Collector to get data directly from a source with the HTTP or HTTPS protocols. See [The HTTP Event Collector endpoint](#).
- You can also get metrics data from your technology infrastructure, security systems, and business applications. See [Metrics](#).
- You can monitor First In, First Out (FIFO) queues. See [Monitor First In, First Out \(FIFO\) queues](#).
- You can get data from APIs and other remote data interfaces and message queues. See [Scripted inputs](#).
- You can define a custom input capability to extend the Splunk Enterprise framework. See [Create custom data inputs for Splunk Cloud Platform or Splunk Enterprise on the Splunk Developer Portal](#).

Get started with getting data in

Now that you know what kind of data the Splunk platform can index, you can start getting data in to the Splunk platform. See [Get started with getting data in](#).

Get started with getting data in

To get started with getting data into your Splunk deployment, point your deployment at some data by configuring an input. You can get data in using several ways. For the most straightforward option, use Splunk Web. With a Splunk Cloud Platform deployment, you might need to configure a heavy forwarder or universal forwarder to send the data to your Splunk Cloud Platform instance.

Alternatively, you can download and enable an app, such as the Splunk App for Microsoft Exchange or Splunk IT Service Intelligence. See [Use apps and add-ons to get data in](#).

After you configure the inputs or enable an app, your Splunk deployment stores and processes the specified data. You can go to either the Search & Reporting app or the main app page and begin exploring the data that you collected.

Understand your needs

Before you start adding inputs to your deployment, ask yourself the following questions:

Question	Documentation
What kind of data do I want to index?	What data can I index?
Is there an app for that?	Use apps to get data in
Where does the data reside? Is it local or remote?	Where is my data?
Should I use forwarders to access remote data?	Use forwarders to get data in
What do I want to do with the indexed data?	What is Splunk knowledge?

Add new inputs

To add data, follow these high-level steps:

1. Create a test index and add a few inputs. Any data you add to your test index counts against your maximum daily indexing volume for licensing purposes.
2. Preview and modify how your data will be indexed before committing the data to the test index.
3. Review the test data that you added with the Search & Reporting app. Ask yourself these questions:
 - ◆ Do you see the sort of data you were expecting?
 - ◆ Did the default configurations work well for your events?
 - ◆ Is data missing or mixed up?
 - ◆ Are the results optimal?
4. If necessary, tweak your input and event processing configurations further until events look the way you want them to.
5. Delete the data from your test index and start over, if necessary.
6. When you are ready to index the data permanently, configure the inputs to use the default `main` index.

You can repeat this task to add other inputs as you familiarize yourself with getting data in.

Index custom data

The Splunk platform can index any time-series data, usually without additional configuration. If you have logs from a custom application or device, process it with the default configuration first. If you do not get the results you want, you can tweak things to make sure the software indexes your events correctly.

See [Overview of event processing](#) and How indexing works so that you can make decisions about how to make the Splunk platform work with your data.

Then, consider the following scenarios for collecting data:

- Are the events in your data more than one line? See [Configure event line breaking](#).
- Is your data in an unusual character set? See [Configure character set encoding](#).
- Is the Splunk platform unable to determine the timestamps correctly? See [How timestamp assignment works](#).

Further reading on configuring data inputs and getting data into the Splunk platform

Refer to the following table for some ways you can explore and further configure your data:

Task	Documentation
Configure an input	Other ways to get data in
Add data to your Splunk deployment	How do you want to add data?
Experiment with adding a test index	Use a test index to test your inputs
Add source types	Assign the correct source types to your data
Configure event processing	How Splunk Enterprise handles your data
Delete data from your Splunk deployment	Delete indexed data and start over
Configure your inputs with a default index	Configure your inputs to use the default index

Is my data local or remote?

If you use Splunk Cloud Platform or run Splunk Enterprise in the cloud, all data that you index is remote. If you use Splunk Enterprise on-premises, the answer to whether your data is local or remote depends on a number of things:

- The operating system on which your Splunk Enterprise instance resides.
- Where the data is physically.
- The types of data storage that are connected to the Splunk Enterprise instance.
- Whether or not you need to perform any authentication or other intermediate to access the data store that contains the data you want to index.

Local data

A local resource is a fixed resource that your Splunk Enterprise instance has direct access to. You are able to access a local resource, and whatever it contains, without having to attach, connect, or perform any other intermediate action, such as authenticating or mapping a network drive. If your data is on such a resource, the data is local.

Here are some examples of local data:

- Data on a hard disk or solid state drive installed in a desktop, laptop, or server host.
- Data on a resource that has been permanently mounted over a high-bandwidth physical connection that the machine can access at boot time.
- Data on a RAM disk.

Remote data

A remote resource is any resource that doesn't meet the definition of a local resource. Data that exists on such a resource is remote data.

Here are some examples of remote resources:

- Network drives on Windows hosts.
- Active Directory schemas.
- NFS or other network-based mounts on *nix hosts.
- Most cloud-based resources.

Remote data exceptions

There are some cases where resources might be considered remote, but they are actually local:

- A host has a volume that has been permanently mounted over a high-bandwidth physical connection, like USB or FireWire. Because the computer can mount the resource at boot time, Splunk Enterprise treats it as a local resource, even though the resource can theoretically be disconnected at a later time.
- A host has a resource that has been permanently mounted over a high-bandwidth network standard, like iSCSI, or to a Storage Area Network over fiber. Because the standard treats such volumes as local block devices, such a resource isn't considered local.

Use forwarders to get data into Splunk Cloud Platform

You can get data into Splunk Cloud Platform in a number of ways. The best way depends on the source of the data and what you want to do with that data. You use one or more instances of the following tools to get data into Splunk Cloud Platform:

- Forwarders.
A forwarder is a Splunk Enterprise instance that has been optimized to send data. You might use multiple forwarders to send data depending on the volume and location of your source data.
- Inputs Data Manager (IDM).
The IDM is a hosted solution specific to Splunk Cloud Platform for scripted and modular inputs. In a majority of cases, an IDM eliminates the need for customer-managed infrastructure. For more information, see *Work with Inputs Data Manager (IDM)* in the *Splunk Cloud Platform Admin Manual*.

Customers on the Splunk Cloud Platform Victoria Experience don't need to use an IDM. For more information, see *Determine your Splunk Cloud Platform Experience*.

Forwarder types for getting data into Splunk Cloud Platform

Usually, to get data from your customer site to Splunk Cloud Platform, you use a forwarder. Splunk forwarders send data from a datasource to your Splunk Cloud Platform deployment for indexing, which makes the data searchable. Forwarders are lightweight processes, so they can usually run on the machines where the data originates.

There are two types of forwarders that you can use to get data in:

- To forward data to Splunk Cloud Platform, you typically use the Splunk universal forwarder. A universal forwarder is a dedicated, streamlined version of Splunk Enterprise that contains only the essential components needed to send data. The universal forwarder is oftentimes the best tool for forwarding data to indexers. Its main limitation is that it forwards only unparsed data, with a few exceptions. To send event-based data to indexers, you must use either a heavy forwarder or IDM.
- A heavy forwarder is a full Splunk Enterprise instance with some features disabled to achieve a smaller footprint. For data sources that must be collected using programmatic means, for example APIs and database access, or if you need to route and filter data, use a heavy forwarder to avoid running these kinds of inputs in the search head tier. Since the heavy forwarder adds metadata to the messages, you might see as much as two to three times the network traffic when you use a heavy forwarder.

Splunk Cloud Platform and the forwarder credentials app

When you work with forwarders to send data to Splunk Cloud Platform, you must download an app that has the credentials specific to your Splunk Cloud Platform instance. You install the forwarder credentials app on your universal forwarder, heavy forwarder, or deployment server, and it lets you connect to Splunk Cloud Platform.

Use a deployment server to deliver configurations to multiple forwarders

If you have multiple forwarders, you might need to use a deployment server to manage them. A deployment server is a Splunk Enterprise instance that acts as a centralized configuration manager. It groups together and collectively manages any number of forwarders. Forwarder instances that are remotely configured by deployment servers are called deployment clients. The deployment server downloads content such as configuration files and apps, to deployment clients to simplify the process of downloading these files to multiple locations for each configuration change.

Work with an intermediate forwarding tier

An intermediate forwarder tier is a collection of heavy forwarders that act as an aggregation point for data sent from other forwarders in your network. Once the data arrives at the intermediate tier, it is forwarded to Splunk Cloud Platform. By sending data from other forwarders through the intermediate forwarder tier, you can limit the hosts that will communicate with Splunk Cloud Platform over the internet.

As your intermediate tier uses heavy forwarders, you can also route or drop data before it leaves your network for Splunk Cloud Platform. For more information about data parsing and routing, see [Use a heavy forwarder to get data into Splunk Cloud Platform](#).

When implementing an intermediate forwarder tier, try to maintain a 2:1 or greater ratio of intermediate forwarders to your Splunk Cloud Platform indexers. You can accomplish this by adding additional intermediate forwarder nodes, or by configuring your intermediate forwarder nodes to use multiple pipelines. For more information, see [Configure a forwarder to handle multiple pipeline sets in the *Forwarder Manual*](#).

Use a universal forwarder to get data into Splunk Cloud Platform

The universal forwarder is the best choice for a large set of data collection requirements from systems in your environment. It is a purpose-built data collection mechanism with minimal resource requirements. The universal forwarder is the default choice for collecting and forwarding log data.

By default, the universal forwarder can forward a maximum of 256 KB of data per second. As a best practice, do not exceed this limit. For more information, read [Possible throughput limits in the Splunk Enterprise Troubleshooting Manual](#).

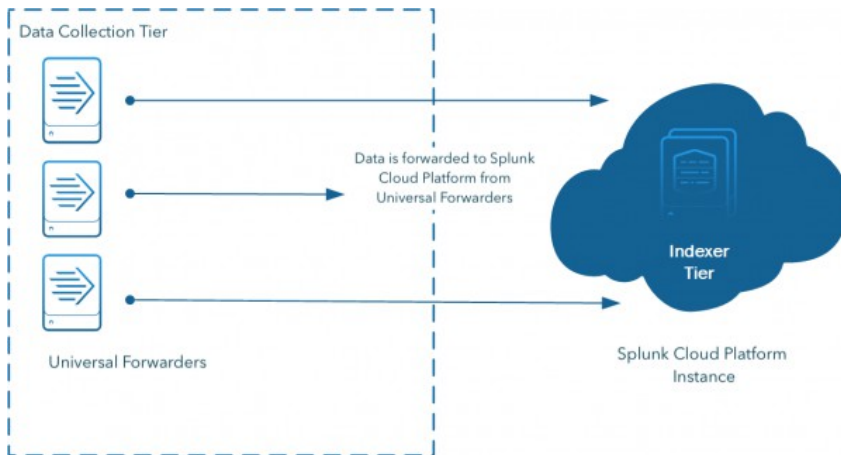
The universal forwarder provides the following functionality:

- Checkpoint and restart function for lossless data collection.
- Efficient protocol that minimizes network bandwidth utilization.
- Throttling capabilities.
- Built-in, load-balancing across available indexers.
- Optional network encryption using SSL or TLS.
- Data compression (use only without SSL or TLS).
- Multiple input methods (files, Windows Event logs, network inputs, scripted inputs).
- Limited event filtering capabilities (Windows event logs only).
- Parallel ingestion pipeline support to increase throughput and reduce latency.

The universal forwarder does not parse log sources into events, so it cannot perform any action that requires the ability to format the logs. It also ships with a stripped-down version of Python, which makes it incompatible with any modular input apps that require a full Splunk platform to function. It is normal for a large number of universal forwarders (from hundreds to tens of thousands) to be deployed on endpoints and servers in a Splunk platform environment and to be centrally managed, either with a Splunk deployment server, or a third-party configuration management tool, such as Puppet or Chef.

There are endpoints that do not allow installation of the universal forwarder, such as network devices, appliances, and logs using the syslog protocol. These are special considerations and are not covered in this document.

The following diagram shows how universal forwarders send data to Splunk Cloud Platform:



Use a heavy forwarder to get data into Splunk Cloud Platform

The heavy forwarder is a full Splunk Enterprise instance configured to act as a forwarder with indexing disabled. A heavy forwarder generally performs no other Splunk Enterprise functions (for example, do not use a heavy forwarder to search data). The main difference between a universal forwarder and a heavy forwarder is that the heavy forwarder contains the full parsing pipeline, performing the identical functions an indexer performs, without writing and indexing events on disk. This configuration enables the heavy forwarder to parse and act on individual events. For example, a heavy forwarder can mask data or perform filtering and routing based on event data. Because it is a full Splunk Enterprise installation, it can host modular inputs that require a full Python stack to function properly for data collection or serve as an endpoint for the Splunk HTTP event collector (HEC).

The heavy forwarder has the following characteristics:

- Parses data into events
- Filters and routes based on individual event data
- Has a larger resource footprint than the UF
- Has a larger network bandwidth footprint than the Universal Forwarder
- Has a GUI for management

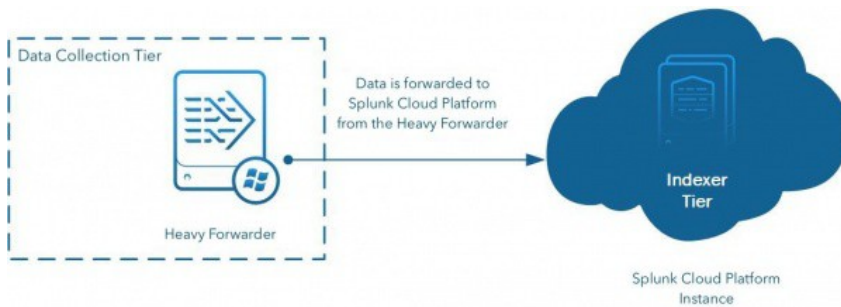
In general, heavy forwarders are not installed on endpoints for the purpose of data collection. Instead, they are used on standalone systems to implement data collection nodes (DCN) or intermediary forwarding tiers. Use a heavy forwarder only when requirements to collect data from other systems can't be met with a Universal Forwarder. Instances in which a heavy forwarder is required include the following examples:

- Reading data from a RDBMS for the purposes of ingesting it into Splunk Cloud Platform (database inputs).
- Collecting data from systems that are reachable using an API, such as cloud services, VMWare monitoring, and proprietary systems.
- Providing a dedicated tier to host the HTTP event collector service.
- Implementing an intermediary forwarding tier that requires a parsing forwarder for routing, filtering, and masking.

If you want to set up a heavy forwarder to send data in Splunk Cloud Platform, request a deployment server license from Splunk support to allow them to carry out functions above and beyond what is covered by the forwarder license. See Data collection in the *Splunk Cloud Platform Service Description*.

For more information about heavy forwarders, see the *Splunk Forwarding Data* manual.

The following example shows a heavy forwarder used to send data to Splunk Cloud Platform:



Use forwarders to get data into Splunk Enterprise

Splunk **forwarders** consume data and send it to an indexer. Forwarders require minimal resources and have little impact on performance, so they can usually reside on the machines where the data originates.

For example, if you have a number of Apache Web servers that generate data that you want to search centrally, you can set up forwarders on the Apache hosts. The forwarders take the Apache data and send it to your Splunk Enterprise deployment for indexing, which consolidates, stores, and makes the data available for searching. Because of their reduced resource footprint, forwarders have a minimal performance impact on the Apache servers.

Similarly, you can install forwarders on your employees' Windows desktops. These forwarders can send logs and other data to your Splunk Enterprise deployment, where you can view the data as a whole to track malware or other issues. The Splunk App for Windows Infrastructure relies on this kind of deployment.

What forwarders do

Forwarders get data from remote machines. Unlike raw network feeds, forwarders have the following capabilities:

- Tag metadata (source, sourcetype, and host)
- Buffer data
- Compress data
- Use SSL security
- Use any available network ports
- Run scripted inputs locally

Forwarders usually do not index the data, but instead, forward the data to a Splunk Enterprise deployment that does the indexing and searching. A Splunk Enterprise deployment can process data that comes from many forwarders. For detailed information on forwarders, see the *Forwarding Data* or *Universal Forwarder* manuals.

In most Splunk Enterprise deployments, forwarders serve as the primary consumers of data. In a large Splunk Enterprise deployment, you might have hundreds or even thousands of forwarders that consume data and forward for consolidation.

How to configure forwarder inputs

The following is a high-level overview of the steps to configure forwarder inputs for Splunk Enterprise.

1. Configure a Splunk Enterprise host to receive the data.
2. Determine the kind of forwarder you want to put on the host with the data.

- You can use a heavy forwarder, which is a full Splunk Enterprise instance with forwarding turned on, or a universal forwarder, which is its own installation package.
- The type of forwarder you use depends on the performance requirements for the host and whether you need to transform the data in any way as it comes into Splunk Enterprise.
- Download Splunk Enterprise or the universal forwarder for the platform and architecture of the host with the data.
- Install the forwarder onto the host.
- Enable forwarding on the host and specify a destination
- Configure inputs for the data that you want to collect from the host. You can use Splunk Web if the forwarder is a full Splunk Enterprise instance.
- Confirm that data from the forwarder arrives at the receiving indexer.

See the *Forwarding Data Manual* or the *Universal Forwarder Manual* for details on how to configure forwarding and receiving

Here are the main ways that you can configure data inputs on a forwarder:

- Specify inputs during the initial deployment of the forwarder.
- For Windows forwarders, specify common inputs during the forwarder installation process.
- For *nix forwarders, specify inputs directly after installation.
- Use the CLI.
- Edit the inputs.conf file.
- Install the app or add-on that contains the inputs you want.
- Use Splunk Web to configure the inputs and a deployment server to copy the resulting inputs.conf file to forwarders.

Forwarder topologies and deployments

- For information on forwarders, including use cases, typical topologies, and configurations, see About forwarding and receiving in the *Forwarding Data* manual.
- For details on how to deploy the universal forwarder, including how to use the **deployment server** to simplify distribution of configuration files and apps to multiple forwarders, see Install the forwarder credentials on many forwarders using a deployment server in the *Universal Forwarder* manual.

Use apps and add-ons to get data in

Splunk **apps** and **add-ons** extend the capability and simplify the process of getting data into your Splunk platform deployment. Download apps from Splunkbase.

Apps typically target specific data types and handle everything from configuring the inputs to generating useful views of the data. For example, the Splunk App for Windows Infrastructure provides data inputs, searches, reports, alerts, and dashboards for Windows host management. The Splunk App for Unix and Linux offers the same for Unix and Linux environments. There is a wide range of apps to handle specific types of application data, including the following apps and add-ons:

- Splunk DB Connect
- Splunk Stream
- Splunk Add-on for Amazon Web Services

Further reading for getting and installing apps

Go to Splunkbase to browse through the large set of apps available for download. Check Splunkbase frequently because new apps get added all the time.

See also

For more information about	See
Downloading and installing apps	Where to get more apps and add-ons in the <i>Splunk Enterprise Admin Manual</i>
Creating your own apps	<i>Developing Views and Apps for Splunk Web</i> manual
Using apps and add-ons with Splunk Cloud	Work with Apps and Add-ons in the <i>Splunk Cloud Platform Admin Manual</i>

Other ways to get data in

You can get data into your Splunk platform instance in a number of ways. The best way depends on the location and volume of data, your infrastructure and security needs, and what you intend to do with that data.

Assess your needs

Answer the following questions to help you determine the best way to get data into your Splunk platform instance.

Question	Considerations
What kind of data do I want to index?	The type of data you want to index affects how you get data in. For example, if you want to get data in from a proprietary application, you might want to use the HTTP Event Collector (HEC). On the other hand, if you want to ingest Windows data, you might want to use an app to help you get the data in. See What data can I index?
Is there an app for that?	Splunk and many third-party developers provide apps that facilitate and improve data ingestion. If there is an app for the type of data you want to get in, you can save yourself considerable time in configuring and tweaking inputs on universal forwarders. Use apps if they exist for the type of data you want to get in. See Use apps to get data in .
Where does the data reside?	For a Splunk Cloud Platform instance, data is always remote, which means that you have to use a universal forwarder or HEC to get the data indexed into Splunk Cloud Platform. For a Splunk Enterprise instance, data can be local or remote. See Is my data local or remote?
Do I need to use forwarders to access remote data?	If you have a Splunk Cloud Platform instance, you might have to. See Use forwarders to get data in to Splunk Cloud .
What do I want to do with the indexed data?	See What is Splunk knowledge? in the <i>Knowledge Manager Manual</i> .

Add your data

To add a new type of data to your Splunk platform instance, configure a data input. You can configure data inputs using the following methods:

- **Apps.** You can use a variety of **apps** that offer preconfigured inputs, views, and knowledge objects for various use cases. For more information, see [Use apps to get data in](#).
- **Splunk Web.** You can configure some inputs using **Splunk Web**. You can access the Add Data page from the Splunk Web home page. In addition, when you upload a file, you can preview and make adjustments to how Splunk Cloud Platform must index the file. See [Assign the correct source types to your data](#).

- **Forwarders.** If your data is remote, you can configure **forwarders** to send data from outlying machines to your Splunk Cloud Platform instance. For non-Splunk Cloud Platform installations, you can use these forwarders to send data to a central **indexer**. Depending on the operating system, you can specify some of the inputs at forwarder installation time. See [Use forwarders to get data in to Splunk Enterprise](#).

There are additional ways to get data in for Splunk Enterprise. See [Add your data to Splunk Enterprise](#).

Use apps to get data in

Splunk apps and **add-ons** extend the capability and simplify the process of getting data into your Splunk Cloud Platform deployment.

Apps typically target specific data types and handle everything from configuring data inputs to generating useful views of the data. For example, the Splunk App for Windows Infrastructure provides data inputs, searches, reports, alerts, and dashboards for Windows host management. The Splunk App for Unix and Linux offers the same for Unix and Linux environments. Most Splunk apps work with Splunk Cloud Platform directly, and others might require you install them on a universal or heavy forwarder to send the data to the Splunk Cloud Platform instance.

You can download these apps on Splunkbase:

- Splunk App for Windows Infrastructure
- Splunk App for Unix and Linux

You can also download apps to handle specific types of application data. Here are a few examples:

- Splunk DB Connect
- Splunk Stream
- Splunk Add-on for Amazon Web Services

Use Splunk Web

You can add data inputs from the Splunk Web home page or by selecting **Settings > Data Inputs**.

- From the Splunk Web home page, click **Add Data**.
- Select **Settings > Add data**.
- Select **Settings > Data inputs** from the **Data** section of the **Settings** drop-down list.

You can choose different options to get data in on the **Add Data** page. Click an icon to go to a page to define the data you want to upload, monitor, or forward. See these topics for more information:

- [Upload data](#)
- [Monitor data](#)

For more help on how to add data in Splunk Web, see [How do you want to add data?](#)

Add your data to Splunk Enterprise

With Splunk Enterprise, you can add data using Splunk Web or Splunk apps. In addition to these methods, you also can use the following methods.

- **The Splunk Command Line Interface (CLI).** This method is available for getting data in to Splunk Enterprise.

You can use the CLI to configure most types of inputs. You can also use it on a heavy forwarder to get data into Splunk Cloud Platform.

- **The `inputs.conf` configuration file.** When you specify your inputs with Splunk Web or the CLI, the details are saved in a **configuration file** on Splunk Enterprise indexer and heavy forwarder instances. While this option is not available on Splunk Cloud Platform, you can use a heavy forwarder to send data directly to your Splunk Cloud Platform instance. You can edit configuration files directly on both indexers and heavy forwarders, and some advanced data input needs might require you to make edits.

The Splunk Enterprise **Add Data** page has an additional option for getting data in:

- [Upload data](#)
- [Monitor data](#)
- [Forward data](#)

Use the CLI

On Splunk Enterprise and the universal forwarder, you can use the Splunk **CLI** to configure many inputs. From a shell or command prompt, navigate to the `$SPLUNK_HOME/bin/` directory and use the `./splunk` command. For example, the following command adds `/var/log/` as a data input:

```
splunk add monitor /var/log/
```

For more information on the CLI, including how to get command line help, see *About the CLI* in the *Admin Manual*.

Edit the `inputs.conf` configuration file

On Splunk Enterprise and the universal forwarder, you can edit the `inputs.conf` file to configure your inputs. You use a text editor to create or modify the file, where you can add a **stanza** for each input. You can add the stanza to the `inputs.conf` file in `$SPLUNK_HOME/etc/system/local/` or in your custom application directory in `$SPLUNK_HOME/etc/apps/<app name>/local/`.

You can configure the data input by adding key/value pairs to its stanza. You can set multiple settings in an input stanza. If you do not specify a value for a setting, Splunk Enterprise uses the default setting value. Default values for all settings in the `inputs.conf` file are in the `inputs.conf` configuration specification file. See the `inputs.conf` specification file in the *Admin Manual*.

If you have not worked with configuration files before, see *About configuration files* before adding inputs.

Example `inputs.conf` configuration file stanza

The following example configuration directs Splunk Enterprise to listen on TCP port 9995 for raw data from any remote host. Splunk Enterprise uses the DNS name of the remote host to set the host of the data. It assigns the source type `log4j` and the source `tcp:9995` to the data.

```
[tcp://:9995]
connection_host = dns
sourcetype = log4j
source = tcp:9995
```

For information on how to configure a specific input, see the topic for that specific input in this manual. For example, to configure file inputs, see [Monitor files and directories with `inputs.conf`](#).

The topic for each data input describes the main attributes available for that input. See the `inputs.conf` specification file in the *Admin Manual* for the complete list of available attributes, including descriptions of the attributes and several examples.

To get started with getting data into your Splunk deployment, point it at some data by configuring an input. There are several ways to do this. The most straightforward way is to use Splunk Web.

Alternatively, you can download and enable an app, such as the Splunk App for Microsoft Exchange or Splunk IT Service Intelligence. See [Use apps to get data in](#) for more information.

How app context determines where Splunk Enterprise writes configuration files

When you add an input through Splunk Web on Splunk Enterprise, the software adds that input to a copy of the `inputs.conf` configuration file. The application context, which is the Splunk app you are currently in when you configure the input, determines where Splunk Enterprise writes the `inputs.conf` file.

For example, if you navigate to the Settings page directly from the Search page and then add an input, Splunk Enterprise adds the input to `$SPLUNK_HOME/etc/apps/search/local/inputs.conf` because Splunk Enterprise is in the Search & Reporting app.

When you add inputs, confirm that you are in the app context that you want to be in. For background on how configuration files work, read About configuration files in the Splunk Enterprise *Admin Manual*.

See also

- Go to Splunkbase to install and download apps.
- For detailed information on what apps are, see Apps and add-ons in the *Admin Manual*. In particular, see Where to get more apps and add-ons to download and install an app.
- For information on how to create your own apps, see the *Developing Views and Apps for Splunk Web* manual.

How Splunk Enterprise handles your data

Splunk Enterprise consumes data and **indexes** it, transforming it into searchable knowledge in the form of **events**. The **data pipeline** shows the main processes that act on the data during indexing. These processes constitute **event processing**. After the data is processed into events, you can associate the events with **knowledge objects** to enhance their usefulness.

The data pipeline

Incoming data moves through the data pipeline. For more detailed information, see How data moves through Splunk deployments: The data pipeline in the *Distributed Deployment Manual*.

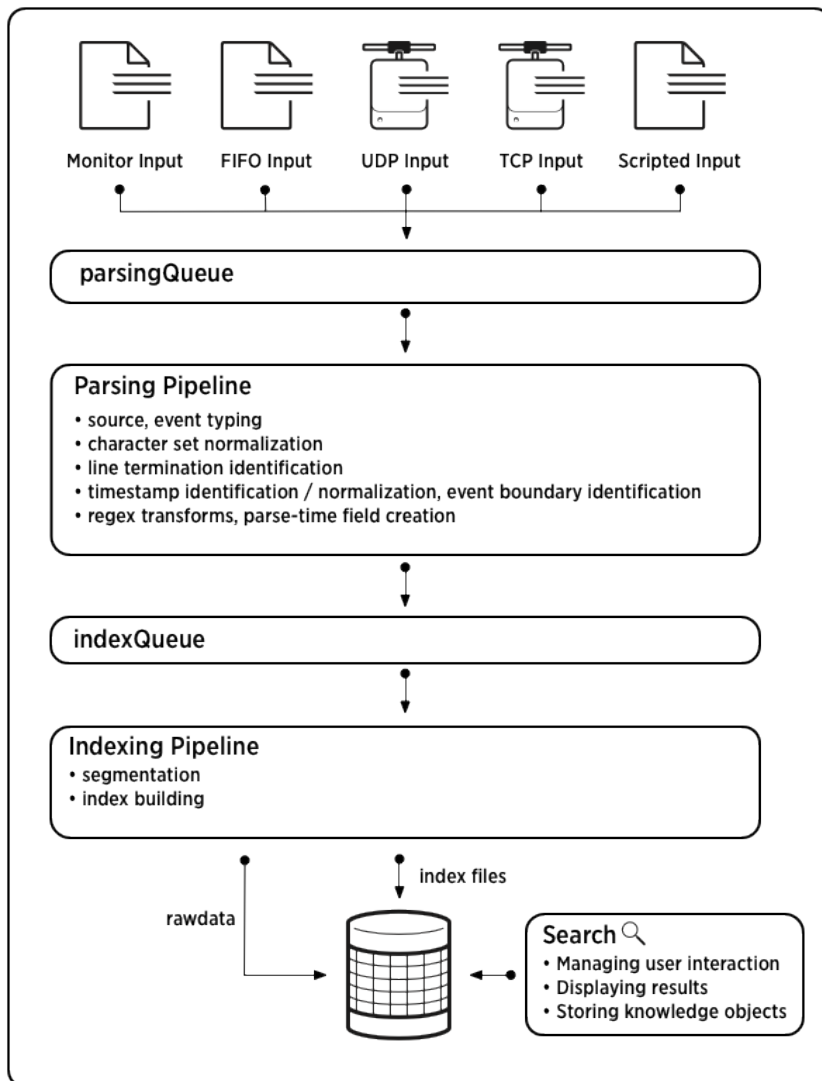
Each Splunk Enterprise processing **component** resides on one of the three typical processing tiers: the data input tier, the indexing tier, and the search management tier. Together, the tiers support the processes occurring in the data pipeline.

As data moves along the data pipeline, Splunk Enterprise components transform the data from its origin in external sources, such as log files and network feeds, into searchable events that encapsulate valuable knowledge.

The data pipeline has these segments:

- **Input**
- **Parsing**
- **Indexing**
- **Search**

This diagram shows the main steps in the data pipeline. In the data input tier, Splunk Enterprise consumes data from various inputs. Then, in the indexing tier, Splunk Enterprise examines, analyzes, and transforms the data. Splunk Enterprise then takes the parsed events and writes them to the index on disk. Finally, the search management tier manages all aspects of how the user accesses, views, and uses the indexed data.



Event processing

Event processing occurs in two stages, parsing and indexing. All data enters through the parsing pipeline as large chunks. During parsing, the Splunk platform breaks these chunks into events. It then hands off the events to the indexing pipeline, where final processing occurs.

During both parsing and indexing, the Splunk platform transforms the data. You can configure most of these processes to adapt them to your needs.

In the parsing pipeline, the Splunk platform performs a number of actions. The following table shows some examples in addition to related information:

Action	Related information
Extracting a set of default fields for each event, including <code>host</code> , <code>source</code> , and <code>sourcetype</code> .	About default fields
Configuring character set encoding.	Configure character set encoding
Identifying line termination using line breaking rules. You can also modify line termination settings interactively, using the Set Source Type page in Splunk Web.	Configure event line breaking Assign the correct source types to your data
Identifying or creating timestamps. At the same time that it processes timestamps, Splunk software identifies event boundaries. You can modify timestamp settings interactively, using the Set Source Type page in Splunk Web.	How timestamp assignment works Assign the correct source types to your data
Anonymizing data, based on your configuration. You can mask sensitive data (such as credit card or social security numbers) at this stage.	Anonymize data
Applying custom metadata to incoming events, based on your configuration.	Assign default fields dynamically

In the indexing pipeline, the Splunk platform performs additional processing. For example:

- Breaking all events into segments that can then be searched. You can determine the level of segmentation, which affects indexing and searching speed, search capability, and efficiency of disk compression. See [About event segmentation](#).
- Building the index data structures.
- Writing the raw data and index files to disk, where post-indexing compression occurs.

The distinction between parsing and indexing pipelines matters mainly for forwarders. Heavy forwarders can parse data locally and then forward the parsed data on to receiving indexers, where the final indexing occurs. Universal forwarders offer minimal parsing in specific cases such as handling structured data files. Additional parsing occurs on the receiving Splunk Enterprise indexer.

For information about events and what happens to them during the indexing process, see [Overview of event processing](#).

Enhance and refine events with knowledge objects

After the data has been transformed into events, you can make the events more useful by associating them with knowledge objects, such as event types, field extractions, and reports. For information about managing Splunk software knowledge, see the *Knowledge Manager Manual*, starting with What is Splunk knowledge?.

How to get data into your Splunk deployment

How do you want to add data?

The fastest way to add data to your Splunk Cloud Platform instance or Splunk Enterprise deployment is to use Splunk Web.

The Add Data page

To add data to the Splunk platform, access the Add Data page in Splunk Web by following these steps:

1. Log into Splunk Web, the Home page appears.
2. Click **Add Data** to access the Add Data page.
The Add Data page does not appear if your search head is part of a **search head cluster**. See About search head clustering in the Splunk Enterprise *Distributed Search* manual for more information.
3. After you access the Add Data page, choose one of three options for getting data into your Splunk platform deployment with Splunk Web:
 - ◆ Upload
 - ◆ Monitor
 - ◆ Forward

Upload

The Upload option lets you upload a file or archive of files for indexing. When you choose Upload option, Splunk Web opens the upload process page. For more details, see [Upload data](#).

Monitor

For Splunk Cloud Platform deployments, you can monitor files and directories with the HTTP Event Collector. For Splunk Enterprise installations, the Monitor option lets you monitor one or more files, directories, network streams, scripts, Event Logs (on Windows hosts only), performance metrics, or any other type of machine data that the Splunk Enterprise instance has access to. When you choose the Monitor option, Splunk Web loads a page that starts the monitoring process. See [Monitor data](#).

Forward

If you have a Splunk Cloud Platform environment, using a forwarder is the most common method for getting data in. The Forward option lets you receive data from **forwarders** into your Splunk Cloud Platform deployment. When you choose the Forward option, Splunk Web takes you to a page that starts the data collection process from forwarders. See [Forward data](#).

The Forward option requires additional configuration. Use this option only in a single-instance Splunk Cloud Platform environment.

Guided Data Onboarding

The Guided Data Onboarding (GDO) feature also provides end-to-end guidance for getting select data sources into specific Splunk platform deployments.

From the home page in Splunk Web, find the data onboarding guides by clicking **Add Data**. From there you can select a data source and configuration type. Then view diagrams, high-level steps, and documentation links that help you set up and configure your data source.

You can find all of the Guided Data Onboarding manuals by clicking the **Add data** tab on the Splunk Enterprise Documentation site.

Upload data

The Upload page lets you upload a file to your Splunk Enterprise instance from your computer. You can upload files by clicking the "Upload" button from Splunk Home in Splunk Web.

Caveats for Windows Event Log files and the upload feature

While you can upload any file to Splunk Enterprise or Splunk Cloud Platform, Windows Event Log (.evt) and Windows Event Log XML (.evtx) files that you exported from another Windows machine don't work with the upload feature because those files contain information that is specific to the Windows machine that generated them. Other machines can't process the files in their unaltered form. See [Index exported event log files](#) for more information about the constraints for working with these kinds of files.

The Upload page

1. Upload data through one of the following methods on this page:

- Drag the file you want to index from your desktop to the **Drop your data file here** area.
- Click **Select File** and select the file that you want to index.
- Splunk Enterprise then loads the file and processes it, depending on what type of file it is.
- After the file has completed loading, click **Next**.

Next step

[Set sourcetype](#)

Monitor data

If you want to monitor files and network ports on your Splunk Enterprise instance, navigate to the Monitor page in Splunk Web:

1. From the Splunk Web system bar, click **Settings**.
2. Choose the **Add Data** page.
3. Click the **Monitor** page.

The Monitor page

From the Monitor page, choose the type of data that you want Splunk Enterprise to monitor. Default inputs appear first, followed by forwarded inputs, then any modular inputs that are on the instance.

The Monitor page shows only the types of data sources that you can monitor, which depends on the type of Splunk deployment you have. If you're running Splunk Enterprise, the page also shows you the platform that the instance runs on. See [Types of data sources](#) for more information on what Splunk Enterprise can monitor.

Add a data input

Some data sources are available only on certain operating systems. For example, Windows data sources are available only on machines that run Windows. Splunk Cloud Platform cannot monitor Windows inputs directly because it doesn't run on Windows, but you can forward data from a universal forwarder that runs Windows to Splunk Cloud Platform.

1. Select a source by clicking it once. The page updates based on the source you selected.
2. Follow the on-screen prompts to complete the selection of the source object that you want to monitor.
3. Click **Next** to proceed to the next step in the Add data process.

If you experience problems with these steps, the logged-in Splunk user account might not have permissions to add data or see the data source that you want to add.

Next step

[Assign the correct source type to your data](#)

Forward data

The Forward data page lets you select and control forwarders that have connected to the Splunk platform instance. You can configure the forwarders so that they send data to the instance.

This page appears when you click the **Forward** button on the Add data page. It appears in the following cases:

- You use a single instance of Splunk Enterprise that acts as an indexer and a deployment server.
- You use a Free Trial Splunk Cloud Platform deployment and have configured a universal forwarder to connect to the Splunk Cloud Platform instance as a deployment client.

If one of these scenarios fits your situation, then you can manage available forwarders by following the "Use the Select Forwarders page to define and populate server classes with forwarders" procedure, described later in this topic.

If you have multiple machines in your Splunk deployment that perform indexing, then this page isn't useful. Instead, see [About deployment server and forwarder management in *Updating Splunk Enterprise Instances*](#) to learn about the deployment server and how to use it to manage forwarder configurations to send to multiple indexers.

If you have a Splunk Cloud Platform deployment, then this page isn't available. Instead, you can install an on-premises deployment server to synchronize forwarder configurations so that you don't have to manage forwarders manually.

Prerequisites for using the Forward data page

To use the Forward Data page to configure data inputs, you must configure at least one forwarder as a deployment client. If you haven't configured a forwarder as a deployment client, the page notifies you that no deployment clients have been found.

To configure a heavy forwarder as a deployment client, see [Configure deployment clients in the *Updating Splunk Enterprise Instances* manual](#).

To configure a universal forwarder as a deployment client, see *Deploy the universal forwarder* in the Splunk *Universal Forwarder* manual. On Windows machines, you can configure the forwarder as a deployment client during installation. See *Install a Windows universal forwarder from an installer* in the Splunk *Universal Forwarder* manual.

Use the Select Forwarders page to define and populate server classes with forwarders

When you select **Forward Data** from the Add Data page, the Select Forwarders page appears.

You can define **server classes** and add forwarders to those classes. Server classes are logical groupings of hosts based on things such as architecture or host name.

This page displays forwarders that you configured to forward data and act as deployment clients to this instance. If you haven't configured any forwarders, the page advises you of this.

The following procedure lets you set up server classes for forwarders that have reported themselves as deployment clients to this Splunk platform instance.

1. In **Select Server Class**, click one of the options.
 - ◆ Click **New** to create a new server class, or if an existing server class doesn't match the group of forwarders for which you want to configure a data input.
 - ◆ Click **Existing** to use an existing server class.
2. In the **Available host(s)** pane, choose the forwarders that you want this instance to receive data from. The forwarders move from the **Available host(s)** pane to the **Selected host(s)** pane.

A server class must contain hosts of a certain platform. You cannot, for example, put Windows and *nix hosts in the same server class.
3. (Optional) You can add all of the hosts by clicking the **add all** link, or remove all hosts by selecting the **remove all** link.
4. If you chose **New** in Select server class, enter a unique name for the server class. Otherwise, select the server class you want from the drop-down list.
5. Click **Next**. The Select Source page shows source types that are valid for the forwarders that you selected.
6. Select the data sources that you want the forwarders to send data to this instance.
7. Click **Next**.

Next step

[Modify input settings](#)

Assign the correct source types to your data

The source type is one of the **default fields** that the Splunk platform assigns to all incoming data, and determines how the Splunk platform formats the data during indexing. By assigning the correct source type to your data, the indexed version of the data appears the way you want it to with correct **timestamps** and **event** breaks.

You can confirm that the Splunk platform indexes your data as you want it to appear using the Set Source Type page in Splunk Web.

Assigning source types to your data

Splunk Enterprise comes with many predefined source types and attempts to assign the correct source type to your data based on its format. In some cases, you might need to manually select a different predefined source type to the data. In other cases, you might need to create a new source type with customized event processing settings.

On the Set Source Type page, you can see how Splunk Enterprise will index the data based on the application of a predefined source type. You can modify the settings interactively and save those modifications as a new source type.

Ensure that you're assigning the right source type to your data by following these steps on the Set Source Type page:

1. See what your data will look like without any changes using the default event-processing configuration.
2. Apply a different source type to see whether it offers more preferable results.
3. Modify settings for timestamps and event breaks to improve the quality of the indexed data and save the modifications as a new source type.
4. Create a new source type.

If you use Splunk Enterprise, you can save any new source types to a props.conf configuration file that you can later distribute across the indexers in your deployment so that the source types are available globally. See [Distribute source type configurations in Splunk Enterprise](#).

Some source types, such as those in the Log to Metrics category, cannot be previewed. See "About the Log to Metrics source type category" later in this topic for details.

For information on source types and why they are so important, see [Why source types matter](#).

About the Log to Metrics source type category

Source types in the Log to Metrics category are special source types. The Splunk platform uses these source types for the ingest-time conversion of log events to metric data points. If you select a source type from this category, a set of **Metrics** controls will appear on the left side of the Set Source Type page. For more information about log-to-metrics conversion and the **Metrics** settings, see Set up ingest-time log-to-metrics conversion in Splunk Web in the *Metrics* manual.

When you apply a Log to Metrics source type to an input, you can't preview the data for that input.

Assign source types to your data

When the Set Source Type page opens, Splunk Enterprise chooses a source type based on the data you specified. You can accept that source type or change it by following these steps.

1. Check the preview pane to see how Splunk Enterprise will index the data. Review event breaks and timestamps.
2. (Optional) View the event summary by clicking **View event summary**.
Splunk Web displays the event summary in a new window. See [View event summary](#).
3. If the data appears the way that you want, then click **Next** to proceed to the **Inputs Settings** page. Otherwise, choose from one of the following options:
 - ◆ Choose an existing source type to change the data formatting. See [Choose an existing source type](#).
 - ◆ Adjust timestamps, delimiters, and line breaking manually, then save the changes as a new source type. See [Adjust timestamps and event line breaks](#).

Choose an existing source type

If the data does not appear in the way that you want, see whether or not an existing source type fixes the problem.

If Splunk Enterprise can detect a source type, it displays the source type in the Source type: <sourcetype> drop-down list. If it can't determine a source type, it displays Sourcetype: System Defaults.

1. Click the **Source type: <sourcetype>** drop-down list to see a list of source type categories. Each category contains a list of source types within that category.
2. Hover over the category that best represents your data.
As you do, the source types under that category appear in a drop-down list.
3. Select the source type that best represents your data.
Splunk Web updates the data preview pane to show how the data looks under the new source type. You might need to scroll to see all source types in a category.
4. Review your data again in the preview pane. If the existing source types don't work for your data, you must manually adjust the timestamps, delimiters, and event breaking. See [Adjust timestamps and event line breaks](#).
5. If you're satisfied with the results, click **Next** to proceed to the **Inputs Settings** page.

View event summary

You can see a summary of the events within the data sample by clicking View Event Summary. This summary shows the following information:

- The size of the sample data, in bytes.
- The number of events that were present in the sample.
- The chart that represents the distribution of the events over time. the Splunk platform uses date stamps within the file to determine how to display this chart.
- A breakdown of the number of lines each event in the sample took up.

Adjust timestamps and event breaks

If you choose an existing source type without success, then you can manually adjust how Splunk Enterprise processes timestamps and event line breaks for the incoming data.

To manually adjust timestamp and event line breaking parameters, use the **Event Breaks**, **Timestamp**, **Delimited Settings**, and **Advanced** drop-down lists on the left pane of the Set Source Type page. The preview pane updates as you make changes to the settings.

The Event breaks tab appears only when Splunk Enterprise can't determine how to line-break the file, or if you select a source type that doesn't define line breaking. The Delimited settings tab appears only when Splunk Enterprise detects that you want to import a structured data file, or you select a source type for structured data such as CSV.

If you need more information about how to adjust timestamps and event breaks, see [Modify event processing](#).

To manually adjust timestamps and event breaks, follow these steps:

1. Click **Event breaks**. The list displays the **Break type** options, which controls how the Splunk platform line-breaks the file into events. You can choose from the following options:
 - ◆ **Auto**: Detect event breaks based on the location of the time stamp.
 - ◆ **By Line**: Breaks every line into a single event.

- ◆ **Regexâ** |: Uses the specified regular expression to determine line breaking.
2. Click **Timestamps**. The list expands to show extraction options. Select from one of the following options:
 - ◆ **Auto**: Extract timestamps automatically by looking in the file for timestamp events.
 - ◆ **Current time**: Apply the current time to all events detected.
 - ◆ **Advanced**: Specify the time zone, timestamp format in `strptime()`, and any fields that comprise the timestamp.
 3. Click **Delimited settings** to display delimiting options.

Field	Description
Field delimiter	The delimiting character for structured data files, such as comma-separated value (CSV) files.
Quote character	The character that Splunk Enterprise uses to determine when something is in quotes.
File preamble	A regular expression that tells Splunk Enterprise to ignore one or more preamble lines, or lines that don't contain any actual data, in the structured data file.
Field names	Determines field names automatically, based on line number, based on a comma-separated list, or through a regular expression.

4. After the results look the way you want, save your changes as a new source type, which you can then apply to the data when it is indexed.
5. If you want to make configuration changes to `props.conf`, click the **Advanced** tab to display fields that let you enter attribute/value pairs that get committed directly to the `props.conf` configuration file. See [Make configuration changes in the Advanced tab](#) for more instructions.

Making configuration changes in the Advanced tab requires advanced knowledge of Splunk software features. Changes made here might negatively affect the indexing of your data. Consider consulting Splunk Professional Services before configuring these options.

6. If you're satisfied with the results, click **Next** to proceed to the **Inputs Settings** page.

Make configuration changes in the Advanced tab

Making configuration changes in the Advanced tab requires advanced knowledge of Splunk software features. Changes made here might negatively affect the indexing of your data. Consider consulting Splunk Professional Services before configuring these options.

1. Click a field to edit `props.conf` entries that Splunk Enterprise generates based on your previous choices.
2. Click the **X** to the right of an attribute/value field pair to delete that pair.
3. Click **New setting** to create a new attribute/value field pair and specify a valid attribute and value for `props.conf`.
4. Click **Apply settings** to commit the changes to the `props.conf` file.

Next step

Once you assign the correct source types to your data, see [Modify input settings](#).

Prepare your data for preview

The Set Source Type page works on single files only, and it accesses files that reside on the Splunk deployment or have been uploaded there. Although the Set Source Type page doesn't directly process network data or directories of files, you can work around those limitations. With Splunk Cloud Platform, you can upload any file to preview it.

Preview network data

You can direct some sample network data into a file, which you can then either upload or add as a file monitoring input. Several external tools can do this. On *nix, the most popular tool is Netcat.

For example, if you want to monitor a network device for network traffic on UDP port 514, you can use Netcat to direct some of that network data into a file. Run a command such as this one:

```
nc -lu 514 > sample_network_data
```

For best results, run the command inside a shell script that has logic to terminate the Netcat process after the file reaches 2 MB in size. By default, Splunk software reads only the first 2 MB of data from a file when you preview the data within that file.

After you've created the `sample_network_data` file, you can add it as an input, preview the data, and assign any new source types to the file.

Preview directories of files

If all of the files in a directory are similar in content, you can preview a single file and be confident that the results are valid for all of the files in the directory. However, if you have directories with files of heterogeneous data, be sure to preview a set of files that represents the full range of data in the directory. Preview each type of file separately, because specifying a wildcard causes Splunk Web to disable the Set Source Type page.)

File size limit

Splunk Web displays the first 2 MB of data from a file in the Set Source Type page. In most cases, this amount provides a sufficient sampling of your data. If you use Splunk Enterprise, you can sample a larger quantity of data by changing the `max_preview_bytes` attribute in the `limits.conf` file. For more information about the `limits.conf` file, see `limits.conf` in the Splunk Enterprise *Admin Manual*.

Alternatively, you can edit the file to reduce large amounts of similar data so that the remaining 2 MB of data contains a representation of all the types of data in the original file.

Modify event processing

You can change the event processing settings and save the improved settings as a new source type.

1. View the event data, as described in [Assign the correct source types to your data](#).
2. Modify the event processing settings.
3. Review the effect of your changes until you are satisfied.
4. Save the modified settings as a new source type.
5. Apply the new source type to any of your inputs.

Modify the event processing settings

To create the new source type, use the event-breaking and timestamp settings, then save the source type.

On the Set Source Type page, you can make three types of adjustments using the following collapsible tabs:

- **Event Breaks.** Adjust the way that Splunk Enterprise breaks the data into events.
- **Timestamps.** Adjust the way Splunk Enterprise determines event timestamps.
- **Advanced tab.** If you have Splunk Enterprise, edit props.conf.

Modify event breaks

To modify event break parameters, click **Event Breaks**. You can choose the following break types:

- **Auto.** Break events based on the location of timestamps in the data.
- **Every line.** Consider every line a single event.
- **Regex...** Use the specified regular expression to break data into events.

For information on line breaking, see [Configure event line breaking](#). You can test your regular expression by using it in a search with the rex search command.

Modify timestamps

To modify timestamp recognition parameters, click the **Timestamps** tab to expand it.

You can choose from these extraction options:

- **Auto.** Locate the timestamp automatically.
- **Current Time.** Uses the current system time.
- **Advanced.** Specify additional advanced parameters to adjust the timestamp.

Then, you can configure additional advanced parameters:

- **Timezone.** The time zone that you want to use for the events.
- **Timestamp format.** A string that represents the timestamp format for Splunk Enterprise to use when searching for timestamps in the data. See [Configure timestamp recognition](#).
- **Timestamp prefix.** A regular expression that represents the characters that appear before a timestamp.
- **Lookahead.** The number of characters that Splunk Enterprise looks either into the event, or for the regular expression that you specified in "Timestamp prefix" for the timestamp.

If you specify a timestamp format in the Timestamp format field and the timestamp is not located at the very start of each event, you must also specify a prefix in the Timestamp prefix field. Otherwise, the Splunk platform can't process the formatting instructions, and every event will contain a warning about the inability to use strtptime. It's possible that you still end up with a valid timestamp, based on how the Splunk platform attempts to recover from the problem.

For information on configuring timestamps, see [How timestamp assignment works](#).

Make advanced modifications

To modify advanced parameters, click the **Advanced** tab. The tab shows options that let you specify source type properties by editing the underlying props.conf file.

You can add or change source type properties by specifying setting/value pairs. See the props.conf configuration file in the *Admin Manual* for details on how to set these properties.

The Advanced tab shows the current complete set of properties for the selected source type:

- Settings generated by changes made in the **Event Breaks** or **Timestamps** tabs after you click **Apply**.
- Preexisting settings for a source type that was either auto-detected or manually selected when you first previewed the file.
- Settings you apply from the **Additional settings** text box after you click **Apply settings**.

For information on how to set source type properties, see props.conf in the *Admin Manual*. See also [How timestamp assignment works](#) and [Configure event line breaking](#).

How Splunk Enterprise combines settings

The settings changes you make in **Advanced tab** take precedence. For example, if you alter a timestamp setting using the **Timestamps** tab and also make a conflicting timestamp change in **Advanced tab**, the **Advanced tab** change takes precedence over the modification that you made in the Timestamps tab.

Starting with highest precedence, the following list shows how Splunk Enterprise combines any adjustments with the underlying default settings:

1. Advanced tab changes
2. Event breaks or timestamp changes
3. Settings for the underlying source type, if any
4. Default system settings for all source types

If you make changes in the Advanced tab and then return to the Event Breaks or Timestamps tabs, the changes are not visible from those tabs.

Review your changes

When you are ready to view the effect of your changes, click **Apply settings**. Splunk Web refreshes the screen, so you can review the effect of your changes on the data.

To make further changes using any of the three adjustment methods available, click **Apply changes** to view the effect of the changes on your data.

Save modifications as a new source type

1. Click **Save As** next to the Sourcetype button.
2. In the dialog box that appears, name your new source type, choose the Source type category in which it will appear, and the application context it uses.

Field	Description
Name	The name of the new source type.
Description	The description of the new source type.
Category	The category that the source type appears as when you click Sourcetype.
App	The app that the new source type uses.

3. Click **Save** to save the source type and return to the Set Source Type page.

Next step

You have several options after you save the source type:

- (Optional) Click **Next** to apply the source type to your data and proceed to the **Input settings** page.
- (Optional) Click the left-pointing angle bracket (<) to go back and choose a new file to upload or monitor.
- (Optional) Click **Add data** to return to the beginning of the Add Data wizard.

Modify input settings

Prerequisites

- [Apply the correct source type to your data](#)

After you select the source or set your source type when uploading or monitoring a single file, the **Modify input settings** page appears in Splunk Enterprise.

You can specify additional parameters for your data input, such as its source type, application context, host value, and the index where data from the input is to be stored.

Configure source type

You can specify the source type to be applied to your data with the Source type setting. This setting appears in these situations:

- When you specify a directory as a data source.
- When you specify a network input as a data source.
- When you specify a data source that has been forwarded from another Splunk instance.

If your data source doesn't meet these criteria, then you won't see the Source type setting.

Specify a source type

To specify a source type, select one of these options:

Option	Description
Select	Click this button to apply the source type that you specify to the data.
New	Click this button to add a new source type.

Choose an existing source type

1. From the **Select Source Type** drop-down list, choose the category that best represents the data's source type.
2. Choose the source type from the list that appears.

Add a new source type

1. In the **Source Type** text field, enter the name of the new source type.
2. Choose a category for the source type in the **Source Type Category** drop-down list.
3. In the **Source Type Description** text field, enter the description for the source type.

Configure host value

Splunk Enterprise tags events with a host. The default host value is the hostname or IP address of the indexer or forwarder that initially ingests the data. However, you can configure how the software determines the host value. Configure a host value by choosing one of these available host values:

Host value	Description
IP	This value uses the IP address of the host from which the event originates.
DNS	This value uses Domain Name Services (DNS). Events are tagged with the host name that Splunk software determines using DNS name resolution.
Custom	This value uses the host value you assign in the "Host field value" text field that appears when you select this option.

Store an event in an index

The Index setting determines the index where the events for this input are to be stored.

1. To use the default index, leave the drop-down list option set to `Default`. Otherwise, click the drop-down list and select the index you want the data to go to.
2. (Optional) If the index you want to send the data to isn't in the list and you have permissions to create indexes, you can create a new index by clicking the **Create a new index** button.
3. After you make your selections, click **Next**.

Distribute source type configurations in Splunk Enterprise

If you create source types in Splunk Cloud Platform using Splunk Web, Splunk Cloud Platform manages the source type configurations automatically. However, if you have Splunk Enterprise and manage a distributed configuration, you must distribute new source type as described in this topic.

You can use either the "[Set source type](#)" or [source type management](#) pages in Splunk Web to create new source types, which you can then assign to inputs from specific files or directories, or for network inputs. Either of these pages saves a new source type to a `props.conf` configuration file on the local Splunk Enterprise instance. You can then distribute this file to other Splunk Enterprise instances so that they recognize the new source type.

You can use a new source type in a distributed environment where you have **forwarders** consuming data and then sending the data to indexers.

To install this new source type, follow these high-level steps:

1. Distribute the `props.conf` file that contains the source type definition to the `$SPLUNK_HOME/etc/system/local` directory on indexers that you want to index data with the source type you created.
2. Use the new source type when you define an input on forwarders that send data to those indexers.

When a forwarder sends data that has been tagged with the new source type to an indexer, the indexer can correctly process it into events.

Data preview props.conf file

When you create a source type in the "Set Sourcetype" page, the software saves the source type definition as a stanza in a props.conf file in the app that you selected when you saved the source type. If you later create additional source types, they are saved to the same props.conf file.

For example, if you selected the "Search and Reporting" app, the file resides in \$SPLUNK_HOME/etc/apps/search/local/props.conf. The only exception is the "System" app: If you choose that app when saving the source type, the file resides in \$SPLUNK_HOME/etc/system/local..

Note: A Splunk Enterprise instance might have multiple versions of some configuration files, in several directories. At run-time, Splunk Enterprise combines the contents of configuration files according to a set of precedence rules. For background on how configuration files work, see About configuration files and Configuration file precedence.

Distribute props.conf to other indexers

After you create source types, you can distribute props.conf to another Splunk Enterprise instance. That instance can then index any incoming data that you tag with the new source type.

A Splunk best practice is to place the configuration file in its own app directory on the target Splunk Enterprise instance; for example, \$SPLUNK_HOME/etc/apps/custom_sourcetype/local/.

To distribute configuration files to other Splunk instances, you can use a **deployment server** or another distribution tool. See the Updating Splunk Instances manual.

Note: Splunk software uses the source type definitions in props.conf to parse incoming data into events. For this reason, you can only distribute the file to a Splunk Enterprise instance that performs **parsing** (either an indexer or a **heavy forwarder**.)

Specify the new source type in forwarder inputs

Forwarders (with the exception of the heavy forwarder) do not have Splunk Web. This means that you must configure their inputs through the CLI or the inputs.conf configuration file. When you specify an input in that file, you can also specify its source type. For information on inputs.conf, read the section on inputs.conf in the Configuration file reference.

1. To tag a forwarder input with a new source type, add the source type to the input stanza in inputs.conf. For example:

```
[tcp://:9995]
sourcetype = new_network_type
```

2. Confirm that all of the indexers that the forwarder sends data to have copies of the props.conf file that contains the source type definition for new_network_type. When the forwarder sends data to the indexers, they can identify the new source type and correctly format the data.

Get data from files and directories

Monitor files and directories

To monitor files and directories in Splunk Cloud Platform, you must use a universal or a heavy forwarder in nearly all cases. You perform the data collection on the forwarder and then send the data to the Splunk Cloud Platform instance. Forwarders have three file input processors:

- **monitor**
- MonitorNoHandle
- upload

While you must use a forwarder for monitor and MonitorNoHandle input processors, you do not need to use a forwarder to upload a single file. You can upload a single file at a time to Splunk Cloud Platform using Splunk Web.

If you have Splunk Enterprise, you can monitor files using the CLI, Splunk Web, or the inputs.conf configuration file directly on your Splunk Enterprise instance. You can also use a universal or heavy forwarder, as you would with Splunk Cloud Platform.

You can use the monitor input to add nearly all your data sources from files and directories. However, you might want to use the upload input to monitor a file such as an archive of historical data, only one time.

On machines that run Windows Vista or Windows Server 2008 and higher, you can use the MonitorNoHandle input to monitor files that Windows rotates automatically. The MonitorNoHandle input works only on Windows machines.

You can add monitor or upload inputs using these methods:

- On a heavy forwarder: See [Monitor files and directories with Splunk Web](#).
- On a universal forwarder configured for Splunk Cloud Platform: See [Forward data from files and directories to Splunk Cloud Platform](#).
- On a universal or heavy forwarder, see the following:
 - ♦ [Monitor files and directories with the inputs configuration file](#).
 - ♦ [Monitor files and directories on Splunk Enterprise using the CLI](#).

You can add MonitorNoHandle inputs using either the CLI or the inputs.conf file.

If you use Splunk Web on a heavy forwarder to configure file monitor inputs, you can use the **Set Sourcetype** page to see how the Splunk platform indexes file. See [The Set Sourcetype page](#) for details.

How the monitor processor works

When you specify a path to a file or directory, the monitor processor consumes any new data written to that file or directory. Using the method of specifying the path, you can monitor live application logs such as those coming from Web access logs, Java 2 Platform Enterprise Edition (J2EE), or .NET applications. Splunk uses memory for each file monitored, even if the file is ignored.

The forwarder monitors and indexes the file or directory as new data appears. You can also specify a mounted or shared directory, including network file systems, as long as the forwarder can read from the directory. If the specified directory contains subdirectories, the monitor process recursively examines them for new files, as long as those directories can be

read.

Monitor inputs may overlap. So long as the stanza names are different, the forwarder treats them as independent stanzas and files matching the most specific stanza will be treated in accordance with its settings. You can include or exclude files or directories from being read by using allow lists or exclude lists.

If you disable or delete a monitor input, the forwarder does not stop indexing the files that the input references. It only stops checking those files again. To stop all in-process data indexing, you must restart the forwarder.

How the forwarder handles the monitoring of files during restarts

When you restart a forwarder, it continues processing files where it left off before the restart. It first checks for the file or directory specified in a monitor configuration. If the file or directory is not present on start, the forwarder checks for it every 24 hours from the time of the last restart. The monitor process scans subdirectories of monitored directories continuously.

How the forwarder monitors archive files

In order to monitor archived files, forwarders decompress archive files, such as a TAR or ZIP file, prior to processing. Splunk then processes these files in a single threaded format. The following types of archive files are supported:

- TAR
- GZ
- BZ2
- TAR.GZ and TGZ
- TBZ and TBZ2
- ZIP
- Z

If you add new data to an existing archive file, the forwarder reprocesses the entire file rather than just the new data. This can result in event duplication.

How the forwarder monitors files that the operating system rotates on a schedule

The monitoring processor detects file rotation and does not process renamed files that it has already processed (with the exception of .tar and .gz archives). See [How the Splunk platform handles log file rotation](#).

How the forwarder monitors nonwritable Windows files

Windows can prevent a forwarder from reading open files. If you need to read files while they are being written to, use the monitorNoHandle input.

Restrictions on file monitoring

The forwarder cannot monitor a file whose path exceeds 1024 characters (256 characters on Windows).

Forwarders also do not monitor files with a .splunk filename extension because files with that extension contain Splunk metadata. If you need to index files with a .splunk extension, use the `add oneshot` CLI command.

When to use upload or batch?

To index a static file once, select **Upload** in Splunk Web on Splunk Cloud Platform or Splunk Enterprise.

Otherwise, use the CLI commands `add oneshot` or `spool` on a forwarder to index a static file. See [Use the CLI](#) for details.

You can use the batch input type in the `inputs.conf` file to load files once and destructively. By default, the Splunk batch processor is located in the `$SPLUNKFORWARDER_HOME/var/spool/splunk` directory on the forwarder. If you move a file into this directory, the forwarder processes and deletes the file.

When to use MonitorNoHandle

This Windows-only input lets you read files on Windows systems as Windows writes to them. You must use either a universal or heavy forwarder to use the input for Splunk Cloud Platform. The input uses a kernel-mode filter driver to capture raw data as the data gets written to the file. You can use this input on files that the system locks open for writing, such as the Windows DNS server log file.

Restrictions for using MonitorNoHandle

The MonitorNoHandle input has the following restrictions:

- MonitorNoHandle only works on Windows Vista or Windows Server 2008 and higher operating systems. It does not work with earlier versions of Windows, nor does it work on operating systems that are not Windows.
- You can only monitor single files with MonitorNoHandle. To monitor more than one file, you must create a MonitorNoHandle input stanza for each file.
- You cannot monitor directories with MonitorNoHandle.
- If a file you choose to monitor with MonitorNoHandle already exists, the forwarder does not index its current contents, only new information that comes into the file as processes write to it.
- When you monitor a file with MonitorNoHandle, the source field for the file is MonitorNoHandle, not the name of the file. If you want to have the source field be the name of the file, you must set the field explicitly in `inputs.conf`. See [Monitor files and directories with inputs.conf](#).

Monitor files and directories in Splunk Enterprise with Splunk Web

You can use Splunk Web to add inputs from files and directories.

Forwarding a file requires additional setup. See the following topics:

- If you work with universal forwarders, see [Configure the universal forwarder in the Splunk Universal Forwarder Forwarder Manual](#).
- If you work with heavy forwarders, see [Enable forwarding on a Splunk Enterprise instance in the Forwarding Data manual](#).

Go to the Add New page

You add an input from the **Add Data** page in Splunk Web.

You can get there by either of these two ways.

Splunk Settings

1. Click **Settings > Data Inputs**.
2. Click **Files & Directories**.
3. Click **New** to add an input.

Splunk home

1. Click **Add Data** in Splunk home.
2. Click **Upload** to upload a file, **Monitor** to monitor a file, or **Forward** to forward a file.

Select the input source

1. To add a file or directory input, click **Files & Directories** in Splunk Web.
2. In the **File or Directory** field, type the full path to the file or directory.
To monitor a network drive that you have mounted on the system, enter `<myhost>/<mypath>` for *nix or `\\<myhost>\<mypath>` for Windows. Confirm that Splunk Enterprise has read access to the mounted drive, as well as to the files you want to monitor.
3. Choose how you want Splunk Enterprise to monitor the file:
 - ◆ Choose **Continuously Monitor** to set up an ongoing input. Splunk Enterprise monitors the file continuously for new data.
 - ◆ Choose **Index Once** to copy a file on the server into Splunk Enterprise.
4. Click **Next**.
If you specified a directory in the **File or Directory** field, Splunk Enterprise refreshes the screen to show fields for **include list** and **exclude list**. These fields let you type regular expressions that Splunk Enterprise then uses to match files for inclusion or exclusion. Otherwise, Splunk Enterprise proceeds to the **Set Sourcetype** page where you can preview how Splunk Enterprise proposes to index the events.

For more information on how to include and exclude data, see [Include or exclude specific incoming data](#).

Preview your data and set its source type

When you add a new file input, Splunk Enterprise lets you set the **source type** of your data and preview how the data looks once it is indexed. This lets you check that the data is formatted properly and make any necessary adjustments.

For information about the **Set Source Type** page, see [Apply the correct source types to your data](#).

If you skip the data preview, the **Input Settings** page appears.

You cannot preview directories or archived files. You also cannot preview inputs with the Log to Metrics source type.

Specify input settings

You can provide application context, the default host value, and the index in the **Input Settings** page. All parameters are optional.

1. Select the appropriate **Application context** for this input.
2. Set the **Host** value.

The Host value sets only the host field in the resulting events. Setting this value does not direct Splunk Enterprise to look on a specific host on your network.

3. Set the **Index** that you want Splunk Enterprise to send data to for this input. Leave the value as "default", unless you have defined multiple indexes and want to use one of those instead.
4. Click **Review** to review all of the choices you have made.

Review your choices

After you provide all input settings, review your selections. Splunk Web lists the options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.
2. If they do not match what you want, click the left-pointing bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**. A **Success** page appears and the Splunk platform begins indexing the specified file or directory.

Monitor Splunk Enterprise files and directories with the CLI

On Splunk Enterprise installations, you can monitor files and directories using the command line interface (CLI). To use the CLI, navigate to the `$SPLUNK_HOME/bin/` directory from a command prompt or shell, and use the `splunk` command in that directory.

The CLI has built-in help. Access the main CLI help by typing `splunk help`. Individual commands have their own help pages as well. Access that help by typing `splunk help <command>`.

CLI commands for input configuration

The following commands are available for input configuration using the CLI:

Command	Command syntax	Action
add monitor	<code>add monitor [-source] <source> [-parameter value] ...</code>	Monitor inputs from <source>.
edit monitor	<code>edit monitor [-source] <source> [-parameter value] ...</code>	Edit a previously added monitor input for <source>.
remove monitor	<code>remove monitor [-source] <source></code>	Remove a previously added monitor input for <source>.
list monitor	<code>list monitor</code>	List the currently configured monitor inputs.
add oneshot	<code>add oneshot <source> [-parameter value] ...</code>	<p>Copy the source file directly into Splunk Enterprise. This uploads the file once, but Splunk Enterprise does not continue to monitor it.</p> <p>You cannot use the <code>oneshot</code> command to monitor files on a remote Splunk Enterprise instance. You also cannot use the command with either recursive folders or wildcards as a source. Provide the exact source path of the file you want to monitor.</p>
spool	<code>spool <source></code>	<p>Copy the source file directly into Splunk Enterprise using the sinkhole directory. Similar to the <code>add oneshot</code> command, except that the file comes from the sinkhole directory, rather than being added immediately.</p> <p>You cannot use the <code>spool</code> command to monitor files on a remote Splunk Enterprise instance. You also cannot use the command with either recursive</p>

Command	Command syntax	Action
		folders or wildcards as a source. Provide the exact source path of the file you want to monitor.

CLI parameters for input configuration

Change the configuration of each data input type by setting additional parameters. To set parameters, use the syntax `-parameter value`.

You can set only one `-hostname`, `-hostregex`, or `-hostsegmentnum` per command.

Parameter	Required?	Description
<code><source></code>	Yes	Provide the path to the file or directory being monitored and uploaded for new input. This parameter can be the value itself. It does not have to follow a parameter flag. You can use either <code>./splunk monitor <source></code> or <code>./splunk monitor -source <source></code> .
<code>sourcetype</code>	No	Provide a <code>sourcetype</code> field value for events from the input source.
<code>index</code>	No	Provide the destination index for events from the input source.
<code>hostname</code> or <code>host</code>	No	Provide a host name to set as the host field value for events from the input source. These parameters are functionally equivalent.
<code>hostregex</code> or <code>host_regex</code>	No	Provide a regular expression to use to extract the host field value from the source key. These parameters are functionally equivalent.
<code>hostsegmentnum</code> or <code>host_segment</code>	No	An integer, which determines what "/" separated segment of the path to set as the host field value. If set to 3, for example, the third segment of the path is used. These parameters are functionally equivalent.
<code>rename-source</code>	No	Provide a value for the <code>source</code> field to be applied to data from this file.
<code>follow-only</code>	No	Set to true or false. Default is false. When set to true, Splunk Enterprise reads from the end of the source, like the <code>tail -f</code> Unix command. This parameter is not available for the <code>add oneshot</code> command.

Example 1: Monitor files in a directory

The following example shows how to monitor files in `/var/log/`.

Add `/var/log/` as a data input:

```
./splunk add monitor /var/log/
```

Example 2: Monitor windowsupdate.log

The following example shows how to monitor the Windows Update log file where Windows logs automatic updates, sending the data to an index called `newindex`.

Add `C:\Windows\windowsupdate.log` as a data input:

```
splunk add monitor c:\Windows\windowsupdate.log -index newindex
```

Example 3: Monitor Internet Information Server (IIS) logging

This example shows how to monitor the default location for Windows IIS logging.

Add `C:\windows\system32\LogFiles\W3SVC` as a data input:

```
./splunk add monitor c:\windows\system32\LogFiles\W3SVC
```

Example 4: Upload a file

This example shows how to upload a file into Splunk Enterprise. Splunk Enterprise consumes the file only once. It does not monitor it continuously.

Upload `/var/log/applog` on Unix or `C:\Program Files\AppLog\log.txt` on Windows directly into Splunk Enterprise with the `add oneshot` command:

Unix	Windows
<pre>./splunk add oneshot /var/log/applog</pre>	<pre>./splunk add oneshot C:\Program Files\AppLog\log.txt</pre>

You can also upload a file through the sinkhole directory with the `spool` command:

Unix	Windows
<pre>./splunk spool /var/log/applog</pre>	<pre>./splunk spool C:\Program Files\AppLog\log.txt</pre>

The result is the same with either command.

Monitor files and directories with inputs.conf

You can use the `inputs.conf` file to monitor files and directories with the Splunk platform. The `inputs.conf` file provides the most configuration options for setting up a file monitor input. If you use Splunk Cloud Platform, you can use either Splunk Web or a forwarder to configure file monitoring inputs.

To configure an input, add a **stanza** to the `inputs.conf` file in the `$SPLUNK_HOME/etc/system/local/` directory or your own custom application directory in `$SPLUNK_HOME/etc/apps/`. These locations are on the machine that runs Splunk Enterprise or the forwarder. To learn more about the `inputs.conf` file, see `inputs.conf` in the Splunk Enterprise *Admin Manual*.

You can configure multiple settings in an input stanza. If you don't specify a value for a setting, the Splunk platform uses the default for that setting. You can find the defaults for settings in the `$SPLUNK_HOME/etc/system/default/inputs.conf`

directory.

For more information about configuration files, see [About configuration files](#) in the Splunk Enterprise *Admin Manual*.

Configure a forwarder to send data to Splunk Cloud Platform

If you want to send Active Directory (AD) data to Splunk Cloud Platform, you must install and configure a forwarder before you begin making edits to configuration files on the forwarder.

1. Install a universal forwarder on the machine that you want to collect the AD data.
2. Install the Splunk Cloud Platform universal forward credentials package onto the machine.

Configure file monitoring with inputs.conf

1. On the machine that runs Splunk software, open a shell or command prompt.
2. Change the listed directory to the \$SPLUNK_HOME/etc/system/local directory.
3. If the inputs.conf file doesn't exist, create the file.
4. Open inputs.conf for editing with a text editor.
5. Add a stanza that references the files or directories that you want to monitor. For example, to monitor the /var/log/messages file on a *nix system, use this specification:

```
[monitor:///var/log/messages]
disabled = 0
```

To monitor the C:\Windows\System32\WindowsUpdate.log file on a Windows system, use this specification:

```
[monitor://C:\Windows\System32\WindowsUpdate.log]
disabled = 0
```

6. (Optional) Add settings that further configure the input, depending on what you want the input to do. See [Configuration settings](#) later in this topic, or see inputs.conf in the Splunk Enterprise *Admin Manual* for additional settings.

```
[monitor://path/to/file]
disabled = 0
setting1 = value
setting2 = value
...
```

7. Save the inputs.conf file and close it.
8. Either restart the Splunk platform or reload the configuration by running the following command. The Splunk platform prompts you for credentials if you reload the configuration.

```
./splunk _internal call /services/data/inputs/monitor/_reload -auth
```

Configuration settings

You can use the following settings in both `monitor` and `batch` input stanzas.

Setting	Description	Default
<code>host = <string></code>	Sets the host key to a static initial value for this stanza. The input processor uses the key during parsing and indexing to set the host field and uses the field during searching. The Splunk	The IP address or fully qualified domain name of the host where

Setting	Description	Default
	platform prepends the <code><string></code> with <code>host::</code> .	the data originated.
<code>index = <string></code>	<p>Sets the index where events from this input are stored. The Splunk platform prepends the <code><string></code> with <code>index::</code>.</p> <p>For more information about the index field, see <i>How indexing works in the Splunk Enterprise Managing Indexers and Clusters</i> manual.</p>	The <code>main</code> index or whatever you set the default index to.
<code>sourcetype = <string></code>	<p>Sets the sourcetype key or field for events from this input. This setting explicitly declares the source type for this data, as opposed to letting the Splunk platform determine it automatically. Declaring the sourcetype is important both for searchability and for applying the relevant formatting for this type of data during parsing and indexing.</p> <p>Sets the sourcetype key initial value. The Splunk platform uses the key during parsing and indexing to set the source type field and uses the source type field during searching. The Splunk platform prepends the <code><string></code> with <code>sourcetype::</code>.</p> <p>For more information about source types, see Why source types matter.</p>	The Splunk platform picks a source type based on various aspects of the data. There is no default.
<code>queue = parsingQueue indexQueue</code>	Specifies where the input processor deposits the events that it reads. Set to <code>parsingQueue</code> to apply the props.conf file and other parsing rules to your data. Set to <code>indexQueue</code> to send your data directly into the index.	<code>parsingQueue</code>
<code>_TCP_ROUTING = <tcpout_group_name>, <tcpout_group_name>, ...</code>	<p>Specifies a comma-separated list of tcpout group names. Use this setting to selectively forward your data to specific indexers by specifying the tcpout groups that the forwarder should use when forwarding the data.</p> <p>Define the tcpout group names in the outputs.conf file in <code>[tcpout:<tcpout_group_name>]</code> stanzas.</p>	The groups present in <code>defaultGroup</code> in <code>[tcpout]</code> stanza in the outputs.conf file.
<code>host_regex = <regular expression></code>	A regular expression that extracts the host from the file name of each input. Specifically, the Splunk platform uses the first group of the regular expression as the host.	The default "host =" setting, if the regular expression fails to

Setting	Description	Default
		match.
host_segment = <integer>	Sets the segment of the path as the host, using <integer> to determine the segment. For example, if host_segment = 2, host becomes the second segment of the path. Path segments are separated by the forward slash (/) character.	The default "host =" setting, if the value is not an integer or is less than 1.

Monitor syntax

Monitor input stanzas configure the Splunk platform to watch all files in the <path> or the <path> itself if it represents a single file. You must specify the input type before the path, so add three forward slashes in the path if the path includes the root directory on *nix machines.

You can use wildcards for the path. See [Specify input paths with wildcards](#).

```
[monitor://<path>]
<setting1> = <val1>
<setting2> = <val2>
...
```

The following are additional settings you can use when defining monitor input stanzas:

Setting	Description	Default
source = <string>	<p>Sets the source field for events from this input. You can use this setting when using the <code>MonitorNoHandle</code> input and want to set the source to the name of the file you're monitoring. Otherwise, don't override unless absolutely necessary. Consider using source types, tagging, and search wildcards instead. The input layer provides a more accurate string to aid in problem analysis and investigation by accurately recording the file from which the data was retrieved.</p> <p>The Splunk platform prepends the <string> with <code>source::</code>.</p>	The input file path, except in the case of <code>MonitorNoHandle</code> , where the default is <code>MonitorNoHandle</code> .
crcSalt = <string>	<p>Forces the Splunk platform to index files that have matching cyclic redundancy checks (CRCs). By default, the software performs CRCs only against the first few lines of a file. This behavior prevents indexing of the same file twice, even though you might have renamed it, such as with rolling log files. However, because the CRC counts only the first few lines of the file, it is possible for legitimately different files to have matching CRCs.)</p> <p>If set, the Splunk platform adds <code>string</code> to the CRC. If set to <code><SOURCE></code>, the Splunk platform adds the full source path to the CRC. Adding <code><SOURCE></code> ensures that each file being monitored has a unique CRC.</p> <p>Use caution with this setting for rolling log files. This setting can lead to the log file being re-indexed after it has rolled.</p>	N/A

Setting	Description	Default
	This setting is case-sensitive.	
<code>ignoreOlderThan = <time_window></code>	<p>Causes the input to stop checking files for updates if the file modification time has passed the <code><time_window></code> threshold. Stopping the file checking improves the speed of file tracking operations when you are monitoring directory hierarchies with large numbers of historical files. For example, when active log files share a directory with old files that no longer get writes.</p> <p>The Splunk platform doesn't index files whose modification time falls outside <code><time_window></code> when it first attempts to monitor the file.</p> <p>You must specify <code><number><unit></code>. For example, <code>7d</code> indicates one week. Valid units are <code>d</code> for days, <code>h</code> for hours, <code>m</code> for minutes, and <code>s</code> for seconds.</p>	0 (disabled)
<code>followTail = 0 1</code>	If set to 1, monitoring begins at the end of the file, much like <code>*nix tail -f</code> . This setting applies only to files the first time the Splunk platform attempts to monitor them. After that, the Splunk platform keeps track of the file using its internal file position records.	0
<code>whitelist = <regular expression></code>	If set, the Splunk platform monitors files whose names match the specified regular expression.	N/A
<code>blacklist = <regular expression></code>	If set, the Splunk platform doesn't monitor files whose names match the specified regular expression.	N/A
<code>alwaysOpenFile = 0 1</code>	<p>If set to 1, the Splunk platform opens a file to check if it's been indexed. This setting is useful only for files that don't update their modification time.</p> <p>Use this setting for monitoring files on Windows, and for Internet Information Server (IIS) logs.</p> <p>Use caution with this setting, as it increases load and slows down indexing.</p>	N/A
<code>recursive = true false</code>	If set to <code>false</code> , the Splunk platform doesn't look into subdirectories that it finds within a monitored directory.	true
<code>time_before_close = <integer></code>	The modification time delta required before the Splunk platform can close a file on end-of-file. This setting tells the system not to close files that have been updated in the past <code><integer></code> seconds.	3
<code>followSymlink = true false</code>	If set to <code>false</code> , the Splunk platform ignores symbolic links that it finds within a monitored directory.	true

MonitorNoHandle syntax

The `MonitorNoHandle` input monitors files without using Windows file handles. This input allows Splunk software to read special Windows log files such as the DNS debug server log. There are several limitations when using this input:

- The `MonitorNoHandle` input stanza works on Windows systems only.
- The `MonitorNoHandle` input stanza monitors only a single file.
- You can't use wildcards in the file or directory path.
- You can't monitor directories using a `MonitorNoHandle` stanza.
- The `MonitorNoHandle` input stanza reads only new data written to the monitored file. It doesn't ingest data already written to the file.
- A file monitored using `MonitorNoHandle` has the source metadata set to `MonitorNoHandle` by default. To specify another source, you must define it using the `source` setting in the `inputs.conf` file stanza.

For an example of a `MonitorNoHandle` stanza, see [MonitorNoHandle, single Windows file](#).

Batch syntax

Use batch to set up a one-time, destructive input of data from a source.

For continuous, nondestructive inputs, use the monitor input. The Splunk platform deletes data that it has indexed with the batch input.

```
[batch://<path>]
move_policy = sinkhole
<setting1> = <val1>
<setting2> = <val2>
...
```

When you define batch inputs, you must include the `move_policy = sinkhole` setting. This setting loads the file destructively. Don't use the batch input type for files that you don't want to delete after indexing.

To ensure that the Splunk platform indexes new events when you copy over an existing file with new contents, set the `CHECK_METHOD = modtime` setting in the `props.conf` file for the input source. This setting checks the modification time of the file and re-indexes it when the time changes. The Splunk platform indexes the entire file, which can result in duplicate events. For information about the `props.conf` file, see `props.conf`.

Examples of monitor input stanzas

*Single *nix file*

This example stanza configures the Splunk platform to index the single `/var/log/messages` file:

```
[monitor:///var/log/messages]
disabled = 0
sourcetype = unixlog
```

Single Windows directory

This Windows example configures the Splunk platform to monitor the `C:\Windows\Logs` directory and all the files in it:

```
[monitor://C:\Windows\Logs]
disabled = 0
```

Single Windows directory with spaces in filename

This Windows example configures the Splunk platform to monitor the C:\Program Files\VMWare directory and all the files in it:

```
[monitor://C:\Program Files\VMWare]
disabled = 0
```

Multiple Windows directories

This Windows example tells the Splunk platform to monitor all of the directories in C:\Windows\Debug:

```
[monitor://C:\Windows\Debug\*]
disabled = 0
```

Multiple *nix directories with a wildcard

This example configures the Splunk platform to monitor directories like /apache/foo/log and /apache/bar/log:

```
[monitor:///apache/.../log]
```

Multiple *nix files in one directory with a wildcard

This *nix example configures the Splunk platform to monitor multiple files in one directory, such as /apache/*.log:

```
[monitor:///apache/*.log]
```

MonitorNoHandle, single Windows file

This single Windows file example is from the Splunk Add-on for Microsoft Windows on Splunkbase:

```
##### Monitor Inputs for DNS #####
[MonitorNoHandle://$WINDIR\System32\Dns\dns.log]
sourcetype=MSAD:NT6:DNS
disabled=0
```

Batch

This batch example loads and deletes all files from the system/flight815/ directory:

```
[batch://system/flight815/*]
move_policy = sinkhole
```

Specify input paths with wildcards

You can configure inputs manually by editing the inputs.conf configuration file. In Splunk Cloud Platform, you can edit this file on a forwarder that collects the data. In Splunk Enterprise, you can edit this file on your Splunk Enterprise instance.

Input path specifications in the inputs.conf file do not use regular expressions (regexes) but rather wildcards that are specific to the Splunk platform. To specify wildcards, you must specify file and directory monitor inputs in the inputs.conf file.

When you configure an input path that has a wildcard, the Splunk platform instance must have at least read access to the entire path to the file you want to monitor with the wildcard. For example, if you want to monitor a file with the path

/var/log/server_a/tree_b/directory_c/file.log, the instance must have read permission in the following directories:

- var
- log
- server_a
- tree_b
- directory_c

If it does not have read access to all of the directories in the path, it cannot read the file, even if it has read access to the file directly.

Wildcard overview

A wildcard is a character that you can substitute for one or more unspecified characters when searching text or selecting multiple files or directories. You can use wildcards to specify the input path for a file or directory monitor input. See the following table for a description of the wildcards you can use and examples:

Wildcard	Description	Regex equivalent	Examples
...	<p>The ellipsis wildcard searches recursively through directories and any number of levels of subdirectories to find matches.</p> <p>If you specify a folder separator (for example, //var/log/.../file), it does not match the first folder level, only subfolders.</p>	.*	<p>/foo/.../bar.log matches /foo/1/bar.log, /foo/2/bar.log, /foo/1/2/bar.log, and so on. It does not match /foo/bar.log or /foo/3/notbar.log.</p> <p>Because a single ellipse searches recursively through all folders and subfolders, /foo/.../bar.log matches /foo/.../.../bar.log.</p>
*	<p>The asterisk wildcard matches anything in that specific folder path segment.</p> <p>Unlike ..., * does not recurse through subfolders.</p>	[^/]*	<p>/foo/*/bar matches the following:</p> <ul style="list-style-type: none">• /foo/1/bar• /foo/2/bar <p>but does not match</p> <ul style="list-style-type: none">• /foo/bar• /foo/1/2/bar <p>/foo/m*r/bar matches /foo/mr/bar, /foo/mir/bar, /foo/moor/bar, and so on.</p> <p>/foo/*.log matches all files with the .log extension, such as /foo/bar.log. It does not match /foo/bar.txt or /foo/bar/test.log.</p> <p>A single period (.) is not a wildcard, and is the regular expression equivalent of \..</p>

For more specific matches, combine the ... and * wildcards. For example, /foo/.../bar/* matches any file in the /bar directory within the specified path.

Wildcards and regular expression metacharacters

When determining the set of files or directories to monitor, the Splunk platform splits elements of a monitoring stanza into segments. Segments are blocks of text between directory separator characters (/ or \) in the stanza definition. If you configure a monitor stanza that contains segments with both wildcards and regular expression metacharacters, such as (,), [,], and |, those characters behave differently depending on where the wildcard is in the stanza.

If a monitoring stanza contains a segment with regular expression metacharacters before a segment with wildcards, the metacharacters are treated literally, as if you want to monitor files or directories with those characters in the file or directory names. The following example monitors the `/var/log/log(a|b).log` file:

```
[monitor:///var/log/log(a|b).log]
```

The `(a|b)` is not treated as a regular expression because no wildcards are present.

The following example monitors all files in the `/var/log()/` directory that begin with `log` and have the `.log` extension:

```
[monitor:///var/log()/log*.log]
```

The `()` is not treated as a regular expression because it is in the segment before the wildcard.

If the regular expression metacharacters occur within or after a segment that contains a wildcard, Splunk Enterprise treats the metacharacters as a regular expression and matches files to monitor accordingly. Consider the following example:

```
[monitor:///var/log()/log(a|b)*.log]
```

This example monitors all files in the `/var/log()/` directory that begin with either `loga` or `logb` and have the extension `.log`. The first set of `()` is not treated as a regular expression because the wildcard is in the following segment. The second set of `()` does get treated as a regular expression because it is in the same segment as the wildcard `*`.

The following example monitors all files in any subdirectory of the `/var/` directory named `loga.log` or `logb.log`:

```
[monitor:///var/.../log(a|b).log]
```

Splunk Enterprise treats `(a|b)` as a regular expression because of the wildcard `...` in the previous stanza segment.

Consider the following example:

```
[monitor:///var/.../log[A-Z0-9]*.log]
```

This example monitors all files in any subdirectory of the `/var/` directory that meet the following conditions:

1. Begin with `log`.
2. Contain a single capital letter (from A-Z) or number (from 0-9).
3. Contain any other characters.
4. End in `.log`.

The expression `[A-Z0-9]*` is treated as a regex because of the wildcard `...` in the previous stanza segment.

Here are sets of functional and not functional examples:

Functional examples

```
[monitor:///var/splunk/logA.log] (functional)
[monitor:///var/home/splunk/log0934213.log] (functional)
[monitor:///var/home/log0934213.log] (functional)
[monitor:///var/home/Alog0934213.log] (functional)
[monitor:///var/splunk/log1.log] (functional)
[monitor:///var/splunk/logA_3254.log] (functional)
```

Not functional examples

```
[monitor:///var/log093413.log] (not functional)
[monitor:///var/wer235.log] (not functional)
[monitor:///var/splunk/logA.log1] (not functional)
[monitor:///var/splunk/Alog342.log] (not functional)
```

Input examples

To monitor `/apache/foo/logs`, `/apache/bar/logs`, and `/apache/bar/1/logs`, create the following stanza:

```
[monitor:///apache/.../logs/*]
```

To monitor `/apache/foo/logs`, `/apache/bar/logs`, but not `/apache/bar/1/logs` or `/apache/bar/2/logs`, create the following stanza:

```
[monitor:///apache/*/logs]
```

To monitor any file directly under `/apache/` that ends in `.log`, create the following stanza:

```
[monitor:///apache/*.log]
```

To monitor all log files recursively in `D:\Program Files\Splunk\etc\apps`, create the following stanza:

```
[monitor:///D:\Program Files\Splunk\etc\apps\*\...\*.log]
```

To monitor any file under `/apache/` under any level of subdirectory that ends in `.log`, create the following stanza:

```
[monitor:///apache/.../*.log]
```

The `...` followed by a folder separator implies that the wildcard level folder will be excluded.

```
[monitor:///var/log/.../*.log]
```

The tailing logic becomes `^\var\log/.*/[^\]*\.log$`

Therefore, `/var/log/subfolder/test.log` matches, but `/var/log/test.log` does not match and will be excluded. To monitor all files in all folders, make the following changes:

```
[monitor:///var/log/]
```

```
whitelist=.log$
```

```
recursive=true
```

```
#true by default
```

Wildcards and allow lists

Splunk Enterprise defines allow lists and deny lists with standard Perl Compatible Regular Expression (PCRE) syntax. Splunk Cloud Platform doesn't define allow lists and deny lists natively in this way.

When you configure wildcards in a file input path, Splunk Enterprise creates an implicit allow list for that stanza. The longest wildcard-free path becomes the monitor stanza, and Splunk Enterprise translates the wildcards into regular expressions.

Splunk Enterprise anchors the converted expression to the right end of the file path, so that the entire path must be matched.

For example, in *nix, if you specify the `[monitor:///foo/bar*.log]` stanza in the `inputs.conf` configuration file, Splunk Enterprise translates the path into this:

```
[monitor:///foo/]
whitelist = bar[^/]*\.log$
```

On Windows, if you specify the `[monitor://C:\Windows\foo\bar*.log]` stanza in the `inputs.conf` file, Splunk Enterprise translates the path into this:

```
[monitor://C:\Windows\foo\]
whitelist = bar[^\\]*\.log$
```

In Windows, allow list and deny list rules don't support regular expressions that include backslashes. Use two backslashes (\\) to escape wildcards.

Include or exclude specific incoming data

You can use allow list and deny list rules to determine which files that the Splunk platform consumes or excludes when you **monitor** a directory or set of directories. You can also apply these settings to `batch` type monitoring inputs. When you define an allow list, Splunk Enterprise only indexes the files you specify. When you define a deny list, the Splunk platform ignores the specified files and processes all other files. You define these filters in the `input` stanza in the `inputs.conf` configuration file. If you want to apply the same filters across all of the forwarders in a Splunk platform deployment, you can set up a deployment server to perform this task. See *About deployment server and forwarder management in Updating Splunk Enterprise Instances*.

You don't need to define both an allow list and a deny list in a configuration stanza. The settings are independent, but the deny list filter overrides the allow list filter if a file is in both lists. If you define both filters and a file matches them both, Splunk Enterprise does not index that file.

The list rules use the regular expression syntax to define the match on the file name or path. The rules must be contained within a configuration stanza, such as `[monitor://<path>]`. The Splunk platform ignores filter lists that are not inside a stanza. When you define filter entries, you must use exact regular expression syntax.

Route and filter data

Instead of including or excluding your data inputs, you can filter specific events and send them to different queues or indexes.

Include files

Add the following line to your `monitor` stanza in the `local/inputs.conf` file for the app context that you defined the input.

```
whitelist = <your_custom_regex>
```

For example, to monitor only files with the `.log` extension, make the following change:

```
[monitor:///mnt/logs]
  whitelist = \.log$
```

Include multiple files

You can include multiple files in one line, using the `"|"` (pipe, or "OR") operator. For example, to include file names that contain `query.log` OR `my.log`, add the following line to the `inputs.conf` file:

```
whitelist = query\.log$|my\.log$
```

Or, you can include only files that match exactly. See the following example:

```
whitelist = /query\.log$|/my\.log$
```

The dollar symbol (`$`) anchors the regular expression to the end of the line. There is no space before or after the pipe (`|`) operator.

Exclude files

Add the following line to your `monitor` stanza in the `/local/inputs.conf` configuration file for the app context in which you defined the input.

```
blacklist = <your_custom_regex>
```

If you create a `blacklist` entry for each file you want to ignore, Splunk Enterprise activates only the last filter.

Example 1: Exclude only files with a .txt extension

To ignore and not monitor only files with the `txt` extension, add the following line to the `inputs.conf` file:

```
[monitor:///mnt/logs]
  blacklist = \.txt$
```

Example 2: Exclude files with a .txt or .gz extension

To ignore and not monitor all files with either the `.txt` extension or the `.gz` extension, add the following line to the `inputs.conf` file:

```
[monitor:///mnt/logs]
  blacklist = \.(?:txt|gz)$
```

Example 3: Exclude an entire directory

To ignore entire directories beneath a monitor input, add the following line to the inputs.conf file:

```
[monitor:///mnt/logs]
  blacklist = archive|historical|.bak$
```

This example configures Splunk Enterprise to ignore all files under /mnt/logs/ within the archive or historical directories, and all files ending in the *.bak extension.

Example 4: Exclude a file whose name contains a string

To ignore files whose names contain a specific string, add the following line to the inputs.conf file:

```
[monitor:///mnt/logs]
  blacklist = 2009022[89]file\*.txt$
```

This example ignores the webserver20090228file.txt and webserver20090229file.txt files under /mnt/logs/.

Example 5: Exclude Windows Event Code 4662 events whose Message field contains a specific value

To ignore Windows Event Code 4662 events whose Message field contains events with the value Account Name: "example account", add the following line to the inputs.conf file:

```
[WinEventLog:Security]
blacklist1 = EventCode = "4662" Message = "Account Name:\s+(example account) "
```

How the Splunk platform handles log file rotation

The Splunk platform recognizes when the operating system rotates a file that it's monitoring and doesn't read the rotated file a second time. For example, if the Splunk platform is monitoring /var/log/messages, it doesn't also read /var/log/messages1.

The monitoring processor picks up a new file and reads the first 256 bytes of the file. The processor then hashes this data into a beginning and ending cyclic redundancy check (CRC), which functions as a fingerprint that represents the file content. The Splunk platform uses this CRC to look up an entry in a database that contains all the beginning CRCs of files that it has seen before. If it finds a match in this database, the lookup returns a few values about the file. The most important values are the seekAddress, which represents the number of bytes into the known file that the Splunk platform already read, and the seekCRC, which is a fingerprint of the data at that location.

Using the results of this lookup, the Splunk platform can categorize the file.

How the Splunk platform categorizes a file

The Splunk platform categorizes a file based on the following outcomes of the CRC check.

The CRC doesn't find a match

If the CRC from the file beginning in the database doesn't have a match, this indicates a new file. The Splunk platform then completes these steps:

1. The Splunk platform reads the file data from the start of the file.
2. The Splunk platform updates the database with the new CRCs and Seek Addresses as it consumes the file.

The CRC finds a match

If the CRC from the file beginning in the database has a match, the content at the Seek Address location matches the stored CRC for that location in the file, and the file size is larger than the Seek Address that the Splunk platform stored, the file was read by the Splunk platform before but contains new data since it was last read. The Splunk platform then completes these steps:

1. The Splunk platform opens the file and goes to the seekAddress within the file, which is the end of the file when the Splunk platform last finished with it.
2. The Splunk platform reads the new data from that point.

If the CRC from the file beginning in the database has a match, but the content at the Seek Address location doesn't match the stored CRC at that location in the file, the file results from the following possibilities:

- The Splunk platform read some file with the same initial data, but some of the material that it read was modified in its place.
- The file is a different file that begins with the same content.

Because the database for content tracking is keyed to the beginning CRC, it can't track progress independently for the two different data streams and requires further configuration.

Configuring files with duplicate CRCs

Because the CRC start check runs against only the first 256 bytes of the file by default, non-duplicate files can have duplicate beginning CRCs, particularly if the files have identical headers. To handle such situations, make the following changes:

- Use the `initCrcLength` setting in the `inputs.conf` configuration file to increase the number of characters that the CRC uses for its calculation, and make it longer than any static header that might be present in the file at the beginning.
- Use the `crcSalt` setting when you configure the input for the file in the `inputs.conf` configuration file. If you configure the `crcSalt` setting to `<SOURCE>`, you ensure that each file has a unique CRC. In this way, the Splunk platform assumes that each path name contains unique content. See [Monitor files and directories with inputs.conf](#) for additional information on configuring this setting.

Do not use `crcSalt = <SOURCE>` with log files that the operating system routinely rotates, or any other scenario in which log files get renamed or moved to another location that the Splunk platform monitors. Doing so prevents the Splunk platform from recognizing log files across the rotation or rename, which results in the Splunk platform indexing the data more than once.

Get data from network sources

Get data from TCP and UDP ports

The Splunk platform lets you ingest data that comes in over a network port. It can accept data from both the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) network protocols.

Splunk Enterprise accepts this kind of data from **heavy forwarders** or **universal forwarders** that capture the data and send it to the Splunk Enterprise instance. For security, Splunk Enterprise accepts connections only from forwarders that have the correct Secure Sockets Layer (SSL) certificates to connect to the instance. If you want to send data from a TCP or UDP source such as the syslog service, use the universal forwarder to listen to the source and forward the data to your Splunk Enterprise deployment.

You can configure the forwarder to accept an input on any TCP or UDP port. The forwarder consumes any data that arrives on these ports. You can use this method to capture data from network services such as the syslog service. You can also set up the netcat service and bind it to a network port.

Network ports and Splunk Enterprise

TCP is the network protocol that underlies the Splunk Enterprise data distribution scheme. Use the TCP protocol to send data from any remote host to your Splunk Enterprise server. Splunk Enterprise can index remote data from any application that transmits over TCP.

Both Splunk Enterprise and the universal forwarder support monitoring over UDP. The best practice is to use TCP to send network data whenever possible. UDP is not desirable as a transport because, among other reasons, it does not guarantee the delivery of network packets.

For Syslog, the best practice is to use a syslog server, such as syslog-ng or Splunk Connect for Syslog.

When you monitor TCP network ports, the user that Splunk Enterprise or the universal forwarder runs as must have access to the port you want to monitor. On many UNIX operating systems, by default, you must run Splunk Enterprise as the root user to listen directly on a port below 1024.

Confirm how your network device handles external monitoring before you use the network monitoring input

Before you begin monitoring the output of a network device with the network monitor, confirm how the device interacts with external network monitors.

If you configure some network devices, such as a Cisco Adaptive Security Appliance (ASA), to log TCP network activity and the device can't connect to the monitor, it might reduce performance on the device or stop it from logging. By default, the Cisco ASA stops accepting incoming network connections when it encounters network congestion or connectivity problems.

Add a network input to a forwarder and send the data to Splunk Cloud Platform

Splunk Cloud Platform can accept network data that arrives only from either a universal or heavy forwarder. Before you can collect network data for Splunk Cloud Platform, you must have the following:

- An installed universal or heavy forwarder.
- The Splunk Cloud Platform universal forwarder credentials package. This package sets up the forwarding connection to your Splunk Cloud Platform instances and makes sure that data is transmitted securely between the forwarder and Splunk Cloud Platform.
- A text editor to edit the input and forwarding configurations.

Add a network input using a configuration file

On either a heavy forwarder or a universal forwarder, use a text editor to add a stanza for a network input to the `inputs.conf` configuration file in the `$SPLUNK_HOME/etc/system/local/` directory, or `%SPLUNK_HOME%\etc\system\local` on Windows, or in your own custom application directory in `$SPLUNK_HOME/etc/apps/`. If you haven't worked with Splunk configuration files before, see About configuration files in the Splunk Enterprise *Admin Manual* before you start.

While this procedure focuses on configuring forwarders to send network data to {Splunk Enterprise} instances, you can perform it without modifications on any Splunk Enterprise instance.

You can configure any number of settings and values for an input type. If you do not specify a value for a setting, the forwarder uses default values. These values are either defined in the Splunk platform code or exist in default configuration files within the `$SPLUNK_HOME/etc/system/default/` directory on the instance, or `%SPLUNK_HOME%\etc\system\default` on Windows..

Following is the general procedure to configure a network input:

1. Use a text editor to open the `inputs.conf` configuration file in one of the directories described in this section.
2. Add an input stanza that represents the kind of network data that you want to collect.
3. (Optional) Provide additional settings to configure how the Splunk platform handles the data.
4. Save the file and exit the text editor.
5. Restart the forwarder or Splunk Enterprise instance.

Configure a TCP network input

When you configure a TCP network input, the forwarder listens on that input for incoming network data over the TCP protocol.

This stanza configures the forwarder to listen to the server specified by `<remote server>` on the specified `<port>`. If `<remote server>` is blank, the software listens to all connections on the specified port.

```
[tcp://<remote server>:<port>]
<attribute1> = <val1>
<attribute2> = <val2>
...
```

The following settings control how the data is stored on the Splunk platform:

Setting	Description	Default
---------	-------------	---------

Setting	Description	Default
host = <string>	Sets the host field to a static value for this stanza. Also sets the host key initial value. The Splunk platform uses the key during parsing and indexing, in particular to set the host field. It also uses the host field at search time. The platform prepends <string> with host::.	The IP address or fully-qualified domain name of the host where the data originates.
index = <string>	Sets the index where Splunk Enterprise stores the events from this input. The Splunk platform prepends <string> with index::. On Splunk Cloud Platform, confirm that this index is present before you configure this setting.	main or whatever you set the default index to
sourcetype = <string>	Sets the sourcetype field for events from this input. Also declares the source type for this data, instead of letting Splunk Enterprise determine it. This is important both for searchability and for applying the relevant formatting for this type of data during parsing and indexing. Sets the sourcetype key initial value. Splunk Enterprise uses the key during parsing and indexing, in particular to set the source type field during indexing. Splunk Enterprise uses the source type field that it used at search time. The Splunk platform prepends <string> with sourcetype::.	Splunk Enterprise chooses a source type based on various aspects of the data. There is no hard-coded default.
source = <string>	Sets the source field for events from this input. The Splunk platform prepends <string> with source::. Do not override the source key unless absolutely necessary. The input layer provides a more accurate string to aid in problem analysis and investigation by recording the file from which the data is retrieved. Consider using source types, tagging, and search wildcards before overriding this value.	

The input file pathindexQueue

Specifies where the input processor deposits the events that it reads.

Set it to `parsingQueue` to apply the props.conf file and other parsing rules to your data. Set it to `indexQueue` to send your data directly into the index.

`parsingQueue` | `dns` | `none`

A value of `ip` sets the host to the IP address of the remote server.

`dns` sets the host to the DNS entry of the remote server.

`none` leaves the host as specified.

`ip`

Configure an encrypted TCP network input over SSL

Use this stanza type if you receive encrypted, unparsed network data from a forwarder or third-party system. Set <port> to the port on which the forwarder or third-party system sends unparsed, encrypted data.


```
[tcp-ssl:<port>]
```

Configure a UDP network input

This type of input stanza is similar to the TCP type, except that it listens on a UDP network port. If you provide `<remote server>`, the port that you specify only accepts data from that host. If you don't specify anything for `<remote server>`, the port accepts data that comes from any host.

```
[udp://<remote server>:<port>]
<attribute1> = <val1>
<attribute2> = <val2>
...
```

The following settings control how the Splunk platform stores the data:

Setting	Description	Default
<pre>host = <string></pre>	Sets the host field to a static value for this stanza. Also sets the host key initial value. Splunk Enterprise uses this key during parsing and indexing, in particular to set the host field. It also uses the host field at search time. The <code><string></code> is prepended with <code>host::</code> .	The IP address or fully-qualified domain name of the host where the data originated.
<pre>index = <string></pre>	Sets the index where Splunk Enterprise stores events from this input. The <code><string></code> is prepended with <code>index::</code> .	main or whatever you set the default index to
<pre>sourcetype = <string></pre>	<p>Sets the sourcetype field for events from this input. Also declares the source type for this data, as opposed to letting Splunk Enterprise determine it. This is important both for searchability and for applying the relevant formatting for this type of data during parsing and indexing.</p> <p>Sets the sourcetype key initial value. Splunk Enterprise uses the key during parsing and indexing, in particular to set the source type field during indexing. It also uses the source type field that it used at search time.</p> <p>The <code><string></code> is prepended with <code>sourcetype::</code>.</p>	Splunk Enterprise picks a source type based on various aspects of the data. There is no hard-coded default.
<pre>source = <string></pre>	<p>Sets the source field for events from this input. The <code><string></code> is prepended with <code>source::</code>.</p> <div> <p>Do not override the source key unless absolutely necessary. The input layer provides a more accurate string to aid in problem analysis and investigation by recording the file from which the data is retrieved. Consider use of source types, tagging, and search wildcards before overriding this value.</p> </div>	

The input file path.indexQueueSets where the input processor deposits the events that it reads. Set to `parsingQueue` to apply the props.conf file and other parsing rules to your data. Set to `indexQueue` to send your data directly into the index.`parsingQueue_rcvbuf = <integer>` Sets the receive buffer for the UDP port, in bytes. If the value is 0 or negative, Splunk Enterprise ignores the value.1,572,864 unless the value is too large for an OS. In this case, Splunk Enterprise halves the value from this default continuously until the buffer size is at an acceptable level.`no_priority_stripping = true | false`

Sets how Splunk Enterprise handles receiving syslog data.

If you set this setting to true, Splunk Enterprise does not strip the `<priority>` syslog field from received events.

Depending on how you set this setting, Splunk Enterprise also sets event timestamps differently. When set to true, Splunk Enterprise honors the timestamp as it comes from the source. When set to false, Splunk Enterprise assigns events the local time.

false (Splunk Enterprise strips <priority>.)no_appending_timestamp = true | falseSets how Splunk Enterprise applies timestamps and hosts to events.

If you set this setting to true, Splunk Enterprise does not append a timestamp and host to received events.

Do not configure this setting if you want to append timestamp and host to received events.

false (Splunk Enterprise appends timestamps and hosts to events)

Add a network input using Splunk Web

You can use Splunk Web to add network inputs on Splunk Enterprise or on a heavy forwarder that you want to configure to send data to Splunk Cloud Platform. Splunk Web is not available on universal forwarders, and Splunk Cloud Platform can't monitor network inputs directly using Splunk Web.

Go to the Add Data page

You can get to the Add data page in two ways.

To go to the Add Data page by Splunk Settings, follow these steps:

1. Click **Settings**.
2. Click **Data Inputs**.
3. Select **TCP** or **UDP**.
4. Click **New Local TCP** or **New Local UDP** to add an input.

To go to the Add Data page by Splunk Home, follow these steps:

1. Click the **Add Data** link in Splunk Home.
2. Click **Monitor** to monitor a network port on the local machine, or **Forward** to receive network data from another machine.

Forwarding a file requires additional setup.

3. If you select **Forward**, choose or create the group of forwarders you want this input to apply to.
4. Click **Next**.

Specify the network input

1. Click **TCP / UDP** to add an input.
2. Click the **TCP** or **UDP** button to select a TCP or UDP input.
3. In the **Port** field, enter a port number.
4. In the **Source name override** field, enter a new source name to override the default source value, if necessary.

Consult Splunk Support before changing the Source name override value.

5. If this is a TCP network input, decide whether you want this port to accept connections from all hosts or only one host in the **Only accept connection from** field. If you only want the input to accept connections from one host, enter the host name or IP address of the host. You can use wildcards to specify hosts.
6. Click **Next** to continue to the **Input Settings** page.

Specify input settings

The **Input Settings** page lets you configure source type, application context, default host value, and index. All of these parameters are optional.

1. Set the **Source type**. This is a default field that Splunk Enterprise adds to events and uses to determine processing characteristics, such as timestamps and event boundaries.
2. Set a value for **Host**. You have several choices:
 - ◆ Select **IP** to set the input processor to rewrite the host with the IP address of the remote server.
 - ◆ Select **DNS** to set the host to the DNS entry of the remote server.
 - ◆ Select **Custom** to set the host to a user-defined label.

Learn more about setting the host value in [About hosts](#).

The host value sets only the host field in the resulting events. Setting this value does not direct the Splunk platform to look on a specific host on your network.

3. For **Index**, set the index that you want Splunk Enterprise to send data to for this input. Leave the value as `default` unless you have defined multiple indexes to handle different types of events. In addition to indexes for user data, Splunk Enterprise has a number of utility indexes, which also appear in this dropdown box.
4. Click **Review**.

Review your choices

After entering all your input settings, review your selections. The Splunk platform lists the options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.
2. If they are not what you want, click the left angle bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

A Success page appears and the Splunk platform begins indexing the specified network input.

Add a network input using the CLI

You can use the CLI on a universal or heavy forwarder to configure it to send data to Splunk Cloud Platform. You can also use the CLI on a Splunk Enterprise instance. To access the CLI, navigate to the `$SPLUNK_HOME/bin/` directory (`%SPLUNK_HOME%\bin` on Windows) and use the `./splunk` command.

If you get stuck, the CLI has help. Access the CLI help by typing `splunk help`. Individual commands have their own help pages as well and can be accessed by typing `splunk help <command>`.

The following CLI commands are available for network input configuration:

Command	Command syntax	Action
add	<code>add tcp udp <port> [-parameter value] ...</code>	Add inputs from <port>.
edit	<code>edit tcp udp <port> [-parameter value] ...</code>	Edit a previously added input for <port>.
remove	<code>remove tcp udp <port></code>	Remove a previously added data input.
list	<code>list tcp udp [<port>]</code>	List the currently configured monitor.

The `<port>` is the port number on which to listen for data. The user you run the Splunk platform as must have access to this port.

You can modify the configuration of each input by setting any of these optional parameters:

Parameter	Description
<code>sourcetype</code>	Provide a <code>sourcetype</code> field value for events from the input source.
<code>index</code>	Provide the destination index for events from the input source.
<code>hostname</code>	Provide a host name to set as the host field value for events from the input source.
<code>remotehost</code>	Provide an IP address to exclusively accept data from.
<code>resolvehost</code>	Set to true or false (T F). Default is false. Set to true to use DNS to set the host field value for events from the input source.
<code>restrictToHost</code>	Provide a host name or IP address to accept connections only from the specified host or IP address.

Examples

The following example shows how to configure a UDP input to watch port 514 and set the source type to `syslog` on a *nix system:

```
./splunk add udp 514 -sourcetype syslog
```

The following example shows how to set the UDP input host value using DNS name resolution on a *nix system. Use `auth` with your username and password:

```
./splunk edit udp 514 -resolvehost true -auth admin:ch@ng3d
```

Change restricted hosts on a TCP network input

If you decide to only accept connections from a specific host when you create a TCP input, after you save that input, you can't change or remove that host later, either from Splunk Web or the CLI.

To change or remove the restricted host of a port, you must first delete the input that contains the old restricted host. Then, you must add a new input that either contains the new restricted host or has no restriction.

UDP packets and line merging

The Splunk platform doesn't index each UDP packet as an independent event. Instead, it performs event merging on the data stream and merges events together if they don't have a clear timestamp.

You can avoid this problem by editing the underlying source type in the `props.conf` file and setting the `SHOULD_LINEMERGE` setting to `false`. This keeps the Splunk platform from merging packets together.

How the Splunk platform handles syslog data over the UDP network protocol

If you run Splunk Cloud Platform, you can configure the Splunk universal forwarder to listen on a User Datagram Protocol (UDP) network port and forward that data to your Splunk Cloud Platform deployment.

Splunk Enterprise indexers can act as syslog servers that handle incoming data streams that comply with the syslog messaging standard. Splunk Enterprise can also act as a syslog message sender. Splunk Cloud Platform cannot send syslog messages, nor can it move messages from one device to another.

How the Splunk platform handles syslog inputs

When you configure a UDP network input to listen to a syslog-standard data stream on Splunk Enterprise or the universal forwarder, any syslog events that arrive through the input receive a timestamp and connected host field. The platform prepends these fields to each event before it indexes them. When you configure a universal forwarder to send data to Splunk Cloud Platform, Splunk Cloud Platform indexes the fields as it receives them from the universal forwarder.

The Splunk platform does not modify Transmission Control Protocol (TCP) network packets in this fashion. If you send syslog data over TCP, the platform does not strip priority information from the events. It does, however, prepend a host name and timestamp to the event unless you configure it not to. One TCP source stream will be assigned to one data pipeline and any others, so you should adjust for scalability.

How Splunk Enterprise handles syslog outputs

The following section applies to Splunk Enterprise only. Neither Splunk Cloud Platform nor the universal forwarder has the capability to forward events to another syslog server.

Splunk Enterprise can forward events to another syslog server. When it does, it prepends the priority information to the event so that the downstream syslog server can translate the events properly.

When the event reaches the downstream syslog server, that machine prepends a timestamp, priority, and connected host name, which is the Splunk Enterprise instance, to the event.

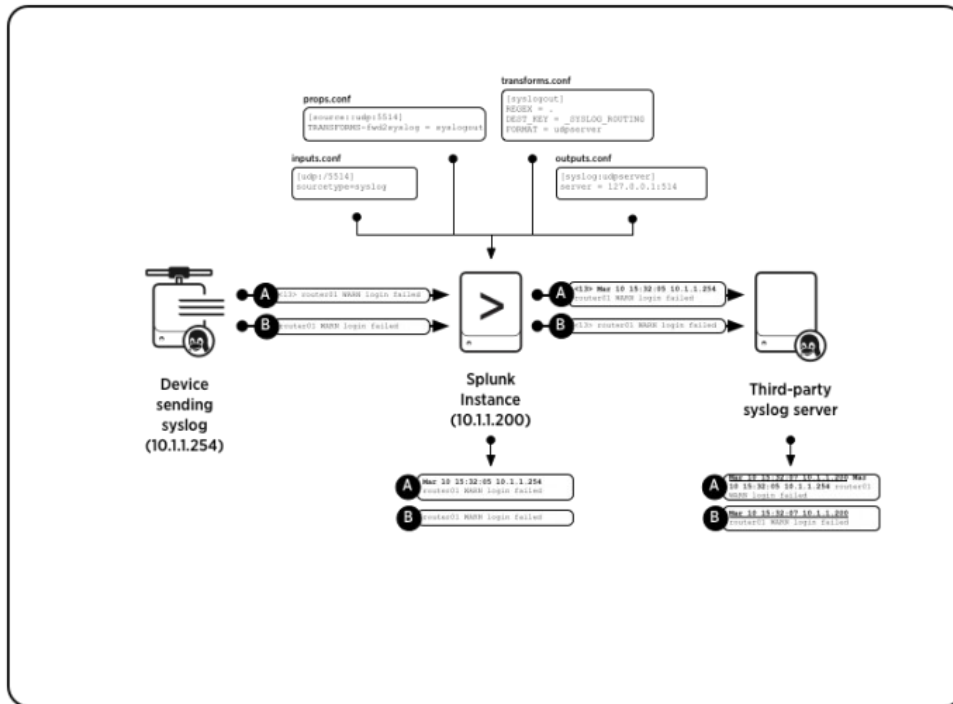
You can also prepend a timestamp and host name to the event at the time you forward the event to the syslog server. You do this as part of modifying the data as it leaves the Splunk Enterprise instance.

For information on configuring routing, filtering, and usage of source types, see *Route and filter data* in the Splunk Enterprise *Forwarding Data* manual and the `props.conf` spec file in the *Admin Manual*.

How Splunk Enterprise moves syslog events when you configure it to use syslog source type

The following section applies to Splunk Enterprise only. Splunk Cloud Platform isn't able to send syslog events to another downstream syslog server.

The following diagram shows how Splunk Enterprise moves two syslog messages from one syslog server to another. In the diagram, Splunk Enterprise listens on a UDP network port and indexes incoming events. On the other side, the same instance forwards events to a third-party syslog server.



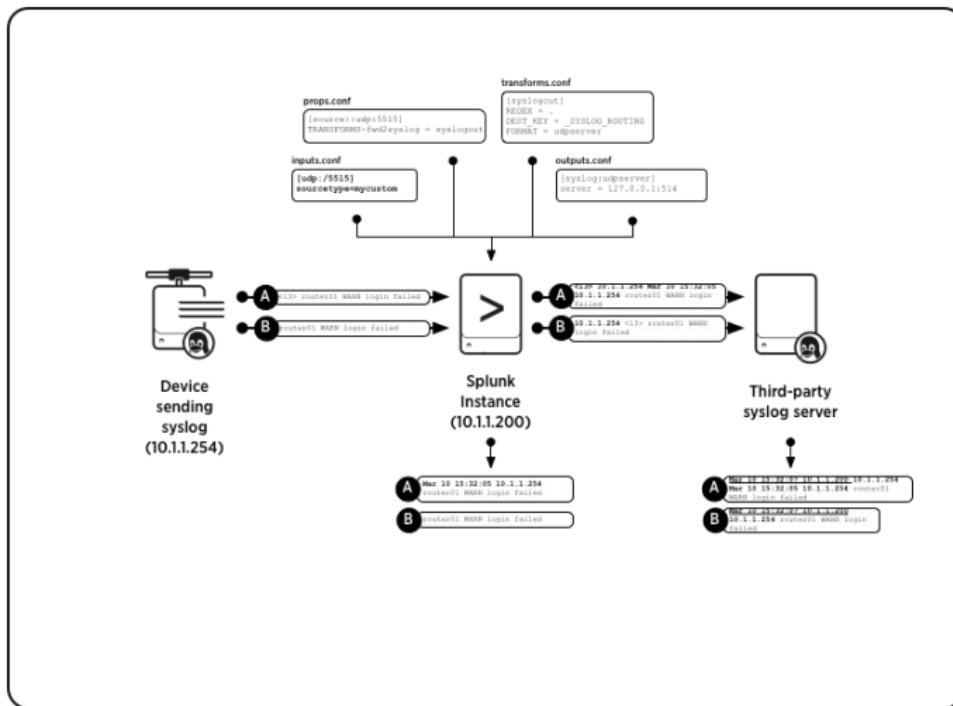
In the diagram, Message A originates as a syslog event and Message B originates as a similar event that does not have priority information associated with it. Upon receipt, Splunk Enterprise tags the events with a timestamp and the host that generated the event.

If you configured the instance as a forwarder, Splunk Enterprise then transforms the events by adding a priority header that you specify in the `outputs.conf` file before it forwards the events on to the syslog server. Once they arrive at the syslog server, that server prepends the timestamp and host data to the events as it received them from the Splunk Enterprise instance.

How Splunk Enterprise moves syslog events when you configure a custom source type

The following section applies to Splunk Enterprise only. Splunk Cloud Platform isn't able to move syslog events.

In this diagram, Splunk Enterprise has been configured to use a non-syslog source type. The initial Messages A and B are identical to the first example. In this example, Splunk Enterprise prepends the event with an originating host name or IP address.

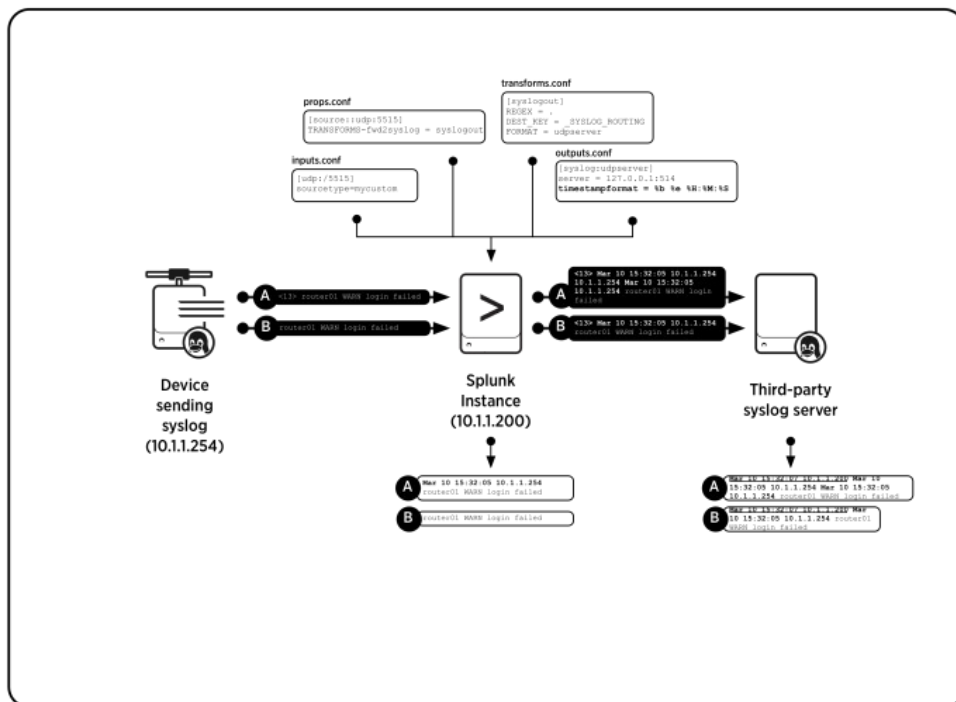


How Splunk Enterprise moves syslog events when you configure it with a timestamp

The following section applies to Splunk Enterprise only. Splunk Cloud Platform isn't able to move syslog events.

You can also configure Splunk Enterprise to add timestamps to syslog events when you forward those events. You could add a timestamp to the events when you don't want the downstream server to add its own timestamp.

The following diagram shows the required attribute and depicts how Splunk Enterprise deals with the data. The initial Messages A and B are identical to the first and second examples. Splunk Enterprise prepends the events with a timestamp and an originating host name or IP address.



Caveats to using Splunk Enterprise as a syslog server or message sender

The following section applies to Splunk Enterprise only. Splunk Cloud Platform isn't able to be used as a syslog server or message sender.

While you can configure Splunk Enterprise to receive syslog events directly, refrain from doing so for the following reasons:

- Splunk best practice involves setting up a separate machine that runs a syslog service to handle syslog tasks.
- The Splunk platform modifies syslog data by default as part of the indexing process. It assigns a timestamp and a host to the event.
- Syslog data streams to only one Splunk Enterprise instance in this scenario. In a deployment with multiple indexers, you must perform additional work to distribute the streams across those indexers
- If Splunk Enterprise or fails for any reason, any syslog messages that arrive during the downtime can be irrecoverably lost.

Don't substitute Splunk Enterprise for a syslog server in regular use unless you have no other options.

If you must retain raw syslog data, such as when a data retention policy requires access to untouched events, consider using a tool such as syslog-ng to simultaneously save the raw data to a log file and forward those events to your Splunk Enterprise instance. These tools give you the advantage of indexing the log file later if you want.

Send SNMP events to your Splunk deployment

Simple Network Management Protocol (SNMP) is a network protocol used to monitor network devices. SNMP data

sources include polling messages and traps.

An SNMP trap represents notifications or alerts that remote agents send. In a typical network environment, a central network management system collects the SNMP traps. SNMP polling requires the following components:

- Network agent devices that are capable of receiving polling requests
- A polling node that queries agents to request specific status information

Where to find SNMP support for the Splunk platform

The Splunk platform does not include native support for the SNMP protocol. You can choose from multiple Splunk apps and tools that offer support for SNMP:

- If the SNMP traps that your network management software collects are written to a log file, you can use a forwarder to monitor the log file and send the data to the Splunk platform. See [Monitor files and directories with inputs.conf](#).
- You can review the apps available on Splunkbase to assist you in collecting traps or polling SNMP data from the network. See the relevant apps on Splunkbase.
- You can use Splunk Stream to collect message statistics from SNMP messages using the built-in protocol support. See the Splunk Stream Installation and Configuration Manual.

See also

If you're looking for an example of installing and configuring the `snmptrapd` service on Linux, review the Splunk blog post for [Managing SNMP Traps with ITSI Event Analytics](#).

For guidance on integrating SNMP data sources into Splunk Enterprise, current Splunk customers can use OnDemand Services support offering. See [Support Programs](#).

Get Windows data

Monitor Windows data with the Splunk platform

You can bring any kind of Windows data into the Splunk platform. For example, you can index an Event Log channel, the Registry, or Active Directory. You also have available the standard set of Splunk inputs, such as files and directories, network monitoring inputs, and scripted inputs.

With Splunk Cloud Platform, as with many other input types, you must use either a universal or heavy forwarder that runs on Windows to collect data and send it to your Splunk Cloud Platform instance. Splunk Enterprise comes with installers for several versions of Windows and Windows Server. If you run Splunk Enterprise, you can install it or the universal forwarder on your Windows machines directly.

The following specialized inputs are available only on Windows installations:

Input	Description	Documentation
Windows Event Logs	Monitor events that the Windows Event Log service generates on any available event log channel on the machine. You can collect events on the local Windows machine or remotely by using either a universal forwarder or Windows Management Instrumentation (WMI).	Monitor Windows event log data with Splunk Cloud
Performance monitoring	Collect performance data on Windows machines with Splunk Cloud Platform and then alert or report on that data. Any performance counter that is available in Performance Monitor is also available to Splunk Cloud Platform. You can monitor performance locally or remotely through a universal forwarder, or by using WMI.	Monitor Windows performance
Remote monitoring over WMI	Splunk Cloud Platform can use WMI through a universal forwarder to access event log and performance data on remote machines.	Monitor data through Windows Management Instrumentation (WMI)
Registry monitoring	You can monitor changes to the local Windows Registry using the Registry monitoring capability. You can use a universal forwarder to gather Registry data from Windows machines and send the data to Splunk Cloud Platform.	Monitor Windows Registry data
Active Directory monitoring	Splunk Cloud Platform can audit any changes to the Active Directory, including changes to user, group, machine, and group policy objects. You can forward Active Directory data to another Splunk Enterprise server.	Monitor Active Directory

Forwarding Windows data to Splunk Cloud Platform

A Splunk Cloud Platform deployment that monitors Windows data consists of the following components:

- The Splunk Cloud Platform instance, where you see the Windows data.
- Universal forwarders on every Windows machine from which you want to collect Windows data.

Depending on the size of your Windows network, you might want to set up a tier of **intermediate forwarders** to aggregate and send the data to your Splunk Cloud Platform instance. If you want to transform this data in any way before you index it, you must use at least one Splunk Enterprise heavy forwarder to perform the transformations.

The universal forwarders on the Windows instances collect the Windows data. They then send the data to Splunk Cloud Platform using the Splunk Cloud Platform universal forwarder credentials package, which handles connecting and authenticating into the instance. If you set up an intermediate forwarder, this forwarder also uses the same credentials package to connect to and authenticate in Splunk Cloud Platform.

The Splunk Cloud Platform instance indexes the data and makes it available for you to search. You can install the Splunk App for Windows Infrastructure to view the Windows data in prebuilt dashboards and reports.

The universal forwarder must run as a user with access to the particular Windows data you want to collect. See [Install a Windows universal forwarder](#) for information on determining this Windows user.

Forwarding Windows data to Splunk Enterprise

Similar to forwarding Windows data to Splunk Cloud Platform, a Splunk Enterprise deployment that monitors Windows data consists of the Splunk Enterprise installation and, optionally, forwarders on every Windows machine from which you want to collect Windows data. Unlike a Splunk Cloud Platform deployment, Splunk Enterprise can exist on the same Windows machine.

If you want to forward Windows data from another Windows machine you can use a universal forwarder, like you can and must with a Splunk Cloud Platform deployment.

Considerations for installing Splunk Enterprise on Windows machines

When you install and deploy Splunk Enterprise on Windows, consider the following:

Consideration	Description
Authentication	To perform any operations on remote Windows machines in your network, Splunk Enterprise must run as a user with credentials to access those machines. Make these credentials available before deploying. See Considerations for deciding how to monitor remote Windows data .
Disk bandwidth	Splunk Enterprise indexers require lots of disk I/O bandwidth, particularly when indexing large amounts of data. Make sure that you configure any installed antivirus software to avoid monitoring Splunk Enterprise directories or processes, because such scans significantly reduce performance.
Shared hosts	Before you install Splunk Enterprise on a host that runs other services, such as Exchange, SQL Server, or a hypervisor, see Introduction to capacity planning for Splunk Enterprise in the <i>Capacity Planning</i> manual.

How to get Windows data into your Splunk deployment

You can collect the following Windows data with Splunk software:

Windows data you can collect	Link to supporting documentation
Event Logs	Monitor Windows event log data with Splunk Enterprise
File system changes	Monitor file system changes on Windows
Active Directory	Monitor Active Directory
Data through the Windows Management Instrumentation (WMI) infrastructure	Monitor data through Windows Management Instrumentation (WMI)
Registry data	Monitor Windows Registry data
Performance metrics	Monitor Windows performance
Host information	Monitor Windows host information
Print information	Monitor Windows printer information
Network information	Monitor Windows network information

Because only Windows machines provide these types of data, only the Windows version of the Splunk platform can get

the data. Other operating systems cannot collect Windows data directly. You can send Windows data from Windows machines to Splunk platform instances that don't run Windows. If you use Splunk Cloud Platform and want to monitor these inputs, the Splunk universal forwarder is the only option.

How the Splunk platform interacts with Windows modular and scripted inputs on start-up and shutdown

When you configure a scripted or **modular** Windows data input in the Splunk platform, the splunkd service sends a signal to the input to begin collecting the data. Similarly, when you shut down the Splunk platform cleanly, the service sends a different signal to the inputs to tell them to stop collecting data, clean up, and exit.

The following table shows the signals, or control messages, that the splunkd service sends to modular and scripted Windows inputs during start-up and shutdown.

Process	Signal
Start-up	CreateProcess
Shut-down	CTRL_BREAK_EVENT

Use Splunk Web to collect Windows data

Almost all Windows inputs let you use the Splunk Web interface to get data in Splunk Enterprise. The exception is the `MonitorNoHandle` input, which you must set up with a configuration file.

Follow these steps to collect Windows data in Splunk Web:

1. Log into your Splunk deployment.
2. Click **Settings** > **Data inputs**.
The **Data inputs** page appears.
3. From the list of available inputs, find the Windows input that you want to add from the list of available inputs.
4. Click **Add new** in the Actions column for the input.
5. Follow the instructions for the input type you selected.
6. Click **Save**. In most cases, data collection begins immediately.

Use configuration files to collect Windows data

In cases where you can't use Splunk Web to configure Windows inputs, such as on a universal forwarder, you must use configuration files. The universal forwarder installer on Windows lets you configure some Windows inputs at installation time.

Configuration files offer more control over Splunk Web in many cases. Some inputs can only be configured this way.

Follow these steps to use configuration files to collect Windows data:

1. Open a command prompt or PowerShell window.
2. Change the directory to the `%SPLUNK_HOME%\etc\system\local` directory on your Splunk platform instance.
3. Edit the `inputs.conf` configuration file in this directory. You might need to create the file if it doesn't already exist.
4. Add inputs to the `inputs.conf` file by defining input stanzas, settings, and values.
5. Save the file and close it.
6. Restart the Splunk platform instance.
The software reloads the configuration files and begins collecting data based on the new configuration.

Considerations for deciding how to monitor remote Windows data

If you want to monitor Windows data that is not on a local Windows machine, consider these options.

The Splunk platform collects remote Windows data for indexing in one of two ways:

- From Splunk forwarders
- Using Windows Management Instrumentation (WMI)

For Splunk Cloud Platform deployments, you must use the Splunk Universal Forwarder on a Windows machine to monitor remote Windows data.

Using a forwarder to collect remote Windows data

Use a universal forwarder to get remote Windows data whenever possible. The universal forwarder has these advantages:

- It uses minimal network and disk resources on the installed machines.
- You can install it as a non-privileged user, whereas you require administrative access for WMI.
- If you install it as the Local System user, then it has administrative access to the machine and requires no authentication to get data from there, as WMI does.
- It scales well in large environments and is easy to install. You can install it manually, with either a Microsoft deployment tool like System Center Configuration Manager (SCCM) or a third party distribution solution such as Puppet.

After you install a universal forwarder, it gathers information locally and sends it to Splunk Enterprise. You tell the forwarder what data to gather either during the installation process or later, by distributing configuration updates manually or with a **deployment server**. You can also install add-ons on the universal forwarder.

There are some drawbacks to using the universal forwarder, depending on your network configuration and layout. See [Splunk forwarders versus WMI](#) in this topic.

Using WMI to collect remote Windows data

The WMI framework lets the Splunk platform collect virtually any kind of data from remote Windows machines. In this configuration, the Splunk platform runs as a user that you specify at installation or later on, in the Services control panel. For more information, see Choose the Windows user Splunk Enterprise should run as in the *Installation Manual*.

This configuration has the following advantages:

- It gives the Splunk platform as much access to the network as the specified account has for remote access.
- It lets indexers collect data from remote Windows machines across the enterprise and place that data into a central repository.
- It is ideal for small to medium-sized networks with at least one indexer in each network segment.

There are some caveats to this method of collection. See [Considerations for getting data over WMI](#) and [Splunk forwarders versus WMI](#) later in this topic.

Also, while Active Directory (AD) monitoring does not use WMI, it has the same authentication considerations as data inputs that do use it. For information on how the Splunk platform monitors AD, see [Monitor Active Directory](#) in this manual.

Considerations for getting data over WMI

When collecting remote Windows data over WMI, take into account the following considerations:

- Authentication for remote Windows data
- Using managed system accounts to access Windows data
- Network and I/O usage

Authentication for remote Windows data

Windows requires authentication for remote operations. Failure to understand how the Splunk platform interacts with Windows over the network can lead to suboptimal search results or no results at all.

When you install the Splunk platform, you can specify the Splunk platform to run as the Local System user or another user. This choice has ramifications for both installation and data collection. For more information, see *Choose the Windows user Splunk Enterprise should run as* in the *Installation Manual*.

The user you tell the Splunk platform to run as determines the kind of data it can retrieve from remote machines. To get the data you want, you must provide an appropriate level of permission to this user.

In most cases, configure the Splunk platform user account with least-permissive access to the data sources you want to collect. This means you must do the following:

- Add the user to various domain security groups.
- Modify the access control lists of various AD objects, depending on the data sources you need to access.

If your AD domain security policy enforces password changes regularly, you must also do the following:

- Confirm that either the Splunk platform user password never expires, or that you must manually change the password before it expires, as defined by the password policy.
- Restart the Splunk services that run as that account on all hosts in your network once you change the password.

You must also give this account the "Deny log on locally" user rights assignment in Local Security Policy to prevent the user from logging in interactively to workstations. This method gives you more control and is more secure than giving domain administrator access.

See the other topics in this chapter that deal with remote access to Windows machines for more information. Check the Security and remote access considerations sections for how to configure the user that the Splunk platform runs as for least-permissive access.

Network and I/O usage

Monitor network bandwidth usage closely, especially in networks with slow or thin WAN links. For this reason alone, universal forwarders are a better choice for large remote data collection operations.

Disk bandwidth is a concern as well. Always configure antivirus scanner drivers and drivers that intermediate between the Splunk platform and the operating system to ignore the Splunk platform directory and processes, regardless of the type of installation.

Splunk forwarders versus WMI

Use a universal forwarder to get data in from a remote Windows host. A universal forwarder offers the most types of data sources, provides more detailed data (for example, in performance monitoring metrics), minimizes network overhead, and reduces operational risk and complexity. It is also more scalable than WMI in many cases.

In circumstances where you collect data remotely, such as when corporate or security policy restricts code installation, or when there are performance or interoperability concerns, you can use the native WMI interface to collect event logs and performance data.

WMI and forwarders have the following main areas of tradeoff:

- Performance
- Deployment
- Management

Performance

With respect to performance, a forwarder is a better choice when the following circumstances apply:

Scenario	Considerations
You collect local event logs or flat files.	A forwarder requires less CPU and performs basic precompression of the data in an effort to reduce network overhead.
You want to collect data from a machine without having to worry about authentication.	When you install a forwarder as the Local System user, it has administrative access to the machine, letting you collect any data from it.
You want to collect data from busy hosts such as AD domain controllers or machines that consistently experience periods of high utilization, such as Exchange, SQL Server, Oracle, VMWare, Hyper-V, or SharePoint servers.	Consider using a forwarder in this scenario because WMI might have problems keeping up with the amount of data these services generate. WMI polling is best-effort by design, and the Splunk platform also throttles WMI calls to prevent unintentional denial-of-service attacks.
You are concerned about CPU and network utilization.	Forwarders use as little of these resources as possible, while WMI uses more CPU and network resources to transfer data.
You are concerned about scalability.	Universal forwarders scale very well. Heavy forwarders do not scale as well as universal forwarders, but both types of forwarder scale considerably better than WMI.

WMI is a better choice when you have concerns about memory usage on a system with high memory utilization. Because forwarders have more polling options available and reside on the local machine while collecting data, they use more memory than WMI does.

Deployment

A forwarder is a better choice for deployment when the following circumstances apply:

- You have control of the base build of the OS, as is the case when you create system images.
- You have many data sources to collect, particularly if the data requires transformation of any kind.

Except for a few cases, you cannot use a universal forwarder to process data before it reaches the indexer. If you need to make any changes to your data before you index it, you must use a heavy forwarder.

WMI is a better choice when the following circumstances apply:

- You don't have control of the base OS build, domain administrator access, or local administrator privileges on the machines from which you want to get data.
- You want or need only a limited set of data from a large number of hosts, like CPU data for usage billing.

A common deployment scenario is to first test using remote polling, then add successful or useful data inputs to your forwarder configurations later or when you do large scale forwarder installations.

Management

Both mechanisms offer logging and alerting to advise if a host comes on or offline or is unreachable. To prevent an unintentional denial of service attack, the WMI polling service in the Splunk platform polls less frequently over time if it cannot contact a host and eventually stops polling unreachable hosts altogether.

Do not use remote polling over WMI for machines that are frequently offline, such as laptops or dynamically provisioned virtual machines.

The following table shows a list of data sources and indicates which data collection types are appropriate for each data source:

Data source	Local forwarder	WMI
Event logs	Yes	Yes*
Performance	Yes	Yes
Registry	Yes	No
Active Directory	Yes	No
Log files	Yes	Yes**
Crawl	Yes	No

* For remote event log collection, you must know the name of the event log you want to collect. On local forwarders, you have the option to collect all logs, regardless of name.

** The Splunk platform supports remote log file collection using the `\\SERVERNAME\SHARE` syntax. However, you must use the Common Internet File System (CIFS) or Server Message Block (SMB) as your application layer file access protocol, and the Splunk platform must have at least read access to both the share and the underlying file system.

Search Windows data on a non-Windows instance of the Splunk platform

You can index and search your Windows data on a non-Windows Splunk deployment, but you must first use a Windows instance of the Splunk platform to get the Windows data. You can do this by installing a Splunk forwarder on the Windows computer and configuring it to forward Windows data to the non-Windows instance of the Splunk platform.

You can proceed in one of the following ways:

- Set up forwarders locally on each Windows machine that you want data. These forwarders can send the Windows data to the non-Windows receiving instance.
- Set up a forwarder on a separate Windows machine. The forwarder can use WMI to collect data from all the Windows machines in the environment and then forward the combined data to a non-Windows receiving instance of the Splunk platform.

Monitor Active Directory

The Active Directory (AD) database, also known as the NT Directory Service (NTDS) database, is the central repository for user, computer, network, device, and security objects in a Windows AD domain or forest. You can use Splunk Enterprise to record changes to AD, such as the addition or removal of a user, host, or domain controller (DC).

If you use Splunk Cloud Platform, you must use the Splunk Universal forwarder to collect Active Directory data from a Windows domain controller or member machine and forward that data to Splunk Cloud Platform. On Splunk Enterprise, you can also use the universal forwarder, or you can install Splunk Enterprise directly onto a Windows machine and collect the AD data that way.

You can configure AD monitoring to watch for changes to your Active Directory forest and collect user and machine metadata. You can use this feature combined with dynamic list lookups to decorate or modify events with any information available in AD. See About lookups in the *Knowledge Manager Manual*.

After you configure Splunk Enterprise to monitor your Active Directory, it takes a baseline snapshot of the AD schema. It uses this snapshot to establish a starting point for monitoring.

The AD monitoring input runs as a separate process called `splunk-admon.exe`. It runs once for every Active Directory monitoring input you define in Splunk Enterprise.

Reasons to monitor Active Directory

If you maintain the integrity, security, and health of your Active Directory, then what happens with it day to day is a concern. With Splunk Enterprise, you can monitor what and when things changed in your AD and who changed them.

You can transform this data into reports for corporate security compliance or forensics, for example. You can also use the data retrieved for intrusion alerts for immediate response. Additionally, you can create health reports with the data indexed for future AD infrastructure planning activities, such as assignment of operations master roles, AD replicas, and global catalogs across DCs.

Requirements

You must meet the following requirements to monitor an Active Directory schema:

- Splunk Enterprise must run on Windows. See Install on Windows in the *Installation Manual*.
- Splunk Enterprise must run as a domain user. See Choose the Windows user Splunk Enterprise should run as in the *Installation Manual*.
- The user Splunk Enterprise runs as must have read access to all AD objects that you want to monitor.

Technical considerations for monitoring Active Directory

For best results with monitoring AD, note the following considerations:

- The AD monitor is only available on the Splunk platform on Windows. Splunk Cloud Platform cannot monitor AD directly.
- While you cannot monitor AD changes from a *nix version of Splunk Enterprise, you can forward AD data from a Windows version of Splunk Enterprise or the universal forwarder to a *nix indexer.
- The AD monitoring process can run under a full instance or on any kind of forwarder.

- The host that monitors changes to AD must belong to the domain or forest you want to monitor.
- The user that Splunk Enterprise runs as must also be part of the domain.
- The permissions that the user has determine which parts of AD Splunk can monitor.

For information on deciding which user Splunk Enterprise runs as at installation time, see Choose the Windows user Splunk Enterprise should run as in the *Installation Manual*.

How the AD monitor interacts with AD

When you set up an AD monitoring input, the input connects to an AD domain controller to authenticate and, if necessary, performs any security ID (SID) translations while it gathers the AD schema or changes events.

The AD monitor uses the following logic to interact with Active Directory after you set it up:

1. If you specify a domain controller when you define the input, then the input uses that domain controller for AD operations. You can specify a domain controller either with the `targetDc` setting in `inputs.conf` or the `Target domain controller` field in Splunk Web.
2. If you do not specify a domain controller, then the input does the following:
 1. The input attempts to use the local system cache to authenticate or resolve SIDs.
 2. If the monitor cannot authenticate or resolve SIDs that way, it attempts a connection to the domain controller that the machine that runs the input used to log on.
 3. If that does not work, then the input attempts to use the closest AD domain controller that has a copy of the Global Catalog.
3. If the domain controller that you specify is not valid or a domain controller cannot be found, then the input generates an error message.

The AD monitor does not chase LDAP referrals

If the AD monitor makes a Lightweight Directory Access Protocol (LDAP) query and receives a referral, it does not chase this referral to complete the query. An LDAP referral represents a problem with your LDAP configuration and you or your designated administrators must determine and fix the configuration problem within AD.

Configure Active Directory monitoring with configuration files

You can configure AD monitoring either in Splunk Web or by editing configuration files. You can access more options, such as the ability to configure monitors for multiple DCs, when using configuration files.

The `inputs.conf` configuration file controls Active Directory monitoring configurations. Edit copies of `inputs.conf` in the `%SPLUNK_HOME%\etc\system\local` directory. If you edit them in the default directory, an upgrade overwrites your changes. For more information about configuration file precedence, see Configuration file precedence.

1. Open `%SPLUNK_HOME%\etc\system\local\inputs.conf` for editing. Create this file if it does not exist.
2. Add the appropriate AD monitoring stanzas and settings.

By default, when you enable AD monitoring inputs, Splunk Enterprise gathers AD change data from the first domain controller that it can attach to. If that is acceptable, no further configuration is necessary.

inputs.conf settings

`inputs.conf` contains one stanza for each AD monitoring input, with a header like the following:

```
[admon://<name of stanza>]
```

In each stanza, you can specify the following settings:

Setting	Required?	Description	Default
<code>targetDc</code>	Yes	<p>The unique name of the domain controller you want to use for AD monitoring.</p> <p>Specify a unique name for this setting if the following circumstances apply:</p> <ul style="list-style-type: none"> • You have a very large AD and you only want to monitor information in a particular Organizational Unit (OU), subdomain, and so on. • You have a specific read-only domain controller that can be used for monitoring purposes in a high security environment. • You have multiple domains or forests with transitive trusts established and want to target a different tree than the one where the host that runs Splunk Enterprise resides. • You want to configure multiple AD monitoring inputs to target multiple domain controllers. For example, to monitor AD replication across a distributed environment. <p>To target multiple DCs, add another <code>[admon://<uniquename>targetDc]</code> stanza for a target in that tree.</p>	the local host
<code>startingNode</code>	No	<p>A fully qualified Lightweight Directory Access Protocol (LDAP) name (for example: "LDAP://OU=Computers,DC=ad,DC=splunk,DC=com") that specifies where in the AD tree that Splunk Enterprise begins its indexing. The software starts there and enumerates down to sub-containers, depending on the configuration of the <code>monitorSubtree</code> setting.</p> <p>The value of <code>startingNode</code> must be within the scope of the DC you are targeting for Splunk Enterprise to get AD data.</p>	The highest root domain in the tree that Splunk Enterprise can access
<code>monitorSubtree</code>	No	How much of the target AD container to index. A value of 0 means to index only the target container, and not traverse into subcontainers within that container. A value of 1 means to enumerate all sub-containers and domains that it has access to.	1 (monitor all domains that Splunk Enterprise has access to)
<code>baseline</code>	No	Whether or not the input enumerates all existing available AD objects when it first runs. A value of 0 means not to set a baseline. A value of 1 means to set a baseline.	1 (set the baseline)
<code>index</code>	No	The index to route AD monitoring data to.	the <code>default</code> index
<code>disabled</code>	No	Whether or not Splunk Enterprise runs the input. A value of 0 means that the input is enabled, and a value of 1 means that the input is disabled.	1 (disabled)

Example AD monitoring configurations

The following are examples of how to use the `inputs.conf` file to monitor desired portions of your AD network.

See the following example to index data from the top of the AD directory:

```
#Gather all AD data that this server can see
```

```
[admon://NearestDC]
targetDc =
startingNode =
```

See the following example to use a DC that is at a higher root level than an OU you want to target for monitoring:

```
# Use the pri01.eng.ad.splunk.com domain controller to get all AD metadata for
# the Computers OU in this forest. We want schema data for the entire AD tree, not
# just this node.
```

```
[admon://DefaultTargetDc]
targetDc = pri01.eng.ad.splunk.com
startingNode = OU=Computers,DC=eng,DC=ad,DC=splunk,DC=com
```

See the following example to monitor multiple domain controllers:

```
# Get change data from two domain controllers (pri01 and pri02) in the same AD tree.
# Index both and compare/contrast to ensure AD replication is occurring properly.
```

```
[admon://DefaultTargetDc]
targetDc = pri01.eng.ad.splunk.com
startingNode = OU=Computers,DC=eng,DC=ad,DC=splunk,DC=com
```

```
[admon://SecondTargetDc]
targetDc = pri02.eng.ad.splunk.com
startingNode = OU=Computers,DC=eng,DC=ad,DC=splunk,DC=com
```

Sample AD monitoring output

When the AD monitoring utility runs, it gathers AD change events, which the Splunk platform then indexes. To view these events as they arrive, use the Search & Reporting app.

There are several types of AD change events that the Splunk platform can index. Examples of these events follow. Some of the content of these events is obscured or altered for publication purposes.

Update event

When an AD object changes, the Splunk platform generates an update event. The software logs this change as type

```
admonEventType=Update.
```

```
2/1/10
3:17:18.009 PM
```

```
02/01/2010 15:17:18.0099
```

```
dcName=stuff.splunk.com
```

```
admonEventType=Update
```

```
Names:
```

```
objectCategory=CN=Computer,CN=Schema,CN=Configuration
name=stuff2
displayName=stuff2
distinguishedName=CN=stuff2,CN=Computers
```

```
Object Details:
```

```
sAMAccountType=805306369
sAMAccountName=stuff2
logonCount=4216
accountExpires=9223372036854775807
objectSid=S-1-5-21-3436176729-1841096389-3700143990-1190
```

```

primaryGroupID=515
pwdLastSet=06:30:13 pm, Sat 11/27/2010
lastLogon=06:19:43 am, Sun 11/28/2010
lastLogoff=0
badPasswordTime=0
countryCode=0
codePage=0
badPwdCount=0
userAccountControl=4096
objectGUID=blah
whenChanged=01:02.11 am, Thu 01/28/2010
whenCreated=05:29.50 pm, Tue 11/25/2008
objectClass=top|person|organizationalPerson|user|computer

Event Details:
  uSNChanged=2921916
  uSNCreated=1679623
  instanceType=4

Additional Details:
  isCriticalSystemObject=FALSE
  servicePrincipalName=TERMSRV/stuff2|TERMSRV blah
  dnsHostName=stuff2.splunk.com
  operatingSystemServicePack=Service Pack 2
  operatingSystemVersion=6.0 (6002)
  operatingSystem=Windows Vista? Ultimate
localPolicyFlags=0

```

Delete event

When an AD object has been marked for deletion, the Splunk platform generates a delete event. The event type is similar to `admonEventType=Update`, except that it contains the `isDeleted=True` key/value pair at the end of the event.

```

2/1/10
3:11:16.095 PM

02/01/2010 15:11:16.0954
dcName=stuff.splunk.com
admonEventType=Update
Names:
  name=SplunkTest
DEL:blah
  distinguishedName=OU=SplunkTest\0ADEL:blah,CN=Deleted Objects
DEL:blah
Object Details:
  objectGUID=blah
  whenChanged=11:31.13 pm, Thu 01/28/2010
  whenCreated=11:27.12 pm, Thu 01/28/2010
  objectClass=top|organizationalUnit

Event Details:
  uSNChanged=2922895
  uSNCreated=2922846
  instanceType=4

Additional Details:
  dScorePropagationData=20100128233113.0Z|20100128233113.0Z|20100128233113.0Z|16
010108151056.0Z
  lastKnownParent=stuff
  '''isDeleted=TRUE'''

```

Sync event

When AD monitoring inputs are configured, the Splunk platform tries to capture a baseline of AD metadata when it starts. the Splunk platform generates event type `admonEventType=Sync`, which represents the instance of one AD object and all its field values. the Splunk platform tries to capture all of the objects from the last recorded Update Sequence Number (USN).

When you restart Splunk Enterprise or the `splunk-admon.exe` process, the software logs an extra `sync` event. This is normal.

```
2/1/10
3:11:09.074 PM

02/01/2010 15:11:09.0748
dcName=ftw.ad.splunk.com
admonEventType=Sync
Names:
    name=NTDS Settings
    distinguishedName=CN=NTDS
Settings,CN=stuff,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration
    cn=NTDS Settings
    objectCategory=CN=NTDS-DSA,CN=Schema,CN=Configuration,DC=ad,DC=splunk,DC=com
    fullPath=LDAP://stuff.splunk.com/<GUID=bla bla bla>
    CN=NTDS Settings
Object Details:
    whenCreated=10:15.04 pm, Tue 02/12/2008
    whenChanged=10:23.00 pm, Tue 02/12/2008
    objectGUID=bla bla bla
    objectClass=top|applicationSettings|nTDSDSA
    classPath=nTDSDSA
Event Details:
    instanceType=4
Additional Details:
    systemFlags=33554432
    showInAdvancedViewOnly=TRUE
    serverReferenceBL=CN=stuff,CN=Domain System Volume (SYSVOL share),CN=File Replication
Service,CN=System
    options=1
    msDS
-hasMasterNCs=DC=ForestDnsZones|DC=DomainDnsZones|CN=Schema,CN=Configuration|CN=Configuration
    msDS-HasInstantiatedNCs=
    msDS-HasDomainNCs=blah
    msDS-Behavior-Version=2
    invocationId=bla bla bla
    hasMasterNCs=CN=Schema,CN=Configuration|CN=Configuration
    dScorePropagationData=
    dMDLocation=CN=Schema,CN=Configuration
    nTSecurityDescriptor=NT AUTHORITY\Authenticated Users
SchemaName=LDAP://stuff.splunk.com/schema/nTDSDSA
```

Schema event

When you restart Splunk Enterprise after configuring it for AD monitoring, it generates a schema type event: `admonEventType=schema`. This event shows the definitions of every object in the Active Directory structure. The `available`, `required`, and `optional` fields are listed for each AD object. Failure to see all of these fields can indicate a problem with Active Directory.

```
02/01/2010 15:11:16.0518
```

dcName=LDAP://stuff.splunk.com/
admonEventType=schema
className=msExchProtocolCfgSMTPIPAddress
classCN=ms-Exch-Protocol-Cfg-SMTP-IP-Address
instanceType=MandatoryProperties
nTSecurityDescriptor=MandatoryProperties
objectCategory=MandatoryProperties
objectClass=MandatoryProperties
adminDescription=OptionalProperties
adminDisplayName=OptionalProperties
allowedAttributes=OptionalProperties
allowedAttributesEffective=OptionalProperties
allowedChildClasses=OptionalProperties
allowedChildClassesEffective=OptionalProperties
bridgeheadServerListBL=OptionalProperties
canonicalName=OptionalProperties
cn=OptionalProperties
createTimeStamp=OptionalProperties
description=OptionalProperties
directReports=OptionalProperties
displayName=OptionalProperties
displayNamePrintable=OptionalProperties
distinguishedName=OptionalProperties
dSASignature=OptionalProperties
dSCorePropagationData=OptionalProperties
extensionName=OptionalProperties
flags=OptionalProperties
fromEntry=OptionalProperties
frsComputerReferenceBL=OptionalProperties
frsMemberReferenceBL=OptionalProperties
fsmoRoleOwner=OptionalProperties
heuristics=OptionalProperties
isCriticalSystemObject=OptionalProperties
isDeleted=OptionalProperties
isPrivilegeHolder=OptionalProperties
lastKnownParent=OptionalProperties
legacyExchangeDN=OptionalProperties
managedObjects=OptionalProperties
masteredBy=OptionalProperties
memberOf=OptionalProperties
modifyTimeStamp=OptionalProperties
ms-DS-ConsistencyChildCount=OptionalProperties
ms-DS-ConsistencyGuid=OptionalProperties
msCOM-PartitionSetLink=OptionalProperties
msCOM-UserLink=OptionalProperties
msDFSR-ComputerReferenceBL=OptionalProperties
msDFSR-MemberReferenceBL=OptionalProperties
msDS-Approx-Immed-Subordinates=OptionalProperties
msDs-masteredBy=OptionalProperties
msDS-MembersForAzRoleBL=OptionalProperties
msDS-NCReplCursors=OptionalProperties
msDS-NCReplInboundNeighbors=OptionalProperties
msDS-NCReplOutboundNeighbors=OptionalProperties
msDS-NonMembersBL=OptionalProperties
msDS-ObjectReferenceBL=OptionalProperties
msDS-OperationsForAzRoleBL=OptionalProperties
msDS-OperationsForAzTaskBL=OptionalProperties
msDS-ReplAttributeMetaData=OptionalProperties
msDS-ReplValueMetaData=OptionalProperties
msDS-TasksForAzRoleBL=OptionalProperties
msDS-TasksForAzTaskBL=OptionalProperties
msExchADCGlobalNames=OptionalProperties

msExchALObjectVersion=OptionalProperties
msExchHideFromAddressLists=OptionalProperties
msExchInconsistentState=OptionalProperties
msExchIPAddress=OptionalProperties
msExchTurfList=OptionalProperties
msExchUnmergedAttsPt=OptionalProperties
msExchVersion=OptionalProperties
msSFU30PosixMemberOf=OptionalProperties
name=OptionalProperties
netbootSCPBL=OptionalProperties
nonSecurityMemberBL=OptionalProperties
objectGUID=OptionalProperties
objectVersion=OptionalProperties
otherWellKnownObjects=OptionalProperties
ownerBL=OptionalProperties
partialAttributeDeletionList=OptionalProperties
partialAttributeSet=OptionalProperties
possibleInferiors=OptionalProperties
proxiedObjectName=OptionalProperties
proxyAddresses=OptionalProperties
queryPolicyBL=OptionalProperties
replicatedObjectVersion=OptionalProperties
replicationSignature=OptionalProperties
replPropertyMetaData=OptionalProperties
replUpToDateVector=OptionalProperties
repsFrom=OptionalProperties
repsTo=OptionalProperties
revision=OptionalProperties
sDRightsEffective=OptionalProperties
serverReferenceBL=OptionalProperties
showInAddressBook=OptionalProperties
showInAdvancedViewOnly=OptionalProperties
siteObjectBL=OptionalProperties
structuralObjectClass=OptionalProperties
subRefs=OptionalProperties
subSchemaSubEntry=OptionalProperties
systemFlags=OptionalProperties
unmergedAtts=OptionalProperties
url=OptionalProperties
uSNChanged=OptionalProperties
uSNCreated=OptionalProperties
uSNSALastObjRemoved=OptionalProperties
USNIntersite=OptionalProperties
uSNLastObjRem=OptionalProperties
uSNSource=OptionalProperties
wbemPath=OptionalProperties
wellKnownObjects=OptionalProperties
whenChanged=OptionalProperties
whenCreated=OptionalProperties
WWWHomePage=OptionalProperties

Monitor Windows event log data with Splunk Enterprise

Windows generates log data during the course of its operations. The Windows Event Log service handles nearly all of this communication. It gathers log data that installed applications, services, and system processes publish and places the log data into event log channels. Programs such as Microsoft Event Viewer subscribe to these log channels to display events that have occurred on the system.

You can monitor event log channels and files that are on the local machine or you can collect logs from remote machines. The event log monitor runs once for every event log input that you define.

To monitor Windows Event Log channels in Splunk Cloud Platform, use a Splunk universal or heavy forwarder to collect the data and forward it to your Splunk Cloud Platform deployment. As a best practice, use the Splunk Add-on for Windows to simplify the process of getting data into Splunk Cloud Platform. For instructions on using the Splunk Add-on for Windows to get data into Splunk Cloud Platform, see *Get Windows Data Into Splunk Cloud* in the *Splunk Cloud Admin Manual*.

Why monitor event logs?

Windows event logs are the core metric of Windows machine operations. If there is a problem with your Windows system, the Event Log service has logged it. The Splunk platform indexing, searching, and reporting capabilities make your logs accessible.

Requirements for monitoring event logs

Activity	Requirements
Monitor local event logs	<ul style="list-style-type: none">• The Splunk universal forwarder or Splunk Enterprise instance must run on Windows. See <i>Install on Windows</i> in the <i>Installation Manual</i>.• The Splunk universal forwarder or Splunk Enterprise instance must run as the Local System Windows user to read all local event logs.
Monitor remote event logs	<ul style="list-style-type: none">• The universal forwarder or heavy forwarder must run on the Windows machine from which you want to collect event logs.• The Splunk universal forwarder or heavy forwarder must run as a domain or remote user with read access to Windows Management Instrumentation (WMI) on the remote machine. See <i>Choose the Windows user Splunk Enterprise should run as</i> in the <i>Installation Manual</i>.• The user that the forwarder runs as must have read access to the event logs you want to collect.

Security and other considerations for collecting event log data from remote machines

You collect event log data from remote machines using a universal forwarder, a heavy forwarder, or WMI. As a best practice, use a universal forwarder to send event log data from remote machines to an indexer. See *The universal forwarder* in the *Universal Forwarder* manual for information about how to install, configure and use the forwarder to collect event log data. If you can't install a forwarder on the machine where you want to get data, you can use a WMI.

To install forwarders on your remote machines to collect event log data, install the forwarder as the Local System user on these machines. The Local System user has access to all data on the local machine, but not on remote machines.

To use WMI to get event log data from remote machines, you must ensure that your network and Splunk Enterprise instances are properly configured. Do not install Splunk software as the Local System user. The user you use to install the software determines the event logs that Splunk software has access to. See [Security and remote access considerations](#) for additional information on the requirements you must satisfy to collect remote data properly using WMI.

By default, Windows restricts access to some event logs depending on the version of Windows you run. For example, only members of the local Administrators or global Domain Admins groups can read the Security event logs by default.

How the Windows Event Log monitor interacts with Active Directory

When you set up an Event Log monitoring input for WMI, the input connects to an Active Directory (AD) domain controller to authenticate and, if necessary, performs any security ID (SID) translations before it begins to monitor the data.

The Event Log monitor uses the following logic to interact with AD after you set it up:

1. If you specify a domain controller when you define the input with the `evt_dc_name` setting in the `inputs.conf` file, then the input uses that domain controller for AD operations.
2. If you do not specify a domain controller, then the input does the following:
 1. The input attempts to use the local system cache to authenticate or resolve SIDs.
 2. If the monitor cannot authenticate or resolve SIDs that way, it attempts a connection to the domain controller that the machine that runs the input used to log in.
 3. If that does not work, then the input attempts to use the closest AD domain controller that has a copy of the Global Catalog.
3. If the domain controller that you specify is not valid, or a domain controller cannot be found, then the input generates an error message.

Collect event logs from a remote Windows machine

You have two choices to collect data from a remote Windows machine:

- Use a universal forwarder
- Use WMI

Use a universal or heavy forwarder

You can install a universal forwarder or a heavy forwarder on the Windows machine and instruct it to collect event logs. You can do this manually or use a deployment server to manage the forwarder configuration.

1. On the Windows machine for which you want to collect Windows Event Logs, download Splunk Enterprise or the universal forwarder software.
2. Run the universal forwarder installation package to begin the installation process.
3. When the installer prompts you, configure a receiving indexer.
4. When the installer prompts you to specify inputs, enable the event log inputs by checking the **Event logs** checkbox.
5. Complete the installation procedure.
6. On the receiving indexer, use Splunk Web to search for the event log data as in the following example:

```
host=<name of remote Windows machine> sourcetype=Wineventlog
```

For specific instructions to install the universal forwarder, see *Install a Windows universal forwarder in the Forwarder Manual*.

Use WMI

If you want to collect event logs remotely using WMI, you must install the universal or heavy forwarder to run as an Active Directory domain user. If the selected domain user is not a member of the Administrators or Domain Admins groups, then you must configure event log security to give the domain user access to the event logs.

To change event log security to get access to the event logs from remote machines, you must meet the following requirements:

- Have administrator access to the machine from which you are collecting event logs.
- Understand how the Security Description Definition Language (SDDL) works and how to assign permissions with it. See [http://msdn.microsoft.com/en-us/library/aa379567\(v=VS.85\).aspx](http://msdn.microsoft.com/en-us/library/aa379567(v=VS.85).aspx) on the Microsoft website for more information.
- Decide how to monitor your data. See [Considerations for deciding how to monitor remote Windows data](#) for information on collecting data from remote Windows machines.

You can use the wevtutil utility to set event log security.

1. Download a Splunk Enterprise instance onto a Windows machine.
2. Double-click the installer file to begin the installation.
3. When the installer prompts you to specify a user, select **Domain user**.
4. On the next installer pane, enter the domain user name and password that you want Splunk Enterprise to use when it runs.
5. Follow the prompts to complete the installation of the software.
6. Once the software has installed, log in to the instance.
7. Use Splunk Web to add the remote event log input. See [Configure remote event log monitoring](#) later in this topic.

Anomalous machine names are visible in event logs on some systems

On some Windows systems, you might see some event logs with randomly-generated machine names. This is the result of those systems logging events before the user has named the system during the OS installation process.

This anomaly occurs only when you collect logs from versions of Windows remotely over WMI.

Configure local event log monitoring with Splunk Web

To get local Windows event log data, point your Splunk Enterprise instance at the Event Log service.

Go to the Add Data page

You can get there in two ways:

- Splunk Settings
- Splunk Home

From Splunk Settings:

1. Click **Settings > Data Inputs**.
2. Click **Local event log collection**.
3. Click **New** to add an input.

From Splunk Home:

1. Click the **Add Data** link in Splunk Home.
2. Click **Monitor** to monitor Event Log data on the local Windows machine, or **Forward** to forward Event Log data from another Windows machine.
Splunk Enterprise loads the Add Data - Select Source page.
3. If you selected **Forward**, select or create the group of forwarders you want this input to apply to. See [Forward data](#) in this manual.
4. Click **Next**.

Select the input source

1. Select **Local Event Logs**
2. In the **Select Event Logs** list, select the Event Log channels you want this input to monitor.
3. Click each Event Log channel you want to monitor once.
Splunk Enterprise moves the channel from the **Available items** window to the **Selected items** window.

4. To deselect a channel, click its name in the **Available Items** window.
Splunk Enterprise moves the channel from the **Selected items** window to the **Available items** window.
5. To select or deselect all of the event logs, click the **add all** or **remove all** links.

Selecting all of the channels can result in the indexing of a lot of data.

6. Click **Next**.

Specify input settings

The **Input Settings** page lets you specify the application context, default host value, and index. All of these parameters are optional.

The **Host** field sets only the **host** field in the resulting events. It doesn't direct Splunk Enterprise to look on a specific machine on your network.

1. Select the appropriate **Application context** for this input.
2. Set the **Host** value. You have several choices for this setting. For more information about setting the host value, see [About hosts](#).
3. Set the **Index** that you want Splunk Enterprise to send data to. Leave the value as **default**, unless you defined multiple indexes to handle different types of events. In addition to indexes for user data, Splunk Enterprise has a number of utility indexes, which also appear in this dropdown box.
4. Click **Review**.

Review your choices

After you specify all your input settings, you can review your selections. Splunk Enterprise lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.
2. If they do not match what you want, click the left angle bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

Splunk Enterprise then displays the "Success" page and begins indexing the specified Event Log channels.

Configure remote event log monitoring with Splunk Web

The process for configuring remote event log monitoring is nearly identical to the process for monitoring local event logs.

1. Follow the instructions to get to the Add Data page. See [Go to the Add Data page](#).
2. Locate and select **Remote Event Logs**.
3. In the **Event Log collection name** field, enter a unique, memorable name for this input.
4. In the **Choose logs from this host** field, enter the host name or IP address of the machine that contains the Event Log channels you want to monitor.

Selecting all of the Event Log channels can result in the indexing of a lot of data.

5. Click the **Find logs** button to refresh the page with a list of available Event Log channels on the machine you entered.
6. Click once on each Event Log channel you want to monitor.
Splunk Enterprise moves the channel from the **Available items** window to the **Selected items** window.
7. To deselect a channel, click its name in the **Available Items** window.
Splunk Enterprise moves the channel from the **Selected items** window to the **Available items** window.
8. To select or deselect all of the event logs, click the **add all** or **remove all** links.

9. In the **Collect the same set of logs from additional hosts** field, enter the host names or IP addresses of additional machines that contain the Event Logs you selected previously. Separate multiple machines with commas.
10. Click the green **Next** button.
11. Follow the instructions to specify input settings. See [Specify input settings](#).
12. Follow the instructions to review your choices. See [Review your choices](#).

Use the inputs.conf configuration file to configure event log monitoring

On either a universal or heavy forwarder, you can edit the inputs.conf configuration file to configure Windows event log monitoring.

1. Using Notepad or a similar editor, open %SPLUNK_HOME%\etc\system\local\inputs.conf for editing. You might need to create this file if it doesn't exist.
2. Enable Windows event log inputs by adding input stanzas that reference Event Log channels.
3. Save the file and close it.
4. Restart the Splunk platform.

For more information on configuring data inputs with the inputs.conf file, see [Edit inputs.conf](#).

Specify global settings for Windows Event Log inputs

When you define Windows Event Log inputs in inputs.conf, make sure you explicitly specify global settings in the correct place.

If you specify global settings for Windows Event Log inputs, such as `host`, `sourcetype`, and so on, you can place those settings in one of the following areas:

- Under the `[WinEventLog]` global stanza. This stanza is equal to the `[default]` stanza for other monitoring inputs. For example:

```
[default]
_meta = hf_proxy::meta_test

[WinEventLog]
_meta = hf_proxy::meta_test
host = WIN2K16_DC
index = wineventlog

[WinEventLog://Application]
disabled = 0
```

- Under the Windows Event Log input stanza for the Event Log channel that you want to monitor. For example:

```
[default]
_meta = hf_proxy::meta_test

[WinEventLog]
host = WIN2K16_DC
index = wineventlog

[WinEventLog://Application]
disabled = 0
_meta = hf_proxy::meta_test
```

You can review the defaults for a configuration file by looking at the examples in %SPLUNK_HOME%\etc\system\default or at the spec file in the *Admin Manual*.

Event log monitor configuration values

Windows event log (*.evt) files are in binary format. You can't monitor them like you do a normal text file. The `splunkd` service monitors these binary files by using the appropriate APIs to read and index the data within the files.

Splunk Enterprise uses the following stanzas in `inputs.conf` to monitor the default Windows event logs:

```
# Windows platform specific input processor.
[WinEventLog://Application]
disabled = 0
[WinEventLog://Security]
disabled = 0
[WinEventLog://System]
disabled = 0
```

Monitor non-default Windows event logs

You can also configure Splunk Enterprise to monitor non-default Windows event logs. Before you can do this, you must import them to the Windows Event Viewer. After you import the logs, you can add them to your local copy of `inputs.conf`, as in the following example:

```
[WinEventLog://DNS Server]
disabled = 0
[WinEventLog://Directory Service]
disabled = 0
[WinEventLog://File Replication Service]
disabled = 0
```

Use the Full Name log property in Event Viewer to specify complex Event Log channel names properly

You can use the `Full Name` Event Log property in Event Viewer to ensure that you specify the correct Event Log channel in an `inputs.conf` stanza.

For example, to monitor the Task Scheduler application log, `Microsoft-Windows-TaskScheduler-Operational`, do the following steps:

1. Open Event Viewer.
2. Expand **Applications and Services Logs > Microsoft > Windows > TaskScheduler**.
3. Right-click **Operational** and select **Properties**.
4. In the dialog that appears, copy the text in the **Full Name** field.
5. Append this text into the `WinEventLog://` stanza of `inputs.conf` as in the following example:

```
[WinEventLog://Microsoft-Windows-TaskScheduler/Operational]
disabled = 0
```

Disable an event log stanza

To disable indexing for an event log, add `disabled = 1` after its listing in the stanza in %SPLUNK_HOME%\etc\system\local\inputs.conf.

Configuration settings for monitoring Windows Event Logs

Splunk software uses the following settings in inputs.conf to monitor Event Log files:

Attribute	Description	Default
start_from	<p>How to read events.</p> <p>Acceptable values are <code>oldest</code>, meaning read logs from the oldest to the newest, and <code>newest</code>, meaning read logs from the newest to the oldest.</p> <p>You can't set this attribute to <code>newest</code> while also setting the <code>current_only</code> attribute to <code>1</code>.</p>	oldest
current_only	<p>How to index events.</p> <p>Acceptable values are <code>1</code>, where the input acquires events that arrive after the input starts for the first time, like <code>tail -f</code> on *nix systems, or <code>0</code>, where the input gets all existing events in the log and then continues to monitor incoming events in real time.</p> <p>You can't set this attribute to <code>newest</code> while also setting the <code>current_only</code> attribute to <code>1</code>.</p>	0
checkpointInterval	<p>How frequently, in seconds, the Windows Event Log input saves a checkpoint.</p> <p>Checkpoints store the eventID of acquired events to enable Splunk software to resume monitoring at the correct event after a shutdown or outage.</p>	0
evt_resolve_ad_ds	<p>The domain controller Splunk software uses to interact with Active Directory while indexing Windows Event Log channels. Valid only when you set the <code>evt_resolve_ad_obj</code> attribute to <code>1</code> and omit the <code>evt_dc_name</code> attribute.</p> <p>Valid values are <code>auto</code>, meaning to use the nearest domain controller to bind to for AD object resolution, or <code>PDC</code>, meaning to bind to the primary domain controller for the AD site that the host is in. If you also set the <code>evt_dc_name</code> attribute, Splunk software ignores this attribute.</p>	auto
evt_resolve_ad_obj	<p>How Splunk software interacts with Active Directory while indexing Windows Event Log channels. Valid values are <code>1</code>, meaning resolve Active Directory objects like Globally Unique Identifier (GUID) and Security Identifier (SID) objects to their canonical names for a specific Windows event log channel, and <code>0</code>, meaning not to attempt any resolution.</p> <p>When you set this value to <code>1</code>, you can optionally specify the Domain Controller name or DNS name of the domain to bind to, which Splunk software uses to resolve the AD objects. If you don't set this value, or if you set it to <code>0</code>, Splunk software does not attempt to resolve the AD objects.</p>	0
evt_dc_name	<p>Which Active Directory domain controller to bind to resolve AD objects. This name can be the NetBIOS name of the domain controller, the fully-qualified DNS name of the domain controller, or an environment variable name specified as <code>\$Environment_variable</code>.</p>	N/A

Attribute	Description	Default
	<p>If you set this attribute, then Splunk software ignores the <code>evt_resolve_ad_ds</code> attribute, which controls how the software determines the best domain controller to bind to for AD object resolution.</p> <p>If you specify an environment variable, you must prepend a dollar sign (\$) to the environment variable name. Splunk software uses the specified environment variable as the domain controller to connect to for AD object resolution. For example, to use the <code>%LOGONSERVER%</code> variable, specify <code>evt_dc_name = \$logonserver</code>.</p> <p>You can precede either format with two backslash characters. This attribute does not have a default.</p>	
<code>evt_dns_name</code>	The fully-qualified DNS name of the domain to bind to resolve AD objects.	N/A
<code>evt_exclude_fields</code>	A list of Windows Event Log fields that the Windows Event Log input is to exclude when it ingests Windows Event Log data. When you specify this setting, the input removes both the key and value data for the fields you exclude. This setting works similar to the <code>suppress_*</code> settings, but unlike those settings, this setting is valid for all Windows Event Log fields, and excludes fields that you might have included in an allow list. When this collision happens, the instance logs an error. See "Create advanced filters with 'whitelist' and 'blacklist'" later in this topic for the list of Windows Event Log fields that you can exclude.	N/A
<code>suppress_text</code>	Whether to include the message text that comes with a security event. A value of 1 suppresses the message text, and a value of 0 preserves the text.	0
<code>use_old_eventlog_api</code>	<p>Whether or not to read Event Log events with the Event Logging API.</p> <p>This is an advanced setting. Contact Splunk Support before you change it.</p> <p>If set to true, the input uses the Event Logging API instead of the Windows Event Log API to read from the Event Log on Windows Server 2008, Windows Vista, and higher installations.</p>	false (Use the API that is specific to the OS.)
<code>use_threads</code>	<p>Specifies the number of threads, in addition to the default writer thread, that can be created to filter events with the allow list/deny list regular expression.</p> <p>This is an advanced setting. Contact Splunk Support before you change it.</p> <p>The maximum number of threads is 15.</p>	0
<code>thread_wait_time_msec</code>	<p>The interval, in milliseconds, between attempts to re-read Event Log files when a read error occurs.</p> <p>This is an advanced setting. Contact Splunk Support before you change it.</p>	5000
<code>suppress_checkpoint</code>	<p>Whether or not the Event Log strictly follows the <code>checkpointInterval</code> setting when it saves a checkpoint.</p> <p>This is an advanced setting. Contact Splunk Support before you change it.</p>	false

Attribute	Description	Default
	By default, the Event Log input saves a checkpoint from between zero and <code>checkpointInterval</code> seconds, depending on incoming event volume.	
<code>suppress_sourcename</code>	<p>Whether or not to exclude the <code>sourcename</code> field from events.</p> <p>This is an advanced setting. Contact Splunk Support before you change it.</p> <p>When set to true, the input excludes the <code>sourcename</code> field from events and throughput performance (the number of events processed per second) improves.</p>	false
<code>suppress_keywords</code>	<p>Whether or not to exclude the <code>keywords</code> field from events.</p> <p>This is an advanced setting. Contact Splunk Support before you change it.</p> <p>When set to true, the input excludes the <code>keywords</code> field from events and throughput performance (the number of events processed per second) improves.</p>	false
<code>suppress_type</code>	<p>Whether or not to exclude the <code>type</code> field from events.</p> <p>This is an advanced setting. Contact Splunk Support before you change it.</p> <p>When set to true, the input excludes the <code>type</code> field from events and throughput performance (the number of events processed per second) improves.</p>	false
<code>suppress_task</code>	<p>Whether or not to exclude the <code>task</code> field from events.</p> <p>This is an advanced setting. Contact Splunk Support before you change it.</p> <p>When set to true, the input excludes the <code>task</code> field from events and throughput performance (the number of events processed per second) improves.</p>	false
<code>suppress_opcode</code>	<p>Whether or not to exclude the <code>opcode</code> field from events.</p> <p>This is an advanced setting. Contact Splunk Support before you change it.</p> <p>When set to true, the input excludes the <code>opcode</code> field from events and throughput performance (the number of events processed per second) improves.</p>	false
<code>whitelist</code>	<p>Whether to index events that match the specified text string. This attribute is optional.</p> <p>You can specify one of two formats:</p> <ul style="list-style-type: none"> • One or more Event Log event codes or event IDs (Event Code/ID format.) • One or more sets of keys and regular expressions (Advanced filtering format.) <p>You cannot mix formats in a single entry. You also cannot mix formats in the same stanza.</p> <p>Allow lists are processed first, then deny lists. If no allow list is present, the Splunk platform indexes all events. If a file matches the regexes in both the deny list and allow list settings, the file is NOT monitored. Deny lists take precedence over</p>	N/A

Attribute	Description	Default
	<p>allow lists.</p> <p>When you use the Event Code/ID format:</p> <ul style="list-style-type: none"> • For multiple codes/IDs, separate the list with commas. • For ranges, use hyphens (for example "0-1000,5000-1000"). <p>When using the advanced filtering format:</p> <ul style="list-style-type: none"> • Use = between the key and the regular expression that represents your filter (for example <code>whitelist = EventCode=%^1([8-9])\$%</code> • You can have multiple key/regular expression sets in a single advanced filtering entry. The Splunk platform conjuncts the sets logically. This means that the entry is valid only if all of the sets in the entry are true. • You can specify up to 10 whitelists per stanza by adding a number to the end of the <code>whitelist</code> attribute, for example <code>whitelist1...whitelist9</code>. 	
<code>blacklist</code>	<p>Do not index events that match the text string specified. This attribute is optional.</p> <p>You can specify one of two formats:</p> <ul style="list-style-type: none"> • One or more Event Log event codes or event IDs (Event Log code/ID format.) • One or more sets of keys and regular expressions. (Advanced filtering format.) <p>You cannot mix formats in a single entry. You also cannot mix formats in the same stanza.</p> <p>Allow lists are processed first, then deny lists. If no deny list is present, the Splunk platform indexes all events.</p> <p>When using the Event Log code/ID format:</p> <ul style="list-style-type: none"> • For multiple codes/IDs, separate the list with commas. • For ranges, use hyphens (for example <code>0-1000, 5000-1000</code>). <p>When using the advanced filtering format:</p> <ul style="list-style-type: none"> • Use = between the key and the regular expression that represents your filter (for example <code>blacklist = EventCode=%^1([8-9])\$%</code> • You can have multiple key/regular expression sets in a single advanced filtering entry. The Splunk platform conjuncts the sets logically. This means that the entry is valid only if all of the sets in the entry are true. • You can specify up to 10 deny lists per stanza by adding a number to the end of the <code>blacklist</code> attribute, for example <code>blacklist1...blacklist9</code>. 	
<code>renderXml</code>	<p>Render event data as XML supplied by the Windows Event Log subsystem. This setting is optional.</p> <p>A value of <code>1</code> or <code>true</code> means to render the events as XML. A value of <code>0</code> or <code>false</code> means to render the events as plain text.</p> <div> <p>If you set <code>renderXml</code> to <code>true</code>, if you want to also create allow lists or deny lists to filter event data, you must use the <code>\$XmlRegex</code> special key in your allow lists or deny lists.</p> </div>	

0 (false)`index`The index that this input is to send the data to.the default index`disabled`

Whether or not the input is to run.

Valid values are 0, meaning that the input is to run, and 1, meaning that the input is to not run.

0

Use the Security event log to monitor changes to files

You can monitor changes to files on your system by enabling security auditing on a set of files or directories and then monitoring the Security event log channel for change events. The event log monitoring input includes three attributes which you can use in inputs.conf. For example:

```
[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
checkpointInterval = 5
# only index events with these event IDs.
whitelist = 0-2000,3001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000
```

To enable security auditing for a set of files or directories, read Auditing Security Events How To on MS Technet at <http://technet.microsoft.com/en-us/library/cc727935%28v=ws.10%29.aspx>.

You can also use the `suppress_text` attribute to include or exclude the message text that comes with a security event.

When you set `suppress_text` to 1 in a Windows Event Log Security stanza, the entire message text does not get indexed, including any contextual information about the security event. If you need this contextual information, do not set `suppress_text` in the stanza.

See the following example to include or exclude message text:

```
[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
checkpointInterval = 5
# suppress message text, we only want the event number.
suppress_text = 1
# only index events with these event IDs.
whitelist = 0-2000,2001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000
```

To use a specific domain controller, set the `evt_dc_name` attribute:

```
[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
evt_dc_name = boston-dc1.contoso.com
```

```

checkpointInterval = 5
# suppress message text, we only want the event number.
suppress_text = 1
# only index events with these event IDs.
whitelist = 0-2000,2001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000

```

To use the primary domain controller to resolve AD objects, set the `evt_resolve_ad_ds` attribute to `PDC`. Otherwise, it locates the nearest domain controller.

```

[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
evt_resolve_ad_ds = PDC
checkpointInterval = 5
# suppress message text, we only want the event number.
suppress_text = 1
# only index events with these event IDs.
whitelist = 0-2000,2001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000

```

Create advanced filters with whitelist and blacklist

You can perform advanced filtering of incoming events with the `whitelist` and `blacklist` settings in addition to filtering based solely on event codes. To do this, specify the key/regular expression format in the setting:

```
whitelist = key=<regular expression> [key=<regular expression>] ...
```

In this format, `key` must be a valid entry from the following table:

Key	Description
\$TimeGenerated	The time that the computer generated the event. Splunk Enterprise only generates the time string as the event.
\$Timestamp	The time that the event was received and recorded by the Event Log service. Splunk Enterprise only generates the time string as the event.
\$XmlRegex	A special key that configures Splunk Enterprise to filter on XML events. To use this key, set it to the value that you want Splunk Enterprise to filter on. You must configure the input for which to apply a deny list or allow list to render events in XML. To generate XML events, specify the <code>renderXml = true</code> setting under the input stanza. Splunk Enterprise conjuncts multiple entries in a single allow list or deny list line. All of the filter entries must match for the filter to trigger.
Category	The category number for a specific event source.
CategoryString	A string translation of the category. The translation depends on the event source.
ComputerName	The name of the computer that generated the event.
EventCode	The event ID number for an event. Corresponds to Event ID in Event Viewer.
EventType	A numeric value that represents one of the five types of events that can be logged: Error, Warning, Information, Success Audit, and Failure Audit. Available only on machines that run Windows Server 2003 and lower or clients running Windows XP and lower. See Win32_NTLogEvent class (Windows) on MSDN at http://msdn.microsoft.com/en-us/library/aa394226(v=vs.85).aspx .
Keywords	An element used to classify different types of events within an event log channel. The Security Event Log channel has this element, for example.
LogName	The name of the Event Log channel that received the event. Corresponds to Log Name in Event Viewer.

Key	Description
Message	The text of the message in the event.
OpCode	The severity level of the event. Corresponds to OpCode in Event Viewer.
RecordNumber	The Windows Event Log record number. Each event on a Windows machine gets a record number. This number starts at 0 with the first event generated on the system, and increases with each new event generated, until it reaches a maximum of 4294967295. It then rolls back over to 0.
Sid	The Security Identifier (SID) of the principal, such as a user, group, computer, or other entity, that was associated with or generated the event. See Win32_UserAccount class on MSDN at http://msdn.microsoft.com/en-us/library/windows/desktop/aa394507%28v=vs.85%29.aspx .
SidType	A numeric value that represents the type of SID that was associated with the event. See Win32_UserAccount class on MSDN at http://msdn.microsoft.com/en-us/library/windows/desktop/aa394507%28v=vs.85%29.aspx .
SourceName	The source of the entity that generated the event. Corresponds to Source in Event Viewer.
TaskCategory	The task category of the event. Event sources let you define categories so that you can filter them with Event Viewer using the <code>Task Category</code> field. See Event Categories (Windows) on MSDN at http://msdn.microsoft.com/en-us/library/aa363649%28VS.85%29.aspx .
Type	A numeric value that represents one of the five types of events that can be logged: Error, Warning, Information, Success Audit, and Failure Audit. Only available on machines that run Windows Server 2008 or higher, or Windows Vista or higher. See Win32_NTLogEvent class (Windows) on MSDN at http://msdn.microsoft.com/en-us/library/aa394226(v=vs.85).aspx .
User	The user associated with the event. Correlates to User in Event Viewer.

`<regular expression>` is any valid regular expression that represents the filters that you want to include, when used with the `whitelist` attribute, or exclude, when used with the `blacklist` attribute.

You can specify more than one key/regular expression set on a single entry line. When you do this, Splunk Enterprise logically conjuncts the sets. This means that only events that satisfy all of the sets on the line are valid for inclusion or exclusion. See the following example:

```
whitelist = EventCode="^1([0-5])$" Message="^Error"
```

means to include events that have an `EventCode` ranging from 10 to 15 and contain a `Message` that begins with the word `Error`.

You can specify up to 10 separate allow list or deny list entries in each stanza. To do so, add a number at the end of the `whitelist` or `blacklist` entry on a separate line:

```
whitelist = key=<regular expression>
whitelist1 = key=<regular expression> key2=<regular expression 2>
whitelist2 = key=<regular expression>
```

You cannot specify an entry that has more than one key/regular expression set that references the same key. If, for example, you specify: `whitelist = EventCode="^1([0-5])$" EventCode="^2([0-5])"` Splunk Enterprise ignores the first set and only attempts to include events that match the second set. In this case, only events that contain an `EventCode` between 20 and 25 match. Events that contain an `EventCode` between 10 and 15 do not match. Only the last set in the entry ever matches. To resolve this problem, specify two separate entries in the stanza: `whitelist = EventCode="^1([0-5])$" whitelist1 = EventCode="^2([0-5])"`

Suppress fields from Windows Event Log events

There are two options to limit the ingestion of data by removing Windows Event Log fields from events that a Splunk Platform instance ingests:

- Use the `suppress_*` settings in `inputs.conf` to remove certain Windows Event Log fields from ingested events.
- Use the `evt_exclude_fields` setting, which lets you remove any Windows Event Log field from a Windows Event Log event. This setting removes both the excluded key and value from the event, and excludes events even if the field exists in an allow list.

You define both of these settings in the `inputs.conf` configuration file, under a Windows Event Log monitoring input, for example:

`suppress_*` example (suppresses the message text)

```
[WinEventLog://System]
disabled=0
suppress_text=1
evt_exclude_fields example
```

```
[WinEventLog://System]
disabled=0
evt_exclude_fields=EventCode,RecordNumber
```

See the list of fields in "Create advanced filters with 'whitelist' and 'blacklist'" earlier in this topic. See "Configuration settings for monitoring Windows Event Logs", also earlier in this topic, for more information about the settings.

Resolve Active Directory objects in event log files

To specify whether Active Directory objects like globally unique identifiers (GUIDs) and security identifiers (SIDs) are resolved for a given Windows event log channel, use the `evt_resolve_ad_obj` attribute (1=enabled, 0=disabled) for that channel's stanza in your local copy of `inputs.conf`. The `evt_resolve_ad_obj` attribute is on by default for the Security channel.

For example:

```
[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
checkpointInterval = 5
```

To specify a domain controller for the domain that the Splunk platform instance is to bind to in order to resolve AD objects, use the `evt_dc_name` attribute.

The string specified in the `evt_dc_name` attribute can represent either the domain controller NetBIOS name, or its fully-qualified domain name (FQDN). Either name type can, optionally, be preceded by two backslash characters.

The following examples are correctly formatted domain controller names:

- FTW-DC-01
- \\FTW-DC-01
- FTW-DC-01.splunk.com

- \\FTW-DC-01.splunk.com

To specify the FQDN of the domain to bind to, use the `evt_dns_name` attribute.

For example:

```
[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
evt_dc_name = ftw-dc-01.splunk.com
evt_dns_name = splunk.com
checkpointInterval = 5
```

When you use the `evt_resolve_ad_obj` and `evt_dc_name` attributes, the following constraints apply:

- Splunk software first attempts to resolve SIDs and GUIDs using the domain controller (DC) specified in the `evt_dc_name` attribute first. If it cannot resolve SIDs using this DC, it attempts to bind to the default DC to perform the translation.
- If Splunk software cannot contact a DC to translate SIDs, it attempts to use the local machine for translation.
- If none of these methods works, then Splunk prints the SID as it was captured in the event.
- Splunk software cannot translate SIDs that are not in the format
S-1-N-NN-NNNNNNNNNN-NNNNNNNNNN-NNNNNNNNNN-NNNN.

If you discover that SIDs are not being translated properly, review `%SPLUNK_HOME%\var\log\splunkd.log` for clues about what the problem might be.

Specify whether to start index at the earliest or the most recent event

Use the `start_from` attribute to specify whether events are indexed starting at the earliest event or the most recent. By default, indexing starts with the oldest data and moves forward. Do not change this setting, because Splunk software stops indexing after it has indexed the backlog using this method.

Use the `current_only` attribute to specify whether to index all preexisting events in a given log channel. When set to 1, only events that appear from the moment the Splunk deployment was started are indexed. When set to 0, all events are indexed.

For example:

```
[WinEventLog://Application]
disabled = 0
start_from = oldest
current_only = 1
```

Display Windows Event Log events in XML

To have Splunk Enterprise generate Windows Event Log events in XML, use the `renderXml` setting in a Windows Event Log input stanza:

```
[WinEventLog://System]
disabled = 0
renderXml = 1
evt_resolve_ad_obj = 1
```

```
evt_dns_name = \"SV5DC02\"
```

This input stanza generates events like the following:

```
<Event xmlns='http://schemas.microsoft.com/win/2004/08/events/event'>
  <System>
    <Provider Name='Service Control Manager' Guid='{555908d1-a6d7-4695-8e1e-26931d2012f4}'
EventSourceName='Service Control Manager' />
    <EventID Qualifiers='16384'>7036</EventID>
    <Version>0</Version>
    <Level>4</Level>
    <Task>0</Task>
    <Opcode>0</Opcode>
    <Keywords>0x8080000000000000</Keywords>
    <TimeCreated SystemTime='2014-04-24T18:38:37.868683300Z' />
    <EventRecordID>412598</EventRecordID>
    <Correlation />
    <Execution ProcessID='192' ThreadID='210980' />
    <Channel>System</Channel>
    <Computer>SplunkDoc.splunk-docs.local</Computer>
    <Security />
  </System>
  <EventData>
    <Data Name='param1'>Application Experience</Data>
    <Data Name='param2'>stopped</Data>
    <Binary>410065004C006F006F006B00750070005300760063002F0031000000</Binary>
  </EventData>
</Event>
```

When you instruct Splunk Enterprise to render events in XML, event keys within the XML event render in English regardless of the machine system locale. Compare the following events generated on a French version of Windows Server.

Standard event:

```
04/29/2014 02:50:23 PM
LogName=Security
SourceName=Microsoft Windows security auditing.
EventCode=4672
EventType=0
Type=Information
ComputerName=sacreblue
TaskCategory=Ouverture de session sp<ciale
OpCode=Informations
RecordNumber=2746
Keywords=SuccÃ's de l'audit
Message=PrivilÃ'ges sp<ciaux attribu<s Ã la nouvelle ouverture de session.
```

```
Sujet :
      ID de s<curit< :          AUTORITE NT\SystÃ'me
      Nom du compte :          SystÃ'me
      Domaine du compte :      AUTORITE NT
      ID d'ouverture de session :          0x3e7
```

```
PrivilÃ'ges :          SeAssignPrimaryTokenPrivilege
                        SeTcbPrivilege
                        SeSecurityPrivilege
                        SeTakeOwnershipPrivilege
                        SeLoadDriverPrivilege
                        SeBackupPrivilege
                        SeRestorePrivilege
```



```
SeDebugPrivilege
SeAuditPrivilege
SeSystemEnvironmentPrivilege
SeImpersonatePrivilege
```

XML event:

```
<Event xmlns='http://schemas.microsoft.com/win/2004/08/events/event'>
  <System><Provider Name='Microsoft-Windows-Security-Auditing'
Guid='{54849625-5478-4994-A5BA-3E3B0328C30D}' />
    <EventID>4672</EventID>
    <Version>0</Version>
    <Level>0</Level>
    <Task>12548</Task>
    <Opcode>0</Opcode>
    <Keywords>0x8020000000000000</Keywords>
    <TimeCreated SystemTime='2014-04-29T22:15:03.280843700Z' />
    <EventRecordID>2756</EventRecordID>
    <Correlation/><Execution ProcessID='540' ThreadID='372' />
    <Channel>Security</Channel>
    <Computer>sacreblue</Computer>
    <Security/>
  </System>
  <EventData>
    <Data Name='SubjectUserSid'>AUTHORITY \System</Data>
    <Data Name='SubjectUserName'>System</Data>
    <Data Name='SubjectDomainName'>AUTHORITY NT</Data>
    <Data Name='SubjectLogonId'>0x3e7</Data>
    <Data Name='PrivilegeList'>SeAssignPrimaryTokenPrivilege
      SeTcbPrivilege
      SeSecurityPrivilege
      SeTakeOwnershipPrivilege
      SeLoadDriverPrivilege
      SeBackupPrivilege
      SeRestorePrivilege
      SeDebugPrivilege
      SeAuditPrivilege
      SeSystemEnvironmentPrivilege
      SeImpersonatePrivilege</Data>
  </EventData>
</Event>
```

The `Data Name` keys in the XML event render in English despite rendering in the system's native language in the standard event.

Use allow lists and deny lists to filter on XML-based events

If you render events in XML, and you want to use allow lists and deny lists to filter on those events, you must use the special key `$XmlRegex` when you build your allow lists or deny lists.

The allow list or deny list triggers when Splunk Enterprise finds the value that you specify with `$XmlRegex` anywhere in the XML-rendered event. `$XmlRegex` doesn't work if you don't explicitly specify the input to render events in XML with the `renderXml = true` setting.

The `$XmlRegex` setting doesn't search for key-value pairs. It configures Splunk Enterprise to expect incoming events in XML format.

The following example configures the `whitelist` setting to allow XML events. Splunk Enterprise indexes all XML events that contain the word "Error":

```
[WinEventLog://System]
disabled = 0
renderXml = 1
evt_resolve_ad_obj = 1
evt_dns_name = \"SV5DC02\"
whitelist = $XmlRegex='Error'
```

See the [Create advanced filters with whitelist and blacklist](#) section earlier in this topic for additional information and syntax.

Use the CLI to configure event log monitoring

You can use the CLI to configure local event log monitoring. Before you use the CLI, create stanza entries in `inputs.conf` first. See [Use inputs.conf to configure event log monitoring](#) in this topic.

The CLI is not available for remote Event Log collections.

To list all configured Event Log channels on the local machine, enter the following:

```
> splunk list eventlog
```

You can also list a specific channel by specifying its name as in the following example:

```
> splunk list eventlog <ChannelName>
```

To enable an Event Log channel, enter the following:

```
> splunk enable eventlog <ChannelName>
```

To disable a channel, enter the following:

```
> splunk disable eventlog <ChannelName>
```

Index exported event log files

To index exported Windows event log (.evt or .evtx) files, monitor the directory that contains the exported files. See [Monitor files and directories](#).

Do not attempt to monitor an .evt or .evtx file that is open for writing. Windows does not allow read access to these files. Use the event log monitoring feature instead.

Constraints for monitoring Windows Event log files directly

Directly monitoring Windows Event log files have the following constraints:

- As a result of API and log channel processing constraints on Windows XP and Server 2003 systems, imported .evt files from those systems do not contain the `Message` field. This means that the contents of the `Message` field do not appear in your index.

- Splunk Enterprise on Windows XP and Windows Server 2003/2003 R2 cannot index .evtx files that come from systems running Windows Vista and higher or Windows Server 2008/2008 R2 and higher.
- Splunk Enterprise on Windows Vista and higher and Server 2008/2008 R2 and higher can index both .evt and .evtx files.
- If your .evt or .evtx file is not from a standard event log channel, you must make sure that any dynamic link library (DLL) files required by that channel are present on the computer on which you are indexing.
- Splunk Enterprise indexes an .evt or .evtx file in the primary locale and language of the computer that collects the file.
- Files that you export from another machine do not work with the Splunk Web Upload feature. This is because those files contain information that is specific to the machine that generated them. Other machines cannot process the files in their unaltered form.

When producing .evt or .evtx files on one system, and monitoring them on another, it's possible that not all of the fields in each event expand as they would on the system producing the events. This is caused by variations in DLL versions, availability, and APIs. Differences in OS version, language, Service Pack level, and installed third-party DLLs can also have this effect.

Monitor file system changes on Windows

The Splunk platform supports monitoring Windows file system changes through the Security Windows Event Log channel. To monitor file changes, you must enable security auditing for the files and folders you want to monitor for changes and use the Event Log monitor to monitor the Security event log channel. This procedure of monitoring file system changes replaces the deprecated file system change monitor input.

If you use Splunk Cloud Platform and want to monitor Windows file system changes through the Security Event Log channel, use the Splunk universal forwarder to monitor the changes on a Windows machine.

Requirements

You must meet the following requirements to monitor file system changes:

- The Splunk platform must run on Windows. See *Install on Windows* in the *Installation Manual*.
- The Splunk platform must run as the Local System user or as a domain user with specific security policy rights to read the Security event log
- You must enable security auditing for the files or directories you want the Splunk platform to monitor changes to

Use the Security event log to monitor changes to files

You can monitor changes to files on your system by enabling security auditing on a set of files or directories and then monitoring the Security event log channel for change events. The event log monitoring input includes three settings which you can use in the inputs.conf configuration file. You can't configure monitoring of file system change events from Splunk Web.

You can use these settings outside of the context of the Security event log and file system changes. This list of settings is only a subset of the available settings for the inputs.conf file. For additional settings, see [Monitor Windows event log data with Splunk Cloud](#).

The following table describes the configuration settings available for file monitoring in inputs.conf:

Setting	Description	Default
whitelist	<p>Index events that match the text string specified. This setting is optional.</p> <p>You can specify one of two formats:</p> <ul style="list-style-type: none"> • One or more Event Log event codes or event IDs (Event Log code/ID format) • One or more sets of keys and regular expressions (Advanced filtering format) <p>You cannot mix formats in a single entry or mix formats in the same stanza.</p> <p>The Splunk platform processes allow lists first, then deny lists. If no allow list is present, the Splunk platform indexes all events.</p> <p>When using the Event Code/ID format, follow these rules:</p> <ul style="list-style-type: none"> • For multiple codes/IDs, separate the list with commas. • For ranges, use hyphens (for example "0-1000,5000-1000"). <p>When using the advanced filtering format, follow these rules:</p> <ul style="list-style-type: none"> • Use = between the key and the regular expression that represents your filter, such as <code>whitelist = EventCode=%^1([8-9])\$%</code> • You can have multiple key/regular expression sets in a single advanced filtering entry. The Splunk platform joins the sets logically. This means that the entry is valid only if all of the sets in the entry are true. • You can specify up to 10 allow lists per stanza by adding a number to the end of the <code>whitelist</code> setting, such as <code>whitelist1...whitelist9</code>. 	N/A
blacklist	<p>Do not index events that match the text string specified. This setting is optional.</p> <p>You can specify one of two formats:</p> <ul style="list-style-type: none"> • One or more Event Log event codes or event IDs (Event Log code/ID format) • One or more sets of keys and regular expressions (Advanced filtering format) <p>You cannot mix formats in a single entry or mix formats in the same stanza.</p> <p>The Splunk platform processes allow lists first, then deny lists. If no allow list is present, the Splunk platform indexes all events.</p> <p>When using the Event Code/ID format, follow these rules:</p> <ul style="list-style-type: none"> • For multiple codes/IDs, separate the list with commas. • For ranges, use hyphens (for example "0-1000,5000-1000"). <p>When using the advanced filtering format, follow these rules:</p> <ul style="list-style-type: none"> • Use = between the key and the regular expression that represents your filter, such as <code>whitelist = EventCode=%^1([8-9])\$%</code> 	N/A

Setting	Description	Default
	<ul style="list-style-type: none"> You can have multiple key/regular expression sets in a single advanced filtering entry. the Splunk platform joins the sets logically. This means that the entry is valid only if all of the sets in the entry are true. You can specify up to 10 deny lists per stanza by adding a number to the end of the <code>blacklist</code> setting, for example <code>blacklist1...blacklist9</code>. 	
<code>suppress_text</code>	<p>Whether or not to include the message text that comes with a security event.</p> <p>A value of 1 suppresses the message text. A value of 0 preserves the text.</p>	0

Create advanced filters with the whitelist and blacklist settings

You can perform advanced filtering of incoming events with the `whitelist` and `blacklist` settings in addition to filtering based solely on event codes. To do this, specify the key/regular expression format in the setting:

```
whitelist = key=<regular expression> [key=<regular expression> ...
```

In this format, `key` is a valid entry from the following list:

Key	Description
<code>\$TimeGenerated</code>	The time that the computer generated the event. Only generates the time string as the event.
<code>\$Timestamp</code>	The time that the event was received and recorded by the Event Log service. the Splunk platform only generates the time string as the event.
<code>Category</code>	The category number for a specific event source.
<code>CategoryString</code>	A string translation of the category. The translation depends on the event source.
<code>ComputerName</code>	The name of the computer that generated the event.
<code>EventCode</code>	The event ID number for an event. Corresponds to "Event ID" in Event Viewer.
<code>EventType</code>	A numeric value that represents one of the five types of events that can be logged ("Error", "Warning", "Information", "Success Audit", and "Failure Audit"). Available only on server machines running Windows Server 2003 and lower or clients running Windows XP and lower.
<code>Keywords</code>	An element used to classify different types of events within an event log channel. The Security Event Log channel has this element, for example.
<code>LogName</code>	The name of the Event Log channel that received the event. Corresponds to "Log Name" in Event Viewer.
<code>Message</code>	The text of the message in the event.
<code>OpCode</code>	The severity level of the event. Corresponds to "OpCode" in Event Viewer.
<code>RecordNumber</code>	The Windows Event Log record number. Each event on a Windows server gets a record number. This number starts at 0 with the first event generated on the system, and increases with each new event generated, until it reaches a maximum of 4294967295. It then rolls back over to 0.
<code>Sid</code>	The Security Identifier (SID) of the principal, such as a user, group, computer, or other entity, that was associated with or generated the event. See <code>Win32_UserAccount</code> class at https://msdn.microsoft.com/en-us/library/windows/desktop/aa394507%28v=vs.85%29.aspx on

Key	Description
	MSDN.
SidType	A numeric value that represents the type of SID that was associated with the event. See Win32_UserAccount class at https://msdn.microsoft.com/en-us/library/windows/desktop/aa394507%28v=vs.85%29.aspx on MSDN.
SourceName	The source of the entity that generated the event. Corresponds to "Source" in Event Viewer.
TaskCategory	The task category of the event. Event sources allow you to define categories so that you can filter them with Event Viewer using the "Task Category" field. See Event Categories (Windows) at https://msdn.microsoft.com/en-us/library/aa363649%28VS.85%29.aspx on MSDN.
Type	A numeric value that represents one of the five types of events that can be logged ("Error", "Warning", "Information", "Success Audit", and "Failure Audit"). Only available on server machines that run Windows Server 2008 or higher or clients that run Windows Vista or higher.
User	The user associated with the event. Correlates to "User" in Event Viewer.

`<regular expression>` is any valid regular expression that represents the filters that you want to include when used with the `whitelist` setting or exclude when used with the `blacklist` setting.

You can specify more than one regular expression on a single entry line. Only events that satisfy all of the entries on the line are included or excluded. For example, this entry means to include events that have an `EventCode` ranging from 10 to 15 and contain a `Message` that begins with the word `Error`:

```
whitelist = EventCode="^1([0-5])$" Message="^Error"
```

You can specify up to 10 separate allow lists or deny list entries in each stanza. To do so, add a number at the end of the `whitelist` or `blacklist` setting entry on a separate line:

```
whitelist = key=<regular expression>
whitelist1 = key=<regular expression> key2=<regular expression 2>
whitelist2 = key=<regular expression>
```

You cannot specify an entry that has more than one expression that references the same key.

If you specify an entry using more than one expression referencing the same key, the Splunk platform ignores the first expression and only attempts to include events that match the second expression. See the following example:

```
whitelist = EventCode="^1([0-5])$" EventCode="^2([0-5])$"
```

In this case, only events that contain an `EventCode` between 20 and 25 match. Events that contain an `EventCode` between 10 and 15 do not match. Only the last expression in the entry ever matches.

To resolve this problem, specify two separate entries in the stanza:

```
whitelist = EventCode="^1([0-5])$"
whitelist1 = EventCode="^2([0-5])$"
```

Monitor file system changes

1. Confirm that you have administrator privileges.
2. Enable security auditing. Search for "Enable security auditing" for the version of Windows that you run.
3. Configure the Splunk platform event log monitor input to monitor the Security event log channel.

For instructions on how to configure the Event Log monitor input, see [Monitor Windows event log data](#).

Examples of file system change monitoring

The following inputs.conf stanzas show examples of how to monitor file system changes.

This stanza collects security events with event ID codes 0-2000 and 3001-10000.

```
[WinEventLog:Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
checkpointInterval = 5
# only index events with these event IDs.
whitelist = 0-2000,2001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000
```

This stanza collects security events with event ID codes 0-2000 and 3001-10000. It also suppresses the message text that comes in the event ID.

```
[WinEventLog:Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
checkpointInterval = 5
# suppress message text, we only want the event number.
suppress_text = 1
# only index events with these event IDs.
whitelist = 0-2000,2001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000
```

Monitor data through Windows Management Instrumentation (WMI)

The Splunk platform supports the use of Windows Management Instrumentation (WMI) providers for access to Windows performance and event log data on remote Windows machines without the need to install software on those machines.

To get WMI data into Splunk Cloud Platform, you can install a universal or heavy forwarder on a Windows machine and configure that forwarder to use the WMI data input to collect data remotely from other Windows machines, then forward that data to your Splunk Cloud Platform instance. Splunk Cloud Platform cannot connect directly to a Windows machine using WMI, so a universal or heavy forwarder is the only option.

WMI data inputs can connect to multiple WMI providers. The input runs on the forwarder as a separate process called `splunk-wmi.exe`. It is a **scripted input**.

If possible, when you need to collect Windows data remotely, install a universal forwarder directly onto the machine where you want to collect the Windows data, rather than use WMI. The resource load of WMI can exceed that of installing a universal forwarder in many cases. Use a forwarder if you collect multiple event logs or performance counters from each machine, or from very busy machines like domain controllers. See [Considerations for deciding how to monitor remote Windows data](#).

What do you need to monitor WMI-based data?

Here are the minimum requirements to monitor WMI-based data. You might need additional permissions based on the event logs or performance counters you want to monitor.

For additional details on what you need to monitor WMI-based data, see [Security and remote access considerations](#) in this topic.

Activity	Required permissions
Monitor remote event logs over WMI	<ul style="list-style-type: none"> • Splunk Cloud Platform must receive data from a forwarder that runs on Windows. • Splunk Enterprise can collect WMI data directly if it runs on a Windows machine. • The forwarder must run as a Windows domain user with at least read access to WMI. • The forwarder must run as a domain user with appropriate access to the desired event logs. For example, a user who is a member of the Event Log Readers group has appropriate access.
Monitor remote performance monitor counters over WMI	<ul style="list-style-type: none"> • Splunk Cloud Platform must receive data from a forwarder that runs on Windows. • Splunk Enterprise can collect performance data over WMI if it runs on a Windows machine. • The forwarder must run as a domain user with at least read access to WMI. • The forwarder must run as a domain user with appropriate access to the Windows Performance Data Helper libraries.

If you run Splunk Enterprise, you can collect data over WMI if the machine that you install the instance on runs Windows, or the forwarders that you use to send data to the instance run Windows.

Security and remote access considerations

Both of the Splunk platform instances that get WMI data and your Windows network must be correctly configured for data access over WMI. Whether the instance that indexes your data is a Splunk Cloud Platform or a Splunk Enterprise instance, review the following prerequisites before attempting to use the platform to get data over WMI.

Before the Splunk platform can get WMI-based data:

- The instance that gets the data must be installed with a Windows user that has permissions to perform remote network connections.
- The Windows user that the instance runs as must be a member of an Active Directory (AD) domain or forest and must have appropriate privileges to query WMI providers.
- The instance user must also be a member of the local Administrators group on the machine that runs the instance.
- The machine that runs the instance must be able to connect to the remote machine and must have permissions to get the data from the remote machine after it has connected.
- Both the Splunk platform instance and the target machines must be part of the same AD domain or forest.

The user that the Splunk platform instance runs as does not need to be a member of the Domain Admins group, and for security reasons, should not be. However, you must have domain administrator privileges to configure access for the user. If you don't have domain administrator access, find someone who can either configure Splunk user access or give

domain administrator rights to you.

If you install the instance as the Local System Windows user, remote authentication over WMI does not work. The Local System user has no access to other machines on the network. It is not possible to grant privileges to a Local System account for access to another machine.

You can give the instance user access to WMI providers by doing one of the following:

- Assigning least-permissive rights, as detailed in "Configure WMI for least permissive access" later in this topic. This is the recommended option.
- Adding the user to the local Administrators group on each Windows member machine you want to poll. For security reasons this is not recommended.
- Adding the user to the Domain Admins global group. For security reasons, this is not recommended.

Group memberships and resource access control lists (ACLs)

To maintain security integrity, place the Windows users that Splunk platform instances run as into a domain global group and assign permissions on Windows machines and resource ACLs to that group, rather than assigning permissions directly to the user. Assignment of permissions directly to users is a security risk, and can cause problems during security audits or future changes.

Configure WMI for least permissive access

If the Windows user that you configured the Splunk platform to run as is not a domain administrator, you must configure WMI to provide access to this user. Grant least-permissive access to all Windows resources, including WMI, by following this checklist. For additional information and step-by-step instructions, see *Prepare your Windows network for a Splunk Enterprise installation* in the *Installation* manual. These instructions are also valid for universal forwarders.

You must grant several levels of access to the user the Splunk platform runs as for the instance to collect data over WMI using the least-permissive method.

- **Local Security Policy Permissions.** The Splunk platform Windows user needs the following Local Security Policy user rights assignments defined on each machine you poll for WMI-based data:
 - ◆ Access this Computer from the Network
 - ◆ Act as part of the operating system
 - ◆ Log on as a batch job
 - ◆ Log on as a service
 - ◆ Profile System Performance
 - ◆ Replace a process level token
- **Distributed Component Object Model (DCOM) configuration and permissions.** You must enable DCOM on every Windows machine you want to monitor. In addition, the Splunk platform Windows user must have access to DCOM. There are many methods available to do this, but the best is to nest the Distributed COM Users domain global group into the Distributed COM Users local group on each Windows machine you want to monitor, then add the Splunk platform Windows user to the Distributed COM Users domain global group. Search the Microsoft documentation website for "Securing a Remote WMI Connection" for specific details and instructions.
- **Performance Monitor configuration and permissions.** The Splunk platform Windows user must be a member of the Performance Log Users local group for the Splunk platform to access remote performance objects over WMI. The best way to do this is to put the Performance Log Users domain global group inside of the Performance

Log Users local group on each Windows machine, and then assign the Windows user to the Performance Log Users group.

- **WMI namespace security.** The WMI namespace that the Splunk platform accesses must have proper permissions. In nearly all cases, the `Root\CIMV2` namespace is acceptable for collecting WMI data. You must set namespace permissions manually on each Windows machine in your enterprise, as there is no global security for WMI. You must assign these rights must be assigned to the Root namespace and all subnamespaces below it. Search the Microsoft documentation website for "Managing WMI security". Enable the following permissions on the WMI tree for each host at the Root namespace for the Splunk user:
 - ◆ Execute Methods
 - ◆ Enable Account
 - ◆ Remote Enable
 - ◆ Read Security

There is no standard policy to deploy WMI security settings remotely to multiple machines at once using Group Policy. However, there is a blog on the Microsoft documentation website that describes using a startup script to achieve this task. Search the Microsoft documentation website for "Set WMI namespace security by using GPO (script)".

- **Firewall configuration.** If you have an active firewall, you must configure it to allow access for WMI. If you use the Windows Firewall included with recent versions of Windows, an exceptions list explicitly includes WMI. You must set this exception for both the originating and the target Windows machines. Search the Microsoft documentation website for "Connecting to WMI Remotely".
- **User access control (UAC) configuration.** If you run Windows Vista and higher, or the Windows Server 2012 and higher families, UAC affects how Windows assigns permissions. Search the Microsoft documentation website for "User Account Control and WMI".

Test access to WMI providers

After you configure WMI and set up the Splunk platform instance user for access to your domain, test access to the remote machine.

This procedure includes steps to temporarily change the Splunk platform instance data store directory (the location `SPLUNK_DB` points to). You must do this before testing access to WMI. Failure to do so can result in missing WMI events. This is because the `splunk-wmi.exe` process updates the WMI checkpoint file every time it runs.

If you attempt to log into a domain controller, you might have to change your domain controller security policy to assign the Allow log on locally policy for the designated user.

1. Log into the Windows machine that the Splunk platform runs on as the Splunk platform Windows user.
2. Open a command prompt (click **Start > Run** and type `cmd`).
3. Go to the `bin` subdirectory under your Splunk platform installation (for example, `cd c:\Program Files\Splunk\bin`).
4. Determine where the platform instance currently stores its data by running the following command:

```
> splunk show datastore-dir
```

Remember where the platform instance stores its data. You will recall it later.

5. Run the following command to change where the Splunk platform stores its data temporarily:

```
> splunk set datastore-dir %TEMP%
```

This example sets the data store directory to the directory that the TEMP environment variable specifies. If you want to set it to a different directory, you can do so, but the directory must already exist.

6. Restart the Splunk platform instance:

```
> splunk restart
```

It might take a while for the Splunk platform instance to restart.

7. Once The Splunk platform instance has restarted, test access to WMI providers, replacing `<host>` with the name of the remote machine:

```
> splunk cmd splunk-wmi -wql "select * from win32_service" -namespace \\<host>\root\cimv2
```

- ◆ If you see data stream back and no error messages, then the instance was able to connect to the WMI provider and query successfully.
- ◆ If there is an error, a message with a reason on what caused the error appears. Look for the `error="<msg>"` string in the output for clues on how to correct the problem.

After testing WMI access, point the Splunk platform instance back to the correct database directory by running the following command, and then restarting the instance:

```
> splunk set datastore-dir <directory shown from Step 4>
```

Configure WMI-based inputs

All remote data collection in the Splunk platform on Windows happens through either WMI providers or a forwarder.

If you are using Splunk Cloud Platform, you must use a forwarder to collect WMI data and send it to your Splunk Cloud Platform instance.

If you are using Splunk Enterprise, you can configure WMI-based inputs using either Splunk Web or a configuration file. More configuration options are available when using a file.

Configure WMI-based inputs with Splunk Web

Splunk Cloud Platform isn't able to collect WMI data natively, but you can use Splunk Web on a Splunk Enterprise instance that you have configured as a heavy forwarder to collect the data and send it to your Splunk Cloud Platform instance.

To add WMI-based inputs in Splunk Web, use the Remote event log monitoring and Remote Performance monitoring data inputs. See the following topics for instructions:

- [Configure remote Windows performance monitoring with Splunk Web.](#)
- [Configure remote Windows event log monitoring](#)

Configure WMI-based inputs with configuration files

Splunk Cloud Platform isn't able to collect WMI data natively, so if you do not want to use a heavy forwarder, you must use a universal forwarder and modify configuration files.

The `wmi.conf` configuration file handles remote data collection through WMI. Review this file to see the default values for WMI-based inputs. If you want to make changes to the default values, add the settings and values you want to change to a separate version of the file, in the `%SPLUNK_HOME%\etc\system\local\` directory on the universal or heavy forwarder that collects the WMI data. Set values only for the settings you want to change for a given type of data input.

See the `wmi.conf` file in the *Admin Manual* to learn about the default settings and values for the file, and About configuration files for information on configuring files.

The `wmi.conf` file contains several stanzas:

- The `[settings]` stanza, which specifies global WMI settings.
- One or more input-specific stanzas, which define how to connect to WMI providers to get data from the remote Windows machine.

Global settings

The `[settings]` stanza specifies global WMI parameters. The entire stanza and every setting within it are optional. If the stanza is not present, the Splunk platform uses the default system values.

The following settings control how the Splunk platform reconnects to a given WMI provider when an error occurs.

Setting	Description	Default value
<code>initial_backoff</code>	How long, in seconds, to wait the first time after an error occurs before trying to reconnect to the WMI provider. If connection errors continue to occur, the Splunk platform doubles the wait time until it reaches the value specified in <code>max_backoff</code> .	5
<code>max_backoff</code>	How long, in seconds, to wait between connection attempts, before invoking <code>max_retries_at_max_backoff</code> .	20
<code>max_retries_at_max_backoff</code>	If the wait time between connection attempts reaches <code>max_backoff</code> , how many times to try to reconnect to the provider, every <code>max_backoff</code> seconds. If the Splunk platform continues to encounter errors, it gives up, and won't attempt to connect to the problem provider again until you restart. It will continue to log errors such as the example shown above.	2
<code>checkpoint_sync_interval</code>	How long, in seconds, to wait for state data (event log checkpoint) to be written to disk.	2

Input-specific settings

Input-specific stanzas configure the Splunk platform to connect with WMI providers. They are defined by one of two settings that specify the type of data the Splunk platform should gather. The stanza name can be anything, but usually begins with `WMI:`, for example:

```
[WMI:AppAndSys]
```

When you configure WMI-based inputs in Splunk Web, Splunk uses this naming convention for input-specific stanza headers.

You can specify one of two types of data inputs in an input-specific stanza:

Data input	Description
Event log	The <code>event_log_file</code> setting configures the Splunk platform to expect event log data from the sources that the stanza defines.
Windows Query Language (WQL)	The <code>wql</code> setting configures the Splunk platform to expect data from a WMI provider. You must also specify a valid WQL statement. You must use this setting when you collect performance data.

Do not define both of these settings in one stanza. Use only one or the other. Otherwise, the input defined by the stanza will not run.

The common settings for both types of inputs are:

Setting	Description	Default value
server	A comma-separated list of Windows machines from which to get data. If this setting is missing, the Splunk platform attempts to connect to the local machine.	The local host
interval	How often, in seconds, to poll the WMI provider for new data. If this setting is not present and defined, the input that the stanza defines will not run.	Not applicable
disabled	Whether this input is enabled or disabled. Set this setting to 1 to disable the input, and 0 to enable it.	0 (enabled)

The event log-specific parameters are:

Setting	Description	Default value
event_log_file	A comma-separated list of event log channels to monitor.	Not applicable
current_only	Whether or not to collect events that occur only when the Splunk platform runs. If events are generated when the Splunk platform is stopped, it will not attempt to index those events when it is started again. Set to 1 to collect events that occur only when it is running, and 0 to collect all events.	0 (gather all events)
disable_hostname_normalization	Do not normalize the host name that is retrieved from a WMI event. By default, the Splunk platform normalizes host names by producing a single name for the host through identifying various equivalent host names for the local system. Set this parameter to 1 to disable host name normalization in events, and 0 to normalize host names in events	0 (normalize host names for WMI events)

The WQL-specific parameters are:

Setting	Description	Default value
wql	A valid WQL statement.	Not applicable
namespace	(Optional) Specifies the path to the WMI provider. The local machine must be able to connect to the remote machine using delegated authentication. If you do not specify a path to a remote machine, the Splunk platform connects to the default local namespace (\Root\CIMV2). This default namespace is where most of the providers that you can to query reside.	\\<local server>\Root\CIMV2
current_only	Whether or not an event notification query is expected. See "WQL query types: event notification versus standard" in this topic for additional information. Set this setting to 1 to tell the Splunk platform to expect an event notification query, and 0 to expect a standard query.	0 (expect a standard query)

WQL query types: event notification versus standard

The `current_only` setting in WQL stanzas determines the type of query the stanza expects to use to collect WMI-based data. When you configure the setting to 1, the stanza expects event notification data. Event notification data is data that alerts you of an incoming event. To get event notification data, you must use an event notification query.

For example, to find out when a remote host spawns processes, you must use an event notification query. Standard queries have no facilities for notifying you when an event has occurred, and can only return results on information that already exists.

Conversely, if you want to know what already-running processes on your system begin with the word "splunk", you must use a standard query. Event notification queries cannot tell you about static, preexisting information.

Event notification queries require that the WQL statement defined for the stanza be structurally and syntactically correct. Improperly formatted WQL will cause the input defined by the stanza to not run. Review the `wmi.conf` configuration file reference for specific details and examples.

WQL query stanzas do not update the WMI checkpoint file

When you use a WQL query stanza to gather data through WMI, the Splunk platform does not update the WMI checkpoint file, which is the file that determines if WMI data has been indexed. A WQL query of any type returns dynamic data and a context for saving a checkpoint for the data produced cannot be built. So, the Splunk platform indexes WMI data that it collects through WQL query stanzas as fresh data each time the stanza runs. This process can result in the indexing of duplicate events and possibly impact license volume.

If you need to index data regularly, such as event logs, use the appropriate monitor on a universal forwarder. If you must use WMI, use a standard WMI query type.

Examples of wmi.conf

The following is an example of a `wmi.conf` file:

```
[settings]
initial_backoff = 5
max_backoff = 20
max_retries_at_max_backoff = 2
checkpoint_sync_interval = 2

[WMI:AppAndSys]
server = foo, bar
interval = 10
event_log_file = Application, System, Directory Service
disabled = 0

[WMI:LocalSplunkWmiProcess]
interval = 5
wql = select * from Win32_PerfFormattedData_PerfProc_Process where Name = "splunk-wmi"
disabled = 0

# Listen from three event log channels, capturing log events that occur only
# while the Splunk platform runs. Gather data from three machines.
[WMI:TailApplicationLogs]
interval = 10
event_log_file = Application, Security, System
server = srv1, srv2, srv3
disabled = 0
current_only = 1

# Listen for process-creation events on a remote machine
[WMI:ProcessCreation]
interval = 1
server = remote-machine
wql = select * from __InstanceCreationEvent within 1 where TargetInstance isa 'Win32_Process'
disabled = 0
current_only = 1

# Receive events whenever someone plugs/unplugs a USB device to/from the computer
[WMI:USBChanges]
interval = 1
wql = select * from __InstanceOperationEvent within 1 where TargetInstance ISA 'Win32_PnPEntity' and
TargetInstance.Description='USB Mass Storage Device'
disabled = 0
current_only = 1
```

Fields for WMI data

When the Splunk platform indexes data from WMI-based inputs, it sets the originating host from the data it receives. It sets the **source** for received events to `wmi`. It sets the **source type** of the incoming events based on the following conditions:

- For event log data, the Splunk platform sets the source type to `WinEventLog:<name of log file>`. For example, `WinEventLog:Application`.
- For WQL data, the Splunk platform sets the source type to the name of the stanza that defines the input. For example, for a stanza named `[WMI:LocalSplunkdProcess]`, Splunk sets the source type to `WMI:LocalSplunkdProcess`.

WMI and event transformations

WMI events are not available for transformation at index time. You cannot modify or extract WMI events as the Splunk platform indexes them. This is because WMI events arrive as a single source (a scripted input), which means they can be matched only as a single source.

You can modify and extract WMI events at search time. You can also address WMI-based inputs at parse time by specifying the `sourcetype` `[wmi]`.

For information on how to transform events as they arrive in the Splunk platform, see [About indexed field extraction](#) in this manual.

Troubleshoot WMI inputs

When the Splunk platform cannot connect to a defined WMI provider, it generates an error like the following:

```
05-12-2011 02:39:40.632 -0700 ERROR ExecProcessor - message from ""C:\Program Files\Splunk\bin\splunk-wmi.exe""
WMI - Unable to connect to WMI namespace "\\w2k3m1\root\cimv2" (attempt to connect took 42.06 seconds) (error="The
RPC server is unavailable." HRESULT=800706BA)
```

When this occurs, try the following tips:

- Confirm that the machine that is polling the instance has network access to the machine.
- Confirm that you have configured the instance for least permissive access to the WMI provider and have completed all of the security requirements.
- Confirm that the Splunk platform Windows user has access to WMI

If you encounter problems receiving events through WMI providers or are not getting the results you expect, see [Common Issues with Splunk and WMI](#) in the *Troubleshooting Manual*.

Monitor Windows Registry data

The Windows Registry is the central configuration database on a Windows machine. Nearly all Windows processes and third-party programs interact with it. Without a healthy Registry, Windows does not run. The Splunk platform supports the capture of Windows Registry settings and lets you monitor changes to the Registry in real time.

When a program makes a change to a configuration, it writes those changes to the Registry. When the program runs again, it looks into the Registry to read those configurations. You can learn when Windows programs and processes add,

update, and delete Registry entries on your system. When a Registry entry changes, the Splunk platform captures the name of the process that made the change, as well as the entire path to the entry being changed.

If you use Splunk Cloud Platform, you must install the universal forwarder on a Windows machine to collect data from the Windows Registry and forward it to your Splunk Cloud Platform deployment.

Reasons to monitor the Registry

Many programs and processes read from and write to it at all times. When something is not functioning, Microsoft often instructs administrators and users alike to make changes to the Registry directly using the RegEdit tool. The ability to capture those edits, and any other changes, in real time is the first step in understanding the importance of the Registry.

Registry health is very important. The Splunk platform tells you when changes to the Registry are made and also if those changes were successful. If programs and processes can't write to or read from the Registry, a system failure can occur. The Splunk platform can alert you to problems interacting with the Registry so that you can restore it from a backup and keep your system running.

Requirements to monitor the Registry

The following table lists the explicit permissions you need to monitor the Registry. You might need additional permissions based on the Registry keys that you want to monitor.

Activity	Required permissions
Monitor the Registry	<ul style="list-style-type: none">• Splunk Enterprise or the universal forwarder must run on Windows.• The Splunk platform instance must run as the Local System user, or as a domain user with read access to the Registry hives or keys that you want to monitor.

Performance considerations

When you enable Registry monitoring, you specify which Registry hives to monitor: the user hive, represented as `HKEY_USERS` in RegEdit, or the machine hive, represented as `HKEY_LOCAL_MACHINE`. The user hive contains user-specific configurations required by Windows and programs, and the machine hive contains configuration information specific to the machine, such as the location of services, drivers, object classes, and security descriptors.

Because the Registry plays a central role in the operation of a Windows machine, enabling both Registry paths can result in a lot of data for the Splunk platform to monitor. To achieve the best performance, filter the amount of Registry data that the platform indexes by configuring the `inputs.conf` configuration file.

You can capture a baseline snapshot of the current state of your Windows Registry when you first start the Splunk platform, and again every time a specified amount of time has passed. The snapshot lets you compare what the Registry looks like at a certain point in time and provides for easier tracking of the changes to the Registry over time.

The snapshot process can be CPU-intensive, and might take several minutes to complete. You can postpone taking a baseline snapshot until you have narrowed the scope of the Registry entries to those you specifically want the Splunk platform to monitor.

Enable Registry event monitoring using configuration files

Windows Registries generate a great number of events due to their near-constant use. This can cause problems with licensing. Splunk Registry monitoring can generate hundreds of megabytes of data per day.

Splunk Windows Registry monitoring uses a configuration file to determine what to monitor on your system, `inputs.conf`. This file must reside in `%SPLUNK_HOME%\etc\system\local\` on the machine that runs Registry monitoring.

The `inputs.conf` file contains the specific regular expressions you create to refine and filter the Registry hive paths you want the Splunk platform to monitor.

Each stanza in the `inputs.conf` file represents a particular filter whose definition includes:

Attribute	Description
<code>proc</code>	A regular expression containing the path to the process or processes you want to monitor. Default: <code>.*</code> , or all processes.
<code>hive</code>	<p>A regular expression that contains the hive path to the entry or entries you want to monitor. Splunk supports the root key value mappings predefined in Windows:</p> <ul style="list-style-type: none"> <code>\\REGISTRY\\USER\\</code> maps to <code>HKEY_USERS</code> or <code>HKU</code> <code>\\REGISTRY\\USER_Classes</code> maps to <code>HKEY_CLASSES_ROOT</code> or <code>HKCR</code> <code>\\REGISTRY\\MACHINE\\</code> maps to <code>HKEY_LOCAL_MACHINE</code> or <code>HKLM</code> <code>\\REGISTRY\\MACHINE\\SOFTWARE\\Classes</code> maps to <code>HKEY_CLASSES_ROOT</code> or <code>HKCR</code> <code>\\REGISTRY\\MACHINE\\SYSTEM\\CurrentControlSet\\Hardware Profiles\\Current</code> maps to <code>HKEY_CURRENT_CONFIG</code> or <code>HKCC</code> There is no direct mapping for <code>HKEY_CURRENT_USER</code> or <code>HKCU</code> because the Registry monitor runs in kernel mode. Use <code>\\REGISTRY\\USER\\.*</code> The period and asterisk at the end are required. Use this mapping to generate events that contain the security identifier (SID) of the logged-in user. As an alternative to using the previously-described mappings, you can specify the user whose Registry keys you want to monitor by using <code>\\REGISTRY\\USER\\<SID></code>, where <code>SID</code> is the SID of the user.
<code>type</code>	The subset of event types to monitor. This subset can be one or more of <code>delete</code> , <code>set</code> , <code>create</code> , <code>rename</code> , <code>open</code> , <code>close</code> or <code>query</code> . The values for this attribute must be a subset of the values for <code>event_types</code> that you set in <code>inputs.conf</code> .
<code>baseline</code>	Whether or not to capture a baseline snapshot for that particular hive path. Set to 1 for yes, and 0 for no.
<code>baseline_interval</code>	How much time, in seconds, must have elapsed since the last baseline was taken before the Splunk platform takes another baseline on startup. For example, if you set <code>baseline_interval</code> to 600, then when the Splunk platform starts or restarts, it takes a baseline if the existing baseline is more than 600 seconds old. If no baseline exists, then the Splunk platform takes a baseline immediately. This setting has no effect if you do not also set <code>baseline</code> to 1. The default value is 86,400 seconds (1 day).
<code>disabled</code>	Whether or not a filter is enabled. Set to 1 to disable the filter, and 0 to enable it.

Enable Registry monitoring in Splunk Web

You can use Splunk Web to configure Windows Registry monitoring on a Splunk Enterprise instance.

Go to the Add New page

You can get there by two routes:

- Splunk Home
- Splunk Settings

By Splunk Settings:

1. Click **Settings** in the upper right corner of Splunk Web.
2. Click **Data Inputs**.
3. Click **Registry monitoring**.

4. Click **New** to add an input.

By Splunk Home:

1. Click the **Add Data** link in Splunk Home.
2. Click **Monitor** to monitor Registry data on the local Windows machine.

Select the input source

1. Locate and select **Registry monitoring**.
2. In the **Collection Name** field, enter a unique name for the input that you will remember.
3. In the **Registry hive** field, enter the path to the Registry key that you want the Splunk platform to monitor. If you plan to monitor more than one hive, each hive requires its own separate input.
If you are not sure of the path, click the **Browse** button to select the Registry key path that you want the Splunk platform to monitor. The **Registry hive** window opens and displays the Registry in tree view. Hives, keys and subkeys display as folders, and values display as document icons. The `HKEY_USERS`, `HKEY_CURRENT_USER`, `HKEY_LOCAL_MACHINE`, and `HKEY_CURRENT_CONFIG` hives display as top-level objects. The `HKEY_CLASSES_ROOT` hive is not shown because of the number of subkeys present in the first sublevel of that hive. To access `HKEY_CLASSES_ROOT` items, choose `HKEY_LOCAL_MACHINE\Software\Classes`.
4. In the **Registry hive** window, click the name of the Registry key you want. The qualified key name appears in the **Qualified name** field at the bottom of the window.
5. Click **Select** to confirm the choice and close the window.
6. (Optional) Select **Monitor subnodes** if you want to monitor the child nodes below the starting hive.

The Monitor subnodes node determines what the Splunk platform adds to the inputs.conf file that it creates when you define a Registry monitor input in Splunk Web.

If you use the tree view to select a key or hive to monitor and check **Monitor subnodes**, then the Splunk platform adds a **regular expression** to the stanza for the input you are defining. This regular expression (`\\\\\\?.*`) filters out events that do not directly reference the selected key or any of its subkeys.

If you do not check **Monitor subnodes**, then the Splunk platform adds a regular expression to the input stanza which filters out events that do not directly reference the selected key, including events that reference subkeys of the selected key.

If you do not use the tree view to specify the key you want to monitor, then the Splunk platform adds the regular expression only if you have checked **Monitor subnodes** and have not entered your own regular expression in the **Registry hive** field.

7. Under **Event types**, select the Registry event types that you want the Splunk platform to monitor for the chosen Registry hive:

Event Type	Description
Set	The Splunk platform generates a Set event when a program executes a SetValue method on a Registry subkey, thus setting a value or overwriting an existing value on an existing Registry entry.
Create	The Splunk platform generates a Create event when a program executes a CreateSubKey method within a Registry hive, creating a new subkey within an existing Registry hive.
Delete	The Splunk platform generates a Delete event when a program executes a DeleteValue or DeleteSubKey method. This method either removes a value for a specific existing key, or it removes a key from an existing hive.
Rename	The Splunk platform generates a Rename event when you rename a Registry key or subkey in RegEdit.

Event Type	Description
Open	The Splunk platform generates an Open event when a program executes an OpenSubKey method on a Registry subkey, such as what happens when a program needs configuration information contained in the Registry.
Close	The Splunk platform generates a Close event when a program executes a Close method on a Registry key. This happens when a program is done reading the contents of a key, or after you make a change to a key's value in RegEdit and exit the value entry window.
Query	The Splunk platform generates a Query event when a program executes the GetValue method on a Registry subkey.

- Specify which processes the Splunk platform should monitor for changes to the Registry by entering appropriate values in the **Process Path** field. Or, leave the default of . * to monitor all processes.
- Specify whether or not you want to take a baseline snapshot of the whole Registry before monitoring Registry changes. To set a baseline, click **Yes** under **Baseline index**.

The baseline snapshot is an index of your entire Registry, at the time the snapshot is taken. Registry events within the snapshot retain their original indexing timestamps.

</caution>Scanning the Registry to set a baseline index is a CPU-intensive process and might take some time.</caution>

- Click **Next**.

Specify input settings

The **Input Settings** page lets you specify application context, default host value, and index. All of these parameters are optional.

- Select the appropriate **Application context** for this input.
- In the **Host** field, set the host name value.

This field only sets the host field in the resulting events. It does not direct the Splunk platform to look on a specific host on your network.

- In the **Index** field, set the index that the Splunk platform should send data to. Leave the value as "default" unless you have defined multiple indexes to handle different types of events. In addition to indexes for user data, the Splunk platform has a number of utility indexes, which also appear in this dropdown list box.
- Click **Review**.

Review your choices

After specifying all your input settings, review your selections. The Splunk platform lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

- Review the settings.
- If the settings do not match what you want, click < to go back to the previous step in the wizard.
- Click **Submit**.

The Splunk platform then loads the Success page and begins indexing the specified Registry nodes.

View Registry change data

To view Registry change data that the Splunk platform indexed, go to the Search app and search for events with a source of `WinRegistry`. An example event, which Group Policy generates when a user logs in to a domain, follows:

```

3:03:28.505 PM
06/19/2011 15:03:28.505
event_status="(0) The operation completed successfully."
pid=340
process_image="c:\WINDOWS\system32\winlogon.exe"
registry_type="SetValue"
key_path="HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\History\DCName"
data_type="REG_SZ"
data="//ftw.ad.splunk.com"

```

Each registry monitoring event contains the following attributes:

Setting	Description
event_status	The result of the registry change attempt. This result will be "(0) The operation completed successfully." If it is not, there might be problems with the Registry that require a restore from a backup.
pid	The process ID of the process that attempted to make the Registry change.
process_image	The name of the process that attempted to make the Registry change.
registry_type	The type of Registry operation that the process_image attempted to invoke.
key_path	The Registry key path that the process_image attempted to make a change to.
data_type	The type of Registry data that the process_image making the Registry change tried to get or set.
data	The data that the process_image making the Registry change tried to read or write.

Get a baseline snapshot

When you enable Registry monitoring, you can record a baseline snapshot of the Registry hives the next time the Splunk platform starts. By default, the snapshot covers the `HKEY_CURRENT_USER` and `HKEY_LOCAL_MACHINE` hives. It also establishes a timeline for when to retake the snapshot. By default, if the baseline is more than 24 hours old, when the Splunk platform next starts, it retakes the baseline snapshot. You can customize this value for each of the filters in `inputs.conf` by setting the value of `baseline_interval`, in seconds.

When you create a baseline snapshot, the snapshot uses the index time of the Registry data, not the snapshot creation time. For example, if a change to a Registry key occurred two years ago, the timestamp for that event will be two years ago, not when the baseline snapshot was created.

Monitor Windows performance

Splunk Enterprise supports the monitoring of all Windows performance counters in real time, which includes support for both local and remote collection of performance data.

The performance monitoring input gives you access to the Performance Monitor in a web interface. The Splunk platform uses the Windows Performance Data Helper (PDH) API for performance counter queries on local Windows machines.

The types of performance objects, counters, and instances that are available to the platform depend on the performance libraries that are on the machine. Both Microsoft and third-party vendors provide libraries that contain performance counters. For information on performance monitoring, search the Microsoft documentation website for "Performance Counters".

To get performance monitor data into Splunk Cloud Platform, you must run either a Splunk Enterprise heavy forwarder or universal forwarder on the Windows machine from which you want to collect the performance metrics, and then forward

that data to the Splunk Cloud Platform instance. Both full instances of Splunk Enterprise and universal forwarders can collect local performance metrics. Remote performance monitoring is available through Windows Management Instrumentation (WMI) and requires that the Splunk platform instance on the Windows machine runs as a user with appropriate Active Directory credentials.

On Splunk Enterprise and the universal forwarder, the performance monitor input runs as a process called `splunk-perfmon.exe`. The process runs once for every input you define, at the interval you specify in the input. You can configure performance monitoring either with Splunk Web or by using configuration files.

The performance monitor input uses two files for configuration. The file that you use to configure the input depends on whether you want to get performance data from a local instance or from a remote instance:

- You use the `inputs.conf` configuration file to get local performance data.
- You use the `wmi.conf` configuration file to get performance data from a remote machine.

Why monitor performance metrics?

Performance monitoring is an important part of the Windows administrator toolkit. Windows generates a lot of data about machine health. Properly analyzing that data can mean the difference between a healthy, well-functioning machine, and one that suffers downtime.

What you need to monitor performance counters

The following table lists the minimum requirements you need to monitor performance counters in Windows. You might have additional requirements based on the performance objects or counters that you want to monitor.

For additional information on performance metrics monitoring requirements, see [Security and remote access considerations](#) later in this topic.

Activity	Required permissions
Monitor local performance metrics	<ul style="list-style-type: none">* The Splunk Cloud Platform instance must receive performance data from a forwarder.* The forwarder must run on Windows. See <i>Install on Windows</i> in the Splunk Enterprise <i>Installation Manual</i>.* The forwarder must run as the LocalSystem Windows user. Choose the Windows user Splunk Enterprise should run as in the Splunk Enterprise <i>Installation Manual</i>.
Monitor remote performance metrics on another computer over WMI	<ul style="list-style-type: none">* The Splunk Cloud Platform instance must receive performance data from a forwarder.* The forwarder must run on Windows.* The forwarder must run as a domain or remote user with at least read access to WMI on the target machine.* The forwarder must run as a domain or remote user with appropriate access to the Performance Data Helper libraries on the target machine.

Security and remote access considerations

Where possible, use a universal forwarder to send performance data from remote machines to a Splunk Cloud Platform or Splunk Enterprise indexer.

Splunk Enterprise gets data from remote machines with either a forwarder or WMI.

If you install forwarders on your remote Windows machines to collect performance data, then you can install the forwarder as the LocalSystem user on those machines. The LocalSystem user has access to all data on the local machine, but not to remote computers.

If you want Splunk Enterprise to use WMI to get performance data from remote machines, then you must configure both Splunk Enterprise and your Windows network. You cannot install Splunk Enterprise as the LocalSystem user, and the user that you choose determines what Performance Monitor objects Splunk Enterprise can read.

After you install Splunk Enterprise with a valid user, you must add that user to the following groups before you enable local performance monitor inputs:

- Performance Monitor Users (domain group)
- Performance Log Users (domain group)

To learn more about WMI security, see [Security and remote access considerations](#) in the Monitor data through Windows Management Instrument (WMI) topic. To learn how to use a universal forwarder, see The universal forwarder in the Splunk Universal Forwarder *Forwarder Manual*.

Enable local Windows performance monitoring

On Splunk Cloud Platform, you must forward data from the Windows machines where you want to collect performance data.

On Splunk Enterprise, you can configure local performance monitoring directly either in Splunk Web or with configuration files.

Splunk Web is the preferred way to add performance monitoring data inputs on Splunk Enterprise instances. Typos are easy to make in configuration files, and it is important to specify performance monitor objects exactly as the Performance Monitor API defines them. See "Important information about specifying performance monitor objects in inputs.conf" later in this topic for a full explanation.

Configure local Windows performance monitoring with Splunk Web

You can collect Windows performance monitoring metrics with Splunk Web only on Splunk Enterprise instances.

To begin configuring Windows performance monitoring metrics, access the Add New page in Splunk Web through either Splunk Settings or Splunk Home.

To connect Windows performance monitoring metrics through Splunk Settings, follow these steps:

1. Click **Settings > Data Inputs**.
2. Click **Local performance monitoring**.
3. Click **New** to add an input.
4. Continue with the steps in "Select an input source" later in this topic.

To connect Windows performance monitoring metrics through Splunk Home, follow these steps:

1. Click the **Add Data** link in Splunk Home.
2. Click **Monitor** to monitor performance data from the local Windows machine, or **Forward** to receive performance data from another machine.
3. If you selected **Forward**, choose or create the group of forwarders you want this input to apply to.
4. Click **Next**.
5. Continue with the steps in "Select an input source" later in this topic.

Select the input source

1. In the left pane of Splunk Enterprise, select **Local Performance Monitoring**.
2. In the **Collection Name** field, enter a unique name for this input that you will remember.
3. Click **Select Object** to get a list of the performance objects available on this Windows machine, then choose the object that you want to monitor from the list. Splunk Enterprise displays the **Select Counters** and **Select Instances** list boxes.
4. In the **Select Counters** list box, locate the performance counters you want this input to monitor.

You can add only one performance object per data input. If you need to monitor multiple objects, create additional data inputs for each object.

5. Click once on each counter you want to monitor. Splunk Enterprise moves the counter from the **Available counter(s)** window to the **Selected counter(s)** window.
6. (Optional) To unselect a counter, click its name in the **Available Items** window.
7. (Optional) To select or unselect all of the counters, click the **add all** or **remove all** links.

Selecting all of the counters can result in the indexing of a lot of data.

8. (Optional) In the **Select Instances** list box, select the instances that you want this input to monitor by clicking once on the instance in the **Available instance(s)** window.
Selecting all of the counters can result in the indexing of a lot of data and possibly lead to license violations.
9. In the **Polling interval** field, enter the time, in seconds, between polling attempts for the input.

The `_Total` instance is a special instance, and appears for many types of performance counters. This instance is the average of any associated instances under the same counter.

10. Click **Next**.

Specify input settings

Specify application context, default host value, and index in the **Input Settings** page. All of these parameters are optional.

Setting the **Host** on this page sets only the **host** field in the resulting events. It doesn't direct Splunk Enterprise to look on a specific host on your network.

1. In Splunk Enterprise, select the application context for the input in the **Application context** field.
2. Set the **Host** value. You have several choices for this setting. Learn more about setting the host value in [About hosts](#).
3. Set the **Index** that you want Splunk Enterprise to send data to. Leave the value as `default`, unless you have defined multiple indexes to handle different types of events.
4. Click **Review**.

Review your choices

After you specify input settings, review your selections. Splunk Enterprise lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.
2. If they don't match what you want, click the left-pointing bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

Splunk Enterprise then loads a confirmation page and begins indexing the specified performance metrics. For more information on getting data from files and directories, see [Monitor Windows performance](#) in this manual.

Configure local Windows performance monitoring with configuration files

The `inputs.conf` configuration file controls performance monitoring configurations. To set up performance monitoring using configuration files, you must create or edit `inputs.conf` in `%SPLUNK_HOME%\etc\system\local` on the Windows machine where you want to collect the performance metrics. If you haven't worked with configuration files before, see [About configuration files](#).

The option to configure local Windows monitoring is available for both Splunk Cloud Platform instances that receive forwarded data and Splunk Enterprise instances.

The `[perfmon://<name>]` stanza defines performance monitoring inputs in `inputs.conf`. You specify one stanza per performance object that you want to monitor.

In each stanza, you can specify the following settings:

Setting	Required?	Description
<code>interval</code>	Yes	How often, in seconds, to poll for new data. If this setting is not present, the input runs every 300 seconds (5 minutes).
<code>object</code>	Yes	The performance objects that you want to capture. Specify either a string that exactly matches the name of an existing Performance Monitor object, or use a regular expression to reference multiple objects. If this setting isn't present and defined, the input can't run because there is no default.
<code>counters</code>	Yes	One or more valid performance counters that are associated with the object specified in the <code>object</code> setting. Separate multiple counters with semicolons. You can also use an asterisk (*) to specify all available counters under a given <code>object</code> . If this setting isn't present and defined, the input can't run because there is no default.
<code>instance</code>	No	One or more valid instances associated with the performance counter specified in the <code>counters</code> setting. Multiple instances are separated by semicolons. Specify all instances by using an asterisk (*), which is the default if you don't define the setting in the stanza.
<code>index</code>	No	The index to route performance counter data to. If this setting isn't defined, the default index is used.
<code>disabled</code>	No	Whether or not to gather the performance data defined in this input. Set this setting to <code>1</code> to disable this stanza, and <code>0</code> to enable it. If the setting isn't defined, it defaults to <code>0</code> .

The following table shows advanced options:

Setting	Required?	Description
<code>showZeroValue</code>	No	Whether or not Splunk Enterprise should collect events that have values of zero. Set this setting to <code>1</code> to collect zero-value events, and <code>0</code> to ignore these events. If not present, it defaults to <code>0</code> .
<code>samplingInterval</code>	No	How often, in milliseconds, that Splunk Enterprise is to collect performance data.

Setting	Required?	Description
		Enables high-frequency performance sampling. When you enable high-frequency performance sampling, Splunk Enterprise collects performance data every interval and reports the average of the data as well as other statistics. It defaults to 100 milliseconds, and must be less than what you specify with the <code>interval</code> setting.
<code>stats</code>	No	<p>A semicolon-separated list of statistic values that Splunk Enterprise reports for high-frequency performance sampling.</p> <p>Allowed values are <code>average</code>, <code>min</code>, <code>max</code>, <code>dev</code>, and <code>count</code>.</p> <p>The default is no setting.</p>
<code>mode</code>	No	<p>When you enable high-performance sampling, this setting controls how Splunk Enterprise outputs events.</p> <p>Allowed values are <code>single</code>, <code>multikv</code>, <code>multiMS</code>, and <code>multikvMS</code>.</p> <p>When you enable either <code>multiMS</code> or <code>multikvMS</code>, Splunk Enterprise outputs two events for each performance metric it collects. The first event is the average value, and the second is the statistics event. The statistics event has a special sourcetype depending on which output mode you use: <code>perfmonMSStats</code> for <code>multiMS</code> and <code>perfmonMKMSStats</code> for <code>multikvMS</code>.</p> <p>If you don't enable high-performance sampling, the <code>multikvMS</code> output mode is the same as the <code>multikv</code> output mode.</p> <p>The default is <code>single</code>.</p>
<code>useEnglishOnly</code>	No	<p>Controls how Splunk Enterprise indexes performance metrics on systems whose locale isn't English. Specifically, this setting dictates which Windows Performance Monitor API to use when Splunk Enterprise indexes performance metrics on hosts that don't use the English language.</p> <p>If set to <code>true</code>, Splunk Enterprise collects the performance metrics in English regardless of the system locale. It uses the PdhAddEnglishCounter() API to add the counter string. It also disables regular expression and wildcard matching for the object and counter settings.</p> <p>If set to <code>false</code>, Splunk Enterprise collects the performance metrics in the system language and expects you to configure the <code>object</code> and <code>counter</code> settings in that language. It uses the PdhAddCounter() API to add the counter string. You can use wildcards and regular expressions, but you must specify valid <code>object</code>, <code>counters</code>, and <code>instances</code> values that are specific to the locale of the operating system.</p> <p>The default is <code>false</code>.</p>
<code>useWinApiProcStats</code>	No	When enabled, the <code>useWinApiProcStats</code> setting in the Performance Monitor input uses process kernel mode and user mode times to calculate CPU usage for a process. Currently, the input uses the standard Performance Data Helper (PDH) APIs to calculate CPU usage for a process.

Setting	Required?	Description
		<p>When this setting is configured to <code>true</code>, the input uses the <code>GetProcessTime()</code> function in the core Windows API to calculate CPU usage for a process for the following Performance Monitor counters:</p> <ul style="list-style-type: none"> • Processor Time • User Time • Privileged Time <p>It's a best practice to enable the <code>useWinApiProcStats</code> function on multicore Windows machines.</p> <p>Identify how many processors are in your system. The total number of cores can be verified on the System Information page in your Windows deployment.</p> <p>Each processor core in your system is equal to a maximum processor performance percentage of 100%. So for each core in a multicore windows deployment, 100% is added to the total maximum available processor performance percentage. For example, an 8 core windows environment will have a maximum process capability of 800%.</p> <p>Total processor capability can be validated in your Splunk Enterprise deployment by navigating to Data inputs > Local performance monitoring > Select system.</p> <div> <p>The APIs that this setting uses are English only. If your Windows machine uses a non-English system locale, you must also set <code>useEnglishOnly</code> to <code>true</code>.</p> </div>

See Performance Monitor inputs show maximum values of 100 percent usage for a process on multicore Microsoft Windows machines in the release notes for more information on calculating CPU usage on Windows multicore machines.

`formatStringNoControls` how Splunk Enterprise formats the output of floating-point values for performance counter events.

Windows often prints performance counter events as floating point values. When not formatted, the events print with all significant digits to the right of the decimal point. The `formatString` setting controls the number of significant digits that print as part of each event.

The setting uses format specifiers from the C++ language `printf` function. The function includes many kinds of specifiers, depending on how you want to output the event text.

When specifying the format, do not use quotation marks (") around the format. Specify only the valid characters needed to format the string the way you want.

The default is `%.20g`.

Collect performance metrics in English regardless of system locale

You can collect performance metrics in English even if the system that Splunk Enterprise runs on doesn't use the English language.

To do this, use the `useEnglishOnly` setting in stanzas within `inputs.conf`. There is no way to configure the `useEnglishOnly` setting in Splunk Web.

There are caveats to using `useEnglishOnly` in an `inputs.conf` stanza. See Caveats later in this topic.

Examples of performance monitoring input stanzas

Following are some example stanzas that show you how to use the `inputs.conf` configuration file to monitor performance monitor objects.

```
# Query the PhysicalDisk performance object and gather disk access data for
# all physical drives installed in the system. Store this data in the
# "perfmon" index.
# Note: If the interval setting is set to 0, Splunk resets the interval
# to 1.
```

```
[perfmon://LocalPhysicalDisk]
interval = 0
object = PhysicalDisk
counters = Disk Bytes/sec; % Disk Read Time; % Disk Write Time; % Disk Time
instances = *
disabled = 0
index = PerfMon
```

```
# Gather SQL statistics for all database instances on this SQL server.
# 'object' setting uses a regular expression "\$.*" to specify SQL
# statistics for all available databases.
```

```
[perfmon://SQLServer_SQL_Statistics]
object = MSSQL\$.*:SQL Statistics
counters = *
instances = *
```

```
# Gather information on all counters under the "Process" and "Processor"
# Perfmon objects.
# We use '.*' as a wild card to match the 'Process' and 'Processor' objects.
```

```
[perfmon://ProcessandProcessor]
object = Process.*
counters = *
instances = *
```

```
# Collect CPU processor usage metrics in English only on a French system.
```

```
[perfmon://Processor]
object = Processor
instances = _Total
counters = % Processor Time;% User Time
useEnglishOnly = 1
interval = 30
disabled = 0
```

```
# Collect CPU processor usage metrics in the French system's native locale.
# Note that you must specify the counters in the language of that locale.
```

```
[perfmon://FrenchProcs]
```

```

counters = *
disabled = 0
useEnglishOnly = 0
interval = 30
object = Processeur
instances = *

# Collect CPU processor usage metrics. Format the output to two decimal places only.
[perfmon://Processor]
counters = *
disabled = 0
interval = 30
object = Processor
instances = *
formatString = %.20g

```

Important information about specifying performance monitor objects in the inputs.conf file

When you use the inputs.conf configuration file to configure Windows performance monitor inputs, you must take special care in ensuring that the file contains the correct syntax for the inputs, or the Splunk platform will not index the data correctly.

Use all lowercase letters when specifying the perfmon keyword

When you create a performance monitor input in the inputs.conf file, you must use all lowercase letters for the `perfmon` keyword. See the following example:

Correct)

```
[perfmon://CPUTime]
```

Incorrect

```
[Perfmon://CPUTime]
```

```
[PERFMON://CPUTime]
```

If you use capital or mixed-case letters for the keyword, the Splunk platform warns of the problem on start up, and the specified performance monitor input doesn't run.

Specify valid regular expressions to capture multiple performance monitor objects

To specify multiple objects in a single performance monitor stanza, you must use a valid regular expression to capture those objects. For example, to specify a wildcard to match a string beyond a certain number of characters, do not use an asterisk (*), but rather a period followed by an asterisk (. *). If the object contains a dollar sign or similar special character, you might need to escape it with a backslash (\).

Values must exactly match what is in the Performance Monitor API if you don't use regular expressions

When you specify values for the `object`, `counters`, and `instances` settings in the `[perfmon://]` stanzas, confirm that those values exactly match those defined in the Performance Monitor API, including case, or else the input might return incorrect data or no data at all. If the input cannot match a performance object, counter, or instance value that you've specified, it logs that failure to the splunkd.log file. See the following example of a failed return:

```

01-27-2011 21:04:48.681 -0800 ERROR ExecProcessor - message from "C:\Program
Files\Splunk\bin\splunk-perfmon.exe" -noui" splunk-perfmon - PerfmonHelper::enumObjectByNameEx:
PdhEnumObjectItems failed for object - 'USB' with error (0xc0000bb8): The specified object is not found on
the system.

```

Use Splunk Web to add performance monitor data inputs to ensure that you add them correctly.

Enable remote Windows performance monitoring over WMI

You can configure remote performance monitoring either in Splunk Web or by using configuration files.

When you collect performance metrics over WMI, you must configure the Splunk platform instance to run as an Active Directory (AD) user with appropriate access for remote collection of performance metrics. You must do this before attempting to collect those metrics. Both the machine that runs the Splunk platform instance and the machines the Splunk platform collects performance data from must reside in the same AD domain or forest.

WMI self-throttles by design to prevent denial-of-service attacks. The Splunk platform also reduces the number of WMI calls it makes over time as a precautionary measure if these calls return an error. Depending on the size, configuration, and security profile of your network, installing a local forwarder on the host that you want to collect performance metrics might be a better choice. See [Considerations for deciding how to monitor remote Windows data](#) in this manual.

WMI-based performance values versus Performance Monitor values

When you gather remote performance metrics through WMI, some metrics return zero values or values that are not in line with values that Performance Monitor returns. A limitation in the implementation of WMI for performance monitor counters causes this problem. This is not an issue with the Splunk platform or how it retrieves WMI-based data.

WMI uses the `Win32_PerfFormattedData_*` data classes to gather performance metrics. Find more information about Win32 classes at [https://docs.microsoft.com/en-us/previous-versions//aa394084\(v=vs.85\)?redirectedfrom=MSDN](https://docs.microsoft.com/en-us/previous-versions//aa394084(v=vs.85)?redirectedfrom=MSDN).

WMI defines the data structures within these classes as either 32- or 64-bit unsigned integers, depending on the version of Windows you run. The Windows Performance Data Helper (PDH) API defines Performance Monitor objects as floating-point variables. A floating-point variable means that you might see WMI-based metrics that appear anomalous, due to rounding factors.

For example, if you collect data on the Average Disk Queue Length Performance Monitor counter at the same time you collect the `Win32_PerfFormattedData_PerfDisk_PhysicalDisk\AvgDiskQueueLength` metric through WMI, the WMI-based metric might return zero values even though the Performance Monitor metric returns values that are greater than zero but less than 0.5. This is because WMI rounds the value down before displaying it.

If you require additional granularity in your performance metrics, configure the performance monitoring inputs on a universal forwarder on each machine from which you want to collect performance data. You can then forward that data to an indexer. Data retrieved using this method is more reliable than data gathered remotely using WMI-based inputs.

Configure remote Windows performance monitoring with Splunk Web

This option is available on Splunk Enterprise only, It isn't available on Splunk Cloud instances. You can instead configure a universal forwarder in Splunk Enterprise and forward that data to the Splunk Cloud instance.

In Splunk Enterprise, go to the Add New page in Splunk Web through either Splunk Settings or Splunk Home.

To access the Add New page through Splunk Settings, follow these steps:

1. Click **Settings** in the upper-right corner of Splunk Web.
2. Click **Data Inputs**.
3. Click **Remote performance monitoring**.
4. Click **New** to add an input.

To access the Add New page through Splunk Home, follow these steps:

1. Click the **Add Data** link in Splunk Home.
2. Click **Monitor** to monitor performance data from the local Windows machine, or **Forward** to forward performance data from another Windows machine. Splunk Enterprise loads the **Add Data - Select Source** page.

Forwarding performance data requires additional setup.

3. In the left pane, locate and select **Local Performance Monitoring**.

Select the input source

Win32_PerfFormattedData_* classes don't show up as available objects in Splunk Web. If you want to monitor Win32_PerfFormattedData_* classes, you must add them directly in the wmi.conf file. See [Configure remote Windows performance monitoring with configuration files](#) for more information. Follow these steps:

1. In the left pane of Splunk Enterprise, select Local Performance Monitoring.
2. In the **Collection Name** field, enter a unique name for this input that you will remember.
3. In the **Select Target Host** field, enter the host name or IP address of the Windows computer you want to collect performance data from.
4. Click **Query** to get a list of the performance objects available on the Windows machine you specified in the **Select Target Host** field.
5. Choose the object that you want to monitor from the **Select Class** list. Splunk Enterprise displays the **Select Counters** and **Select Instances** list boxes.
6. In the **Select Counters** list box, locate the performance counters you want this input to monitor.

You can add only one performance object per data input. This is due to how Microsoft handles performance monitor objects. Many objects enumerate classes that describe themselves dynamically upon selection. This can lead to confusion as to which performance counters and instances belong to which object, as defined in the input. If you need to monitor multiple objects, create additional data inputs for each object.

7. Click once on each counter you want to monitor. Splunk Enterprise moves the counter from the **Available counter(s)** window to the **Selected counter(s)** window.
8. To unselect a counter, click its name in the **Available Items** window. Splunk Enterprise moves the counter from the **Selected counter(s)** window to the **Available counter(s)** window.
9. To select or unselect all of the counters, click the **add all** or **remove all** links.

Selecting all of the counters can result in the indexing of a lot of data, possibly more than your license allows.

10. In the **Select Instances** list box, select the instances that you want this input to monitor by clicking once on the instance in the **Available instance(s)** window. Splunk Enterprise moves the instance to the **Selected instance(s)** window.
11. In the **Polling interval** field, enter the time, in seconds, between polling attempts for the input.

The **_Total** instance is a special instance, and appears for many types of performance counters. This instance is the average of any associated instances under the same counter. Data collected for this instance can be significantly different than for individual instances under the same counter. For example, when you monitor performance data for the Disk Bytes/Sec performance counter under the PhysicalDisk object on a system with two disks, the available instances include one for each physical disk (0 C: and 1 D:) and the **_Total** instance, which is the average of the two physical disk instances.

12. Click **Next**.

Specify input settings

Specify application context, default host value, and index in the Input Settings page. All of these parameters are optional.

Setting the **Host** value sets the **host** field only in the resulting events. It doesn't direct Splunk Enterprise to look on a specific host on your network.

1. Select the appropriate **Application context** for this input.
2. Set the **Host** value. You have several choices for this setting. Learn more about setting the host value in [About hosts](#).
3. Set the **Index** that Splunk Enterprise should send data to. Leave the value as `default`, unless you have defined multiple indexes to handle different types of events.
4. Click the **Review** button.

Review your choices

After you specify input settings, review your selections. Splunk Enterprise lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.
2. If they don't match what you want, click the left-pointing bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

Splunk Enterprise then loads a confirmation page and begins indexing the specified performance metrics. For more information on getting data from files and directories, see [Monitor Windows performance](#) in this manual.

Configure remote Windows performance monitoring with configuration files

The wmi.conf configuration file controls remote performance monitoring configurations. To set up remote performance monitoring using configuration files, create or edit wmi.conf in %SPLUNK_HOME%\etc\system\local. If you haven't worked with configuration files before, read [About configuration files](#) before you begin.

For Splunk Cloud instances, install a universal forwarder on the machine where you want to collect the performance data, and configure that forwarder to send the data to Splunk Cloud. On Splunk Enterprise instances, use Splunk Web to create remote performance monitor inputs unless you do not have access to it. The names of performance monitor objects, counters, and instances must exactly match what the Performance Monitor API defines, including case. Splunk Web uses WMI to get the properly formatted names, eliminating the potential for typos.

The wmi.conf file contains one stanza for each remote performance monitor object that you want to monitor. In each stanza, you specify the following settings:

Global settings

Setting	Required?	Description	Default
initial_backoff	No	How long, in seconds, to wait before retrying a connection to a WMI provider when an error occurs. If problems persist on connecting to the provider, then the wait time between connection attempts doubles until either it can connect or	5

Setting	Required?	Description	Default
		until the wait time is greater than or equal to the <code>max_backoff</code> setting.	
<code>max_backoff</code>	No	The maximum amount of time, in seconds, to attempt to reconnect to a WMI provider.	20
<code>max_retries_at_max_backoff</code>	No	How many times, after <code>max_backoff</code> seconds has been reached between reconnection attempts with a WMI provider, to continue to attempt to reconnect to that provider.	2
<code>checkpoint_sync_interval</code>	No	How long, in seconds, to wait for state data to be flushed to disk.	2

Input-specific settings

Setting	Required?	Description	Default
<code>interval</code>	Yes	How often, in seconds, to poll for new data. If this setting isn't present, the input can't run because there is no default.	N/A
<code>server</code>	No	A comma-separated list of one or more valid hosts on which you want to monitor performance.	The local machine
<code>event_log_file</code>	No	The names of one or more Windows event log channels to poll. This setting configures Splunk Enterprise that the incoming data is in event log format. Do not use the <code>event_log_file</code> setting in a stanza that already contains the <code>wql</code> setting.	N/A
<code>wql</code>	No	A valid Windows Query Language (WQL) statement that specifies the performance objects, counters, and instances you want to poll remotely. This setting tells Splunk Enterprise to expect data from a WMI provider. Don't use the <code>wql</code> setting in a stanza that already contains the <code>event_log_file</code> setting.	N/A
<code>namespace</code>	No	The namespace in which the WMI provider you want to query resides. The value for this setting can be either relative, such as <code>Root\CIMV2</code> or absolute, such as <code>\\SERVER\Root\CIMV2</code> , but it must be relative if you specify the <code>server</code> setting. Use the <code>namespace</code> setting only in a stanza that contains the <code>wql</code> setting.	<code>Root\CIMV2</code>
<code>index</code>	No	The desired index to route performance counter data to.	<code>default</code>
<code>current_only</code>	No	The characteristics and interaction of WMI-based event collections based on whether the <code>wql</code> setting or the <code>event_log_file</code> setting is defined: <ul style="list-style-type: none"> If <code>wql</code> is defined, this setting tells Splunk Enterprise whether or not to expect an event notification query. Set to <code>1</code> to expect an event notification query, or <code>0</code> to expect a standard query. If <code>event_log_file</code> is defined, this setting tells Splunk Enterprise whether or not to capture events that occur only when Splunk Enterprise is running. Set to <code>1</code> to only capture events that occur when Splunk Enterprise is running, or <code>0</code> to gather events from the last checkpoint or, if no checkpoint exists, the oldest events available. 	N/A
<code>disabled</code>	No	Tells Splunk Enterprise whether or not to gather the performance data defined in this input. Set to <code>1</code> to disable performance monitoring for this stanza, or <code>0</code> to enable it.	0

Examples of using wmi.conf

The following example of wmi.conf gathers local disk and memory performance metrics and places them into the wmi_perfmon index:

```
[settings]
initial_backoff = 5
max_backoff = 20
max_retries_at_max_backoff = 2
checkpoint_sync_interval = 2

# Gather disk and memory performance metrics from the local system every second.
# Store event in the "wmi_perfmon" Splunk index.

[WMI:LocalPhysicalDisk]
interval = 1
wql = select Name, DiskBytesPerSec, PercentDiskReadTime, PercentDiskWriteTime, PercentDiskTime from \
  Win32_PerfFormattedData_PerfDisk_PhysicalDisk
disabled = 0
index = wmi_perfmon

[WMI:LocalMainMemory]
interval = 10
wql = select CommittedBytes, AvailableBytes, PercentCommittedBytesInUse, Caption from \
  Win32_PerfFormattedData_PerfOS_Memory
disabled = 0
index = wmi_perfmon
```

Additional information on WQL query statements

WQL queries must be structurally and syntactically correct. If they aren't, you might get undesirable results or no results at all. When writing event notification queries by specifying `current_only=1` in the stanza in which a WQL query resides, your WQL statement must contain one of the clauses that specify such a query: (`WITHIN`, `GROUP`, or `HAVING`). See <https://docs.microsoft.com/en-us/windows/win32/wmisdk/querying-with-wql?redirectedfrom=MSDN> on the Microsoft website for more information.

Splunk Web eliminates problems with WQL syntax by generating the appropriate WQL queries when you use it to create performance monitor inputs.

Caveats to using the performance monitoring input

When you use the Windows performance monitor input to collect performance monitoring data from Windows machines, mind the following caveats:

Increased memory usage during collection of performance metrics

When you collect data on some performance objects, such as the `Thread` object and its associated counters, you might notice increased memory usage in your Splunk Enterprise deployment. This increase in usage is normal, as certain performance objects consume more memory than others during the collection process.

Processor Time counters don't return values higher than 100

Due to how Microsoft tallies CPU usage with the `Processor:% Processor Time` and `Process:% Processor Time` counters, these counters don't return a value higher than 100 regardless of the number of CPUs or cores in the system. This return is by design. These counters subtract the amount of time spent on the idle process from 100%.

Limitations to the useEnglishOnly setting

When you edit the `inputs.conf` file on a non-English system to enable performance monitoring, there are some limitations to how the `useEnglishOnly` setting works.

If you set the setting to `true`, you cannot use wildcards or regular expressions for the `object` and `counters` settings. These settings must contain specific entries based on valid English values as defined in the Performance Data Helper library. You can specify a wildcard for the `instances` setting. Here's an example:

```
[perfmon://Processor]
object = Processor
instances = _Total
counters = % Processor Time;% User Time
useEnglishOnly = 1
interval = 30
disabled = 0
```

The `counters` setting contain values in English even though the system language is not English.

If you set the setting to `false`, you can use wildcards and regular expressions for these settings, but you must specify values based on the operating system's language. An example of a stanza on a system running in French follows:

```
[perfmon://FrenchProcs]
counters = *
disabled = 0
useEnglishOnly = 0
interval = 30
object = Processeur
instances = *
```

Note in this example that the `object` setting has been set to `Processeur`, which is the French equivalent of `Processor`. If you specify English values here, Splunk Enterprise will not find the performance object or instance.

Additional impacts of using the useEnglishOnly setting

There are additional items to consider when using the setting.

- When you use Splunk Web to create performance monitor inputs on a non-English operating system, it always specifies `useEnglishOnly = false`.
- Additionally, you can enable, disable, clone, or delete these stanzas within Splunk Web. You cannot, however, edit them in Splunk Web unless the operating system's locale matches the locale specified in the stanza.
- You can use Splunk Web to enable, disable, clone, or delete a performance monitor stanza with the `useEnglishOnly` setting set to `true`. However, you cannot edit them in Splunk Web unless the system's locale is English.

Monitor Windows data with PowerShell scripts

PowerShell is a scripting language that comes with many versions of Windows. It lets you handle Windows operations from a command-line interface. You can create scripts with the language and output the results of those scripts as objects to other scripts.

The Splunk platform supports monitoring events received through PowerShell scripts. You can use the PowerShell input to run a single PowerShell command or reference a PowerShell script. The Splunk platform then indexes the output of

these commands or scripts as events.

If you use Splunk Cloud Platform and want to monitor script output, use the universal forwarder to ingest the output from a Windows machine and forward it to your Splunk Cloud Platform deployment.

Requirements

- Splunk Cloud Platform must receive Windows data that comes from PowerShell scripts from a universal forwarder that is installed on a Windows machine.
- The Splunk platform instance must run on Windows. See *Install on Windows* in the *Installation Manual*.
- The Splunk platform instance must be configured to use the Local System user to run all PowerShell scripts.
- PowerShell version 3.0 or higher must be installed on the machine. Microsoft .NET version 4.5 or higher must be installed on the machine.
- There might be additional requirements to run PowerShell scripts depending on the version of Windows and PowerShell. See the Microsoft documentation on PowerShell for details.

Configure inputs with configuration files

1. Write a PowerShell command or script to capture the information you want.
2. On the Splunk platform instance that will run the script, open a PowerShell window.
3. Create an `inputs.conf` configuration file in the `%SPLUNK_HOME%\etc\system\local` directory.
4. Open the `inputs.conf` file and edit it to enable a Windows PowerShell input.
5. In the input, specify the command or the full path to your script with the `script` setting.
6. (Optional) Specify a schedule on which the command or script will run with the `schedule` setting.
7. Save the `inputs.conf` file.
8. Restart the Splunk platform to enable the input.

PowerShell input configuration values

The Splunk platform uses the following stanzas in the `inputs.conf` file to monitor data gathered by PowerShell.

Setting	Description	Default
<code>script</code>	The PowerShell command or script file to run. When you specify a script file (.ps1), prepend the script name with a period and a space (.).	n/a
<code>schedule</code>	How often the command or script runs. You can specify either a number to indicate the interval, in seconds, or a valid <code>cron</code> schedule format.	Script runs once
<code>disabled</code>	Whether or not to enable the input. Set to 1 to disable and 0 to enable.	0 (enabled)

For information on writing PowerShell scripts, see [Write scripts for the PowerShell input](#) later in this topic.

Single command configuration example

This example runs the `Get-Process` cmdlet and pipes that output to the `Select-Object` cmdlet using the host name that Splunk software is installed on as an argument. It runs the command every 5 minutes.

```
[powershell://Processes-EX1]
script = Get-Process | Select-Object Handles, NPM, PM, WS, VM, Id, ProcessName,
@{n="SplunkHost";e={$Env:SPLUNK_SERVER_NAME}}
schedule = */5 * * * *
sourcetype = Windows:Process
```

Script configuration example

This example runs the `getprocesses.ps1` script located in the `%SPLUNK_HOME\etc\apps\My-App` directory. It sets the source type for these events to `Windows:Process`. The script runs every 20 minutes from 9:00 AM to 4:40 PM on Mondays to Fridays.

```
[powershell://Processes-EX2]
script = . "$SplunkHome\etc\apps\My-App\bin\getprocesses.ps1"
schedule = */20 * 9-16 * 1-5
sourcetype = Windows:Process
```

Configure inputs with Splunk Web

If you use Splunk Enterprise, you can configure inputs with Splunk Web. To configure the PowerShell input on universal forwarders, see [Configure inputs with configuration files](#) earlier in this topic.

Follow these steps to configure inputs with Splunk Web:

1. In Splunk Web, select **Settings > Data inputs**.
2. Select **PowerShell v3 modular input**.
3. Click **New**.
4. Enter an input name in the **Name** field.
5. Enter a command or path to a script in the **Command or Script Path** field.
6. Enter an interval or cron schedule in the **Cron Schedule** field.
7. Click the **More Settings** checkbox to select the source type, host, and default index.
8. Click **Next**.

Write scripts for the PowerShell input

The Splunk platform provides one modular PowerShell input handler. The PowerShell handler supports Microsoft PowerShell version 3 and higher.

The PowerShell modular input provides a single-instance, multi-threaded script host that provides a supporting schema, XML configuration through the `stdin` input/output data stream, and XML streaming output. See [Create custom data inputs for Splunk Cloud Platform or Splunk Enterprise on the Splunk Developer Portal](#).

You can define many PowerShell stanzas in an input configuration file and run them simultaneously. You can schedule each stanza using the `cron` syntax. Because all scripts run within the same process, scripts share environment variables such as the current working directory.

The input doesn't set a host variable in your PowerShell environment. When you write a script for the input, do not refer to `$host` or use the `Write-Host` or `Out-Host` PowerShell cmdlets. Instead, use either the `Write-Output` or `Write-Error` cmdlets.

The input converts all output to key/value pairs based on public properties that are defined in the schema.

The Splunk platform also includes a PowerShell module called `LocalStorage`, which exposes three cmdlets:

- `Get-LocalStoragePath`
- `Export-LocalStorage`
- `Import-LocalStorage`

These cmdlets use the Splunk platform checkpoint directory and let you maintain key/value pairs of data between scheduled runs of your script. Normally, data does not persist from one invocation to the next.

Specify paths

The input sets the `$SplunkHome` PowerShell variable so you can address scripts in add-ons by writing paths like this:

```
[powershell://MSExchange_Health]
script=. $SplunkHome/etc/apps/TA-Exchange-2010/powershell/health.ps1
```

Besides `$SplunkHome`, there are several other read-only constant variables:

Variable	Description
<code>SplunkServerName</code>	The name configured for this machine to use in events
<code>SplunkServerUri</code>	The Splunk platform REST API address
<code>SplunkSessionKey</code>	The session key or authentication token needed for accessing the Splunk platform REST API
<code>SplunkCheckpointPath</code>	The path for storing persistent state
<code>SplunkServerHost</code>	The name of the Splunk platform instance that you want to communicate with
<code>SplunkStanzaName</code>	The name of the inputs.conf stanza that defined this script

Handle output of PowerShell scripts

The Splunk platform takes each object that your script produces as an output and turns it into an event, wrapped in `<event>` and `<data>` tags. The Splunk platform converts the properties of each object into key/value pairs. However, the value can only be a quoted string, converted by calling the `.ToString()` method. Thus, the output must be simple, and you must flatten any complex nested objects in your script before the script outputs them.

The following special property names have significance for the Splunk platform modular inputs and let you override the defaults in the inputs.conf stanza:

Property	Description
<code>SplunkIndex</code>	Overrides the index that the output is stored in.
<code>SplunkSource</code>	Overrides the <code>source</code> for the output.
<code>SplunkHost</code>	Overrides the <code>host</code> name for the output.
<code>SplunkSourceType</code>	Overrides the <code>sourcetype</code> for the output.
<code>SplunkTime</code>	Overrides the <code>time</code> . If you do not specify this, all objects that your script generates in a single execution get roughly the same timestamp. This is because the script holds the objects for output until it finishes executing, and then marks the objects with the output time. You must specify this value in epoch or POSIX time, which is a positive integer that represents the seconds that elapsed

Property	Description
	since 0:00 UTC on Thursday, January 1, 1970.

These properties never appear as objects in the key/value output.

If you want to set these properties and override the defaults, use a calculated expression with the `Select-Object` cmdlet or use the `Add-Member` cmdlet to add a `NoteProperty` property.

Caveats for handling PowerShell script output

The input currently requires that any PowerShell scripts it executes produce output objects that do not have any script properties. Pipe output through the `Select-Object` cmdlet to ensure proper formatting.

The input currently doesn't process the output of scripts until your pipeline and runspace are finished. This means the input does not process `ScriptProperty` values. It also means that all of your output essentially has the same timestamp, unless you override it using the `SplunkTime` variable.

When writing your scripts, avoid long-running scripts. Do not write scripts that wait for things to happen unless the scripts exit every time there is output.

Monitor Windows host information

You can monitor detailed statistics about your local Windows machine with the Splunk platform.

If you use Splunk Cloud Platform, you must collect Windows host information with a forwarder and send it to your Splunk Cloud Platform deployment. Follow these high-level steps:

1. Install the universal forwarder on the Windows machine that you want to collect the host information.
2. Install the app to connect the universal forwarder to the Splunk Cloud Platform instance.
3. Configure the forwarder to collect the Windows host information.

Both full instances of Splunk Enterprise and universal forwarders support direct, local collection of host information. On these instance types, the Windows host monitor input runs as a process called `splunk-winhostmon.exe`. This process runs once for every Windows host monitoring input that you define at the interval that you specify in the input. On Splunk Enterprise, you can configure host monitoring using Splunk Web, and on the universal forwarder you can configure the inputs using the `inputs.conf` configuration file.

Why monitor host information?

You can monitor hosts to get detailed information about your Windows machines. You can monitor changes to the system, such as installation and removal of software, the starting and stopping of services, and uptime. When a system failure occurs, you can use Windows host monitoring information as a first step into the forensic process. With the Splunk Search Processing Language, you can give your team statistics on all machines in your Windows network.

The Splunk platform can collect the following information about a Windows machine:

General computer

The make and model of the computer, its host name, and the Active Directory domain it is in.

Operating system

	The version and build number of the operating system and service packs installed on the computer, the computer name, the last time it started, the amount of installed and free memory, and the system drive.
Processor	The make and model of the CPUs installed in the system, their speed and version, the number of processors and cores, and the processor ID.
Disk	A list of all drives available to the system and, if available, their file system type and total and available space.
Network adapter	Information about the installed network adapters in the system, including manufacturer, product name, and MAC address.
Service	Information about the installed services on the system, including name, display name, description, path, service type, start mode, state, and status.
Process	Information on the running processes on the system, including the name, the command line with arguments), when they were started, and the executable path.

Requirements

To monitor host information, you must fulfill the following requirements:

- Splunk Cloud Platform must receive Windows host information from a forwarder.
- The forwarder must run on Windows. See *Install on Windows* in the *Installation Manual*.
- To read all Windows host information locally, the forwarder must run as the Local System Windows user or a local administrator user.

Security and remote access considerations

The universal forwarder must run as the Local System user to collect Windows host information by default.

Where possible, use a universal forwarder to send Windows host information from remote machines to Splunk Cloud Platform or a Splunk Enterprise indexer. You must use a universal forwarder to send Windows host information to Splunk Cloud Platform. Review the Forwarder Manual for information about how to install, configure, and use the universal forwarder to collect Windows host data.

If you choose to install forwarders on your remote machines to collect Windows host data, then you can install the forwarder as the Local System user on these machines. The Local System user has access to all data on the local machine, but not on remote machines.

If you run Splunk Enterprise or the universal forwarder as a user other than the Local System user, then that user must have local administrator rights and other permissions on the machine that you want to collect host data. See *Choose the Windows user Splunk Enterprise should run as* in the *Installation Manual*.

Use the inputs.conf configuration file to configure host monitoring

To collect Windows host information on your Splunk Cloud Platform instance, you must configure a universal forwarder on the Windows machine that you want to collect host information. Then, you can send the data to Splunk Cloud Platform.

You can edit `inputs.conf` to configure host monitoring. For more information on how to edit configuration files, see *About configuration files* in the *Admin Manual*.

You can also configure this file directly on a Splunk Enterprise instance.

To configure host monitoring on `inputs.conf`, follow these steps:

1. On the machine that you want to collect Windows host information, install a universal forwarder.
2. Download and install the Splunk Cloud Platform universal forwarder credentials package on the forwarder.
3. On the forwarder, use a text editor to create an `inputs.conf` configuration file in `%SPLUNK_HOME%\etc\system\local` and open it for editing.
4. In the same text editor, open `%SPLUNK_HOME%\etc\system\default\inputs.conf` and review it for the Windows event log inputs you want to enable.
5. Copy the Windows event log input stanzas you want to enable from `%SPLUNK_HOME%\etc\system\default\inputs.conf`.
6. Paste the stanzas you copied into the `%SPLUNK_HOME%\etc\system\local\inputs.conf` file.
7. Make edits to the stanzas that you copied to the `%SPLUNK_HOME%\etc\system\local\inputs.conf` file to collect the Windows event log data you want.
8. Save the `%SPLUNK_HOME%\etc\system\local\inputs.conf` file and close it.
9. Restart the universal forwarder.

When the Splunk platform indexes data from Windows host monitoring inputs, it sets the **source** for received events to `windows`. It sets the **source type** of the incoming events to `WinHostMon`.

Windows host monitor configuration values

Splunk Enterprise and the universal forwarder use the following settings in the `inputs.conf` configuration file to monitor Windows host information.

Setting	Required?	Description
<code>interval</code>	Yes	How often, in seconds, to poll for new data. If you set the interval to a negative number, the Splunk platform runs the input one time. If you do not define this setting, the input does not run, as there is no default.
<code>type</code>	Yes	The type of host information to monitor. Can be one of <code>Computer</code> , <code>operatingSystem</code> , <code>processor</code> , <code>disk</code> , <code>networkAdapter</code> , <code>service</code> , <code>process</code> , or <code>driver</code> . The input does not run if this setting is not present.
<code>disabled</code>	No	Whether or not to run the input. If you set this setting to 1, then the platform instance does not run the input.

For examples, see Examples of Windows host monitoring configurations later in this topic.

Use Splunk Web to configure host monitoring

You can configure Windows host information on Splunk Web in Splunk Enterprise only. Follow these high-level steps to configure host monitoring through Splunk Web:

1. [Go to the Add Data page](#)
2. [Select the input source](#)
3. (Optional) [Specify input settings](#)
4. [Review your choices](#)

Go to the Add Data page

Follow these steps to get to the Add Data page from Settings:

1. Click **Settings > Data Inputs**.
2. Click **Files & Directories**.

3. Click **New Local File & Directory** to add an input.

Follow these steps to get to the Add Data page from your Splunk Enterprise home page:

1. Click **Add Data** on the page.
2. Click **Monitor** to monitor host information from the local Windows machine.

Select the input source

1. In the left pane, locate and select **Local Windows host monitoring**.
2. In the **Collection Name** field, enter a unique, memorable name for this input.
3. In the **Event Types** box, locate the host monitoring event types you want this input to monitor.
4. Click once on each type you want to monitor.
Splunk Enterprise moves the type from the **Available type(s)** window to the **Selected type(s)** window.
5. To deselect a type, click its name in the **Selected type(s)** window.
Splunk Enterprise moves the counter from the **Selected type(s)** window to the **Available type(s)** window.
6. (Optional) To select or deselect all of the types, click the **add all** or **remove all** links.

Selecting all of the types can index a lot of data and might exceed the data limits of your license.

7. In the **Interval** field, enter the time, in seconds, between polling attempts for the input.
8. Click **Next**.

Specify input settings

Go to the **Input Settings** page to specify the application context, default host value, and index. All of these parameters are optional.

1. Select the appropriate **Application context** for this input.
2. Set the **Host** name. You have several choices for this setting. For more about setting the host value, see [About hosts](#).

Host sets the host field only in the resulting events. It does not configure Splunk Enterprise to look on a specific host on your network.

3. Set the **Index** to send data to. Leave the value as **default**, unless you defined multiple indexes to handle different types of events. In addition to indexes for user data, the Splunk platform has multiple utility indexes, which also appear in this dropdown list.
4. Click **Review**.

Review your choices

After specifying all your input settings, review your selections. Splunk Enterprise lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.
2. If they do not match what you want, click the left-angle bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

Splunk Enterprise then loads the Success page and begins indexing the specified host information.

When Splunk Enterprise indexes data from Windows host monitoring inputs, it sets the **source** for received events to `windows`. It sets the **source type** of the incoming events to `WinHostMon`.

Examples of Windows host monitoring configurations

The following examples of how to configure Windows host monitoring in inputs.conf.

```
# Queries computer information.
[WinHostMon://computer]
type = Computer
interval = 300

# Queries OS information.
# 'interval' set to a negative number tells Splunk Enterprise to
# run the input once only.
[WinHostMon://os]
type = operatingSystem
interval = -1

# Queries processor information.
[WinHostMon://processor]
type = processor
interval = -1

# Queries hard disk information.
[WinHostMon://disk]
type = disk
interval = -1

# Queries network adapter information.
[WinHostMon://network]
type = networkAdapter
interval = -1

# Queries service information.
# This example runs the input every 5 minutes.
[WinHostMon://service]
type = service
interval = 300

# Queries information on running processes.
# This example runs the input every 5 minutes.
[WinHostMon://process]
type = process
interval = 300
```

Monitor Windows printer information

With the Splunk platform, you can monitor statistics about all of the printers and drivers, print jobs, and printer ports on your local Windows machine. You can collect the following print system information:

- **Printer.** Information on the print subsystem, such as the status of installed printers and when printers get added or deleted.
- **Job.** Information on print jobs, including who printed what, details on the jobs, and the status of existing jobs.
- **Driver.** Information on the print driver subsystem, including information on existing print drivers and when a print driver gets added or removed.
- **Port.** Information on printer ports installed on the system and when they get added or removed.

Both full instances of the Splunk platform and universal forwarders support local collection of printer subsystem information. If you use Splunk Cloud Platform and want to monitor printer subsystem information, use the universal forwarder to ingest the information and forward it to your Splunk Cloud Platform deployment.

The printer monitor input runs as a process called `splunk-winprintmon.exe`. This process runs once for every input you define at the interval you specify in the input. You can configure printer subsystem monitoring using Splunk Web or the `inputs.conf` configuration file.

Reasons to monitor printer information

Windows printer monitoring gives you detailed information about your Windows printer subsystem. You can monitor any changes to the system, such as installation and removal of printers, print drivers, and ports, the starting and completion of print jobs, and learn who printed what and when. When a printer failure occurs, you can use print monitoring information as a first step into the forensic process. With the Splunk search processing language, you can give your team at-a-glance statistics on all printers in your Windows network.

Requirements

Meet the following requirements before you monitor host information:

- The Splunk platform must run on Windows. See *Install on Windows* in the *Installation Manual*.
- The Splunk platform must run as the Local System user to read all local host information.

Security and remote access considerations

The Splunk platform must run as the Local System user to collect Windows print subsystem information by default.

Use a universal forwarder to send printer information from remote machines to an indexer. If you choose to install forwarders on your remote machines to collect printer subsystem data, then you can install the forwarder as the Local System user on these machines. The Local System user has access to all data on the local machine, but not on remote machines.

If you run the Splunk platform as a user other than the Local System user, then that user must have local Administrator rights to the machine and other permissions as detailed in *Choose the Windows user the Splunk platform should run as* in the *Installation Manual*.

Use Splunk Web to configure printer information

Follow these high-level steps to configure printer information on Splunk Web:

1. Go to the **Add Data** page.
2. Select the input source.
3. Specify input settings.
4. Review your choices.

Go to the Add Data page

Choose one of the following methods to get to the **Add Data** page.

To add data from the **Settings** drop-down list, follow these steps:

1. Click **Settings**.
2. Click **Data Inputs**.
3. Click **Local Windows print monitoring**.
4. Click **New** to add an input.

To add data from the Splunk Web home page, follow these steps:

1. Click **Add Data**.
2. Click **Monitor** to monitor print information from the local Windows machine.
3. In the left pane, locate and select **Local Windows print monitoring**.

Select the input source

1. In the **Collection Name** field, enter a unique and memorable name for this input.
2. In **Event Types**, locate the print monitoring event types you want this input to monitor.
3. Click each type you want to monitor once.
The Splunk platform moves the type from the Available type(s) window to the Selected type(s) window.
4. To deselect a type, click its name in the Selected type(s) window.
The Splunk platform moves the counter from the Selected type(s) window to the Available type(s) window.
5. (Optional) To select or deselect all of the types, click **Add all** or **Remove all**.

Selecting all of the types can result in the indexing of a lot of data, possibly more than your license allows.

6. In the **Baseline** control, click **Yes** to run the input as soon as it starts and no further. Click **No** to run the input at the interval specified in the **Interval (in minutes)** field.
7. Click **Next**.

Specify input settings

You can specify application context, default host value, and index on the Input Settings page. All of these parameters are optional.

1. Select the appropriate **Application context** for this input.
2. Set the **Host** name. You have several choices for this setting. Learn more about setting the host value in [About hosts](#).

Host only sets the host field in the resulting events. It does not direct the Splunk platform to look on a specific host on your network.

3. Set the **Index** that the Splunk platform will send data to. Leave the value as "default", unless you defined multiple indexes to handle different types of events. In addition to indexes for user data, the Splunk platform has a number of utility indexes, which also appear in this drop-down list.
4. Click **Review**.

Review your choices

After specifying all your input settings, review your selections. The Splunk platform lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.
2. If they do not match what you want, click the left-pointing angle bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

The Splunk platform loads the Success page and begins indexing the specified print information.

Use the inputs.conf configuration file to configure printer monitoring

You can edit the inputs.conf file to configure printer monitoring. Refer to the print monitoring configuration values and examples later in this topic.

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 inputs.conf file in this directory. You might need to create this file.
4. Add [WinPrintMon] configuration stanzas, settings, and values to enable Windows print monitoring inputs.
5. Save the file and close it.
6. Restart the Splunk platform.

For information on how to edit configuration files, see *About configuration files* in the *Admin Manual*.

Print monitoring configuration values

The Splunk platform uses the following settings in inputs.conf to monitor Windows printer subsystem information:

Attribute	Required?	Description
type	Yes	The type of host information to monitor. Can be <code>printer</code> , <code>job</code> , <code>driver</code> , or <code>port</code> . The input doesn't run if this variable isn't present.
baseline	No	Whether or not to generate a baseline of the existing state of the printer, job, driver, or port. If you set this attribute to <code>1</code> , then the Splunk platform writes a baseline. This might take additional time and CPU resources when the Splunk platform starts.
disabled	No	Whether or not to run the input. If you set this setting to <code>1</code> , then the Splunk platform does not run the input.

Examples of Windows printer monitoring configurations

The following examples show how to use the Windows printer monitoring configuration settings in inputs.conf.

```
# Monitor printers on system.
[WinPrintMon://printer]
type = printer
baseline = 0

# Monitor print jobs.
[WinPrintMon://job]
type = job
baseline = 1

# Monitor printer driver installation and removal.
[WinPrintMon://driver]
type = driver
baseline = 1

# Monitor printer ports.
[WinPrintMon://port]
type = port
baseline = 1
```

Fields for Windows print monitoring data

When the Splunk platform indexes data from Windows print monitoring inputs, it sets the **source** for received events to `windows`. It sets the **source type** of the incoming events to `WinPrintMon`.

Monitor Windows network information

With the Splunk platform, you can monitor detailed statistics about network activity into or out of a Windows machine. On Splunk Cloud Platform, you can monitor Windows network information from a universal forwarder that you install on the Windows machine from which you want to collect the information. Then, you can forward that data to Splunk Cloud Platform.

The Splunk platform can collect the following network information:

- **Network activity.** When a Windows machine performs any kind of network action, the Splunk platform can monitor it.
- **Address family.** Whether or not the network transaction was made over the IPv4 or IPv6 protocols.
- **Packet type.** The type of packet sent in the transaction, such as a connect or transport packet.
- **Protocol.** Whether or not the network transaction was made over the TCP or UDP protocols.
- **Hosts.** Information about the hosts involved in the network transaction, including the local and remote hosts, the ports which the hosts used to communicate, and any available DNS information.
- **Application.** Which application initiated the network transaction.
- **User.** The user that initiated the network transaction, including their ID and SID.
- **Miscellany.** Miscellaneous information about the network transaction, including the transport header size and whether or not the transaction was protected by IPsec.

Windows versions of Splunk Enterprise and the universal forwarder support local collection of network information.

The network monitor input runs as a process called `splunk-netmon.exe`. This process runs once for every input you define, at the interval you choose in the input. You can configure network monitoring using either Splunk Web or the `inputs.conf` configuration file.

Windows network monitoring is only available on 64-bit Windows systems. It does not function on 32-bit Windows systems.

Monitoring network information

Windows network monitoring gives you detailed information about your Windows network activity. You can monitor all transactions on the network, such as the initiation of a network connection by a user or process or whether or not the transaction uses the IPv4 or IPv6 address families. The network monitoring facilities in Splunk Enterprise can help you detect and interrupt an incoming or outgoing denial of service attack by telling you the involved machines. With the Splunk search processing language, you can give your team at-a-glance statistics on all Windows network operations.

Requirements

Meet the following requirements before you monitor network information:

- Splunk Enterprise or the universal forwarder must run on Windows. See *Install on Windows* in the Splunk Enterprise *Installation Manual*.
- The Windows version on the machine must be one of the following:

- ◆ Windows 8.1
- ◆ Windows 10
- ◆ Windows Server 2012 R2
- ◆ Windows Server 2016
- ◆ Windows Server 2019
- The Windows system must have all available updates and service packs applied. Network monitoring input might not function if all updates are not present on your Windows machine.
- Splunk Enterprise or the universal forwarder must run as the Local System user or a local administrator account to read all local host information.

Security and remote access considerations

The Splunk platform must run as the Local System user to collect Windows network information by default.

Use a universal forwarder to send host information from remote machines to an indexer when possible. If you choose to install forwarders on your remote machines to collect Windows network information, then install the forwarder as the Local System user on these machines. The Local System user has access to all data on the local machine, but not on remote machines.

If you run the Splunk platform as a user other than the Local System user, then that user must have local Administrator rights to the machine and other explicit permissions, as detailed in [Choose the Windows user Splunk Enterprise should run as](#) in the *Installation* manual.

Use the inputs.conf file to configure Windows network monitoring

To define a Windows network monitoring input, use the `[WinNetMon://<name>]` stanza in the `inputs.conf` file. The Splunk platform uses the following settings to configure the Windows network monitor input.

1. Using a text editor, create the `inputs.conf` file in the `%SPLUNK_HOME%\etc\system\local` directory on the instance where you want to collect Windows network information.
2. Add a `[WinNetMon://<name>]` stanza to the file.
3. Specify one or more of the settings in the table following this task, based on how you want to configure monitoring of the Windows network.
4. Save the file and close it.
5. Restart the Splunk platform.
Monitoring of the Windows network begins immediately.

The following table shows the settings you can configure to monitor a Windows network:

Setting	Description	Default
<code>disabled = [0 1]</code>	Whether or not the input runs. Set to 1 to disable the input and 0 to enable it.	0 (enabled)
<code>index = <string></code>	The index that this input sends the data to. This attribute is optional.	The default index
<code>remoteAddress = <regular expression></code>	Matches against the remote IP address involved in the network transaction. Accepts regular expressions that represent IP addresses only, not host names. Filters out events with remote addresses that do not match the regular expression. Passes through events with remote	Empty string (matches everything)

Setting	Description	Default
	addresses that match the regular expression. For example: 192\.\163\..* matches all IP addresses in the 192.163.x.x range.	
process = <regular expression>	Matches against the process or application name which performed the network access. Filters out events generated by processes that do not match the regular expression. Passes through events generated by processes that match the regular expression.	Empty string (matches all processes or applications)
user = <regular expression>	Matches against the user name which performed the network access. Filters out events generated by users that do not match the regular expression. Passes through events generated by users that match the regular expression.	Empty string (includes access by all users)
addressFamily = [ipv4;ipv6]	If set, matches against the address family used in the network access. Accepts semicolon-separated values, for example <code>ipv4;ipv6</code> .	Empty string (includes all IP traffic)
packetType = [connect;accept;transport]	Matches against the packet type used in the transaction. Accepts semicolon-separated values, for example <code>connect;transport</code> .	Empty string (includes all packet types)
direction = [inbound;outbound]	If set, matches against the general direction of the network traffic. Inbound means traffic coming into the monitoring machine, and outbound means traffic leaving the monitoring machine. Accepts semicolon-separated values, for example <code>inbound;outbound</code> .	Empty string (includes both directions)
protocol = [tcp;udp]	Matches against the specified network protocol. <code>tcp</code> means Transmission Control Protocol, where networks use handshakes to and state to set up transactions. <code>udp</code> means User Datagram Protocol, a stateless, fire and forget network protocol. Accepts semicolon-separated values, for example <code>tcp;udp</code> .	Empty string (includes both protocol types.)
readInterval = <integer>	Advanced option. Use the default value unless there is a problem with input performance. How often, in milliseconds, to read the network monitor filter driver. Allows for the adjustment of call frequency into the kernel driver. Higher frequencies might affect network performance, while lower frequencies can cause event loss. The minimum legal value is 10 and the maximum legal value is 1000.	100
driverBufferSize = <integer>	Advanced option. Use the default value unless there is a problem with input performance.	1024

Setting	Description	Default
	The number of network packets it keeps in the network monitor filter driver buffer. Controls the amount of packets that the driver caches. Lower values might result in event loss, while higher values might increase the size of non-paged memory. The minimum legal value is 128 and the maximum legal value is 8192.	
mode = <string>	How to output each event. The Splunk platform can output each event in either <code>single</code> or <code>multikv</code> (key-value pair) mode.	<code>single</code>
multikvMaxEventCount = <integer>	<p>Advanced option. Use the default value unless there is a problem with input performance.</p> <p>The maximum amount of events to output when you set <code>mode</code> to <code>multikv</code>. The minimum legal value is 10 and the maximum legal value is 500.</p>	100
multikvMaxTimeMs = <integer>	<p>Advanced option. Use the default value unless there is a problem with input performance.</p> <p>The maximum amount of time, in milliseconds, to output <code>multikv</code> events when you set <code>mode</code> to <code>multikv</code>. The minimum legal value is 100 and the maximum legal value is 5000.</p>	1000

Use Splunk Web to configure host monitoring

You can only use Splunk Web to monitor Windows network information on a Splunk platform instance. In any other scenario, you must configure host monitoring with a configuration file.

Follow these high-level steps to configure host monitoring with Splunk Web:

1. Go to the Add Data page.
2. Select the input source.
3. Specify input settings.
4. Review your choices.

Go to the Add Data page

Choose one of the following options to get to the Add Data page.

To add data from settings, follow these steps:

1. Click **Settings** in the upper right corner of Splunk Web.
2. Click **Data Inputs**.
3. Click **Local Windows network monitoring**.
4. Click **New** to add an input.

To add data from the Splunk Web homepage, follow these steps:

1. Click **Add Data**.
2. Click **Monitor** to monitor network information from the local Windows machine or **Forward** to forward network information from another Windows machine.

Forwarding network information requires additional setup.

Splunk Web displays the Add Data - Select Source page.

3. In the left pane, locate and select **Local Windows network monitoring**.

Select the input source

1. In the **Network Monitor Name** field, enter a unique and memorable name for this input.
2. Under **Address family**, check the IP address family types that you want the Splunk platform to monitor. The types are either IPv4 or IPv6.
3. Under **Packet Type**, check the packet types you want the input to monitor. You can choose **connect**, **accept**, or **transport**.
4. Under **Direction**, check the network directions that you want the input to monitor. Choose **inbound** to monitor toward the monitoring host or **outbound** to monitor away from the host.
5. Under **Protocol**, check the network protocol types that you want the input to monitor. Choose **tcp** (Transmission Control Protocol) or **udp** (User Datagram Protocol).
6. In the **Remote address** text field, enter the host name or IP address of a remote host whose network communications with the monitoring host that you want the input to monitor. If you want to monitor multiple hosts, enter a regular expression in this field.
7. In the **Process** text field, enter the partial or full name of a process whose network communications you want the input to monitor. You can monitor multiple processes by entering a regular expression.
8. In the **User** text field, enter the partial or full name of a user whose network communications you want the input to monitor. You can monitor multiple users by entering a regular expression.
9. Click **Next**.

Specify input settings

You can specify application context, default host value, and index on the Input Settings page. All of these parameters are optional.

1. Select the appropriate **Application context** for this input.
2. Set the **Host** name. You have several choices for this setting. Learn more about setting the host value in [About hosts](#).

Host sets only the host field in the resulting events. It does not direct the Splunk platform to look on a specific host on your network.

3. Set the **Index** that the Splunk platform will send data to. Leave the value as **default** unless you defined multiple indexes to handle different types of events. In addition to indexes for user data, the Splunk platform has a number of utility indexes, which also appear in this drop-down list.
4. Click **Review**.

Review your choices

After specifying all your input settings, review your selections. The Splunk platform lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.

2. If they do not match what you want, click the left-pointing angle bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

The success page loads, and the Splunk platform begins indexing the specified print information.

Fields for Windows network monitoring data

When the Splunk platform indexes data from Windows network monitoring inputs, it sets the **source** for received events to `windows`. It sets the **source type** of the incoming events to `WinNetMon`.

Get data with HTTP Event Collector

Share HEC Data

When you use HEC to collect data, the Splunk platform sends de-identified usage data ingested through HEC from Splunk add-ons, apps, and connectors to Splunk. This data is used to target pain points and improve functionality in future releases. For information about how this data is collected, stored, and governed, see [Share data in Splunk Enterprise](#).

Types of data collected

`deployment.httpEventCollector` determines data usage by aggregating the following information:

Data collected	Field name
The name of the add-on, app, or connector	app
Number of bytes ingested	bytes
The version of the add-on, app, or connector	version

For example:

```
data: { [-]  
app: stream333  
bytes: 50  
version: 3.1
```

Telemetry data for HEC is collected by default, and you can opt out of data sharing at any time. See [How to opt out](#).

Components

HEC telemetry collects the following information:

Component	Description	Example
<code>deployment.httpEventCollector</code>	Tracks the amount of data that is processed through HEC for an add-on, app, or connector.	<pre>{ [-] app: component: deployment.httpEventCollector data: { [-] app: stream333 bytes: 50 version: 3.1 } deploymentID: 18393d55-3552-546c-a5ab-61a96a04ae04 eventID: 367E743C-D629-4B25-B46A-78447116F3A4 executionID: 319FB159-0B47-4CA0-B29D-4CD0EDDF0DCF optInRequired: 1 timestamp: 1586974636 type: event userID: 574f5debd4e54c49ef018a6e1bde0379df499a23a865ab83e8d23d1170256f40 visibility: [[-] anonymous support</pre>

Component	Description	Example
		<pre>] } </pre>

Set up and use HTTP Event Collector in Splunk Web

The HTTP Event Collector (HEC) lets you send data and application events to a Splunk deployment over the HTTP and Secure HTTP (HTTPS) protocols. HEC uses a token-based authentication model. You can generate a token and then configure a logging library or HTTP client with the token to send data to HEC in a specific format. This process eliminates the need for a Splunk forwarder when you send application events.

After you enable HEC, you can use HEC tokens in your app to send data to HEC. You do not need to include Splunk credentials in your app or supported files to access the Splunk platform instance.

HEC functionality varies based on Splunk software type

HTTP Event Collector runs on Splunk Cloud Platform and Splunk Enterprise. How it works depends on the type of Splunk platform instance you have.

HEC and Splunk Cloud Platform

You can enable HEC on a Splunk Cloud Platform deployment. The following caveats apply to using HEC on a Splunk Cloud Platform instance:

- If you need to use a configuration file to configure an HEC input, you must do this on a heavy forwarder, then forward the data to Splunk Cloud Platform. This is because Splunk Cloud Platform does not provide access to configuration files locally.
- You must file a ticket with Splunk Support to enable HEC for use with Amazon Web Services (AWS) Kinesis Firehose. Standard HEC is enabled by default on all Splunk Cloud Platform deployments and does not require a Splunk Support ticket.
- You cannot make changes to global settings. You can only make settings changes to tokens that you create.
- You cannot forward data that HEC receives to another set of Splunk indexers as Splunk Cloud Platform does not support forwarding output groups.
- The index that you choose to store events that HEC receives must already exist. You cannot create a new index during the setup process.
- Indexer acknowledgment is only available for AWS Kinesis Firehose at this time.
- After you create tokens, you can monitor progress of the token as it is deployed across your Splunk Cloud Platform instance.

For instructions on how to enable and manage HEC on Splunk Cloud Platform, see [Configure HTTP Event Collector on Splunk Cloud](#).

HEC and Splunk Enterprise

HEC offers full configurability and functionality on the Splunk Enterprise platform on-premises. It offers the following additional benefits over HEC on Splunk Cloud Platform:

- HEC can accept events that you send to it over the HTTP protocol in addition to the HTTPS protocol.
- HEC can forward events to another Splunk indexer with an optional forwarding **output group**.
- You can use the deployment server to distribute HEC tokens across indexers in a distributed deployment.

For instructions on how to enable and manage HEC on Splunk Enterprise, see [Configure HTTP Event Collector on Splunk Enterprise](#).

How the Splunk platform uses HTTP Event Collector tokens to get data in

Tokens are entities that let logging agents and HTTP clients connect to the HEC input. Each token has a unique value, which is a 128-bit number that is represented as a 32-character globally unique identifier (GUID). Agents and clients use a token to authenticate their connections to HEC. When the clients connect, they present this token value. If HEC receives a valid token, it accepts the connection and the client can deliver its payload of application events in either text or JavaScript Object Notation (JSON) format.

HEC receives the events and Splunk Enterprise indexes them based on the configuration of the token. HEC uses the source, source type, and index that was specified in the token. If a forwarding output group configuration exists on a Splunk Enterprise instance, HEC forwards the data to indexers in that output group.

Configure HTTP Event Collector on Splunk Cloud Platform

HEC is enabled by default in Splunk Cloud Platform. You can create, modify, delete, enable, and disable HEC tokens.

Enable HTTP Event Collector on Splunk Cloud Platform

HTTP Event Collector is enabled by default on Splunk Cloud Platform.

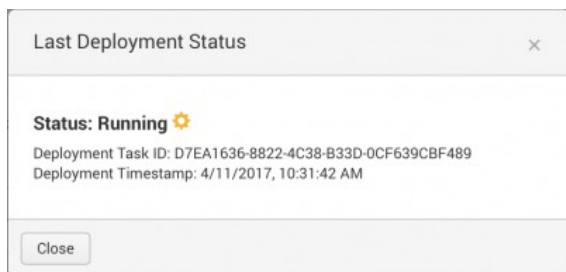
Create an Event Collector token on Splunk Cloud Platform

To use HEC, you must configure at least one token. Splunk Cloud Platform distributes the token across the deployment. The token is not ready for use until distribution has completed.

1. Click **Settings > Add Data**.
2. Click **monitor**.
3. Click **HTTP Event Collector**.
4. In the **Name** field, enter a name for the token.
5. (Optional) In the **Source name override** field, enter a name for a source to be assigned to events that this endpoint generates.
6. (Optional) In the **Description** field, enter a description for the input.
7. (Optional) If you want to enable indexer acknowledgment for this token, click the **Enable indexer acknowledgment** checkbox.
8. Click **Next**.
9. (Optional) Make edits to source type and confirm the index where you want HEC events to be stored. See [Modify input settings](#).
10. Click **Review**.
11. Confirm that all settings for the endpoint are what you want.
12. If all settings are what you want, click **Submit**. Otherwise, click < to make changes.
13. (Optional) Copy the token value that Splunk Web displays and paste it into another document for reference later.
14. (Optional) Click **Track deployment progress** to see progress on how the token has been deployed to the rest of the Splunk Cloud Platform deployment. When you see a status of "Done", you can then use the token to send data to HEC.

Check Event Collector token distribution status on Splunk Cloud Platform

You can check the distribution status of an HEC token from the HEC token page. When a distribution is in progress, the page displays "Operation in progress" and a progress bar. Otherwise, the page displays "Last deployment status."



1. Click **Settings > Data Inputs**.
2. Click **HTTP Event Collector**.
3. Click **Operation in progress** or **Last deployment status**.
4. View the status of the token distribution.
5. Click **Close**.

Modify an Event Collector token on Splunk Cloud Platform

You can make changes to an HEC token after you create it.

1. Click **Settings > Data Inputs**.
2. Click **HTTP Event Collector**.
3. Locate the token that you want to change in the list.
4. In the **Actions** column for that token, click **Edit**. You can also click the link to the token name.
5. (Optional) Edit the description of the token by entering updated text in the **Description** field.
6. (Optional) Update the source value of the token by entering text in the **Source** field.
7. (Optional) Choose a different source type by selecting it in the **Source Type** drop-down list box.
 1. Choose a category.
 2. Select a source type in the pop-up menu that appears.
 3. (Optional) You can also type in the name of the source type in the text box at the top of the drop-down list box.
8. (Optional) Choose a different index by selecting it in the **Available Indexes** pane of the **Select Allowed Indexes** control.
9. (Optional) Choose whether you want indexer acknowledgment enabled for the token.
10. Click **Save**.

Delete an Event Collector token on Splunk Cloud Platform

You can delete an HEC token. Deleting an HEC token does not affect other HEC tokens, nor does it disable the HEC endpoint.

You cannot undo this action. Clients that use this token to send data to your Splunk deployment can no longer authenticate with the token. You must generate a new token and change the client configuration to use the new value.

1. Click **Settings > Data Inputs**.
2. Click **HTTP Event Collector**.

3. Locate the token that you want to delete in the list.
4. In the **Actions** column for that token, click **Delete**.
5. In the Delete Token dialog, click **Delete**.

Enable and disable Event Collector tokens in Splunk Cloud Platform

You can enable or disable a token from within the HEC management page. Changing the active status of one token does not change the status of other tokens.

1. Click **Settings > Data Inputs**.
2. Click **HTTP Event Collector**.
3. In the **Actions** column for a token, click the **Enable** link, if the token is not active, or the **Disable** link, if the token is active. The token status toggles and the link changes to **Enable** or **Disable** based on the changed token status.

Configure HTTP Event Collector on Splunk Enterprise

You can enable HEC and create, modify, delete, enable, and disable HEC tokens in Splunk Enterprise.

Enable HTTP Event Collector on Splunk Enterprise

Before you can use Event Collector to receive events through HTTP, you must enable it. For Splunk Enterprise, enable HEC through the **Global Settings** dialog box.

1. Click **Settings > Data Inputs**.
2. Click **HTTP Event Collector**.
3. Click **Global Settings**.
4. In the **All Tokens** toggle button, select **Enabled**.
5. (Optional) Choose a **Default Source Type** for all HEC tokens. You can also type in the name of the source type in the text field above the drop-down list box before choosing the source type.
6. (Optional) Choose a **Default Index** for all HEC tokens.
7. (Optional) Choose a **Default Output Group** for all HEC tokens.
8. (Optional) To use a deployment server to handle configurations for HEC tokens, click the **Use Deployment Server** check box.
9. (Optional) To have HEC listen and communicate over HTTPS rather than HTTP, click the **Enable SSL** checkbox.
10. (Optional) Enter a number in the **HTTP Port Number** field for HEC to listen on.

Confirm that no firewall blocks the port number that you specified in the "HTTP Port Number" field, either on the clients or the Splunk instance that hosts HEC.

11. Click **Save**.

Create an Event Collector token on Splunk Enterprise

To use HEC, you must configure at least one token.

1. Click **Settings > Add Data**.
2. Click **monitor**.
3. Click **HTTP Event Collector**.
4. In the **Name** field, enter a name for the token.
5. (Optional) In the **Source name override** field, enter a source name for events that this input generates.
6. (Optional) In the **Description** field, enter a description for the input.
7. (Optional) In the **Output Group** field, select an existing forwarder output group.

8. (Optional) If you want to enable indexer acknowledgment for this token, click the **Enable indexer acknowledgment** checkbox.
9. Click **Next**.
10. (Optional) Confirm the source type and the index for HEC events.
11. Click **Review**.
12. Confirm that all settings for the endpoint are what you want.
13. If all settings are what you want, click **Submit**. Otherwise, click < to make changes.
14. (Optional) Copy the token value that Splunk Web displays and paste it into another document for reference later.

Modify an Event Collector token on Splunk Enterprise

You can make changes to an HEC token after you have created it.

1. Click **Settings > Data Inputs**.
2. Click **HTTP Event Collector**.
3. Locate the token that you want to change in the list.
4. In the **Actions** column for that token, click **Edit**. You can also click the link to the token name.
5. (Optional) Edit the description of the token by entering updated text in the **Description** field.
6. (Optional) Update the source value of the token by entering text in the **Source** field.
7. (Optional) Choose a different source type by selecting it in the **Source Type** drop-down list box.
 1. Choose a category.
 2. Select a source type in the pop-up menu that appears.
 3. (Optional) You can also type in the name of the source type in the text box at the top of the drop-down.
8. (Optional) Choose a different index by selecting it in the **Available Indexes** pane of the **Select Allowed Indexes** control.
9. (Optional) Choose a different output group from the **Output Group** drop-down list box.
10. (Optional) Choose whether you want indexer acknowledgment enabled for the token.
11. Click **Save**.

Delete an Event Collector token on Splunk Enterprise

You can delete an HEC token. Deleting an HEC token does not affect other HEC tokens, nor does it disable HEC.

You cannot undo this action. Clients that use this token to send data to your Splunk deployment can no longer authenticate with the token. You must generate a new token and change the client configuration to use the new token.

1. Click **Settings > Data Inputs**.
2. Click **HTTP Event Collector**.
3. Locate the token that you want to delete in the list.
4. In the **Actions** column for that token, click **Delete**.
5. In the Delete Token dialog box, click **Delete**.

Enable and disable Event Collector tokens on Splunk Enterprise

You can enable or disable a single HEC token from within the HEC management page. Changing the status of one token does not change the status of other tokens. To enable or disable all tokens, use the Global Settings dialog. See [Enable the HTTP Event Collector](#).

To toggle the active status of an HEC token:

1. Click **Settings > Data Inputs**.

2. Click **HTTP Event Collector**.

3. In the **Actions** column for that token, click the **Enable** link, if the token is not active, or the **Disable** link, if the token is active. The token status toggles immediately and the link changes to **Enable** or **Disable** based on the changed token status.

Use output groups to specify groups of indexers on Splunk Enterprise

To index large amounts of data, you will likely need multiple indexers. On Splunk Enterprise only, you can specify groups of indexers to handle indexing your HTTP Event Collector data. These groups are called *output groups*. You can use output groups to, for example, index only certain kinds of data, or data from certain sources.

You configure output groups in the `outputs.conf` configuration file. Specifically, for HTTP Event Collector, edit the `outputs.conf` file at `$SPLUNK_HOME/etc/apps/splunk_httpinput/local/` (`%SPLUNK_HOME%\etc\apps\splunk_httpinput\local\` on Microsoft Windows hosts). If either the `local` directory or the `outputs.conf` file doesn't exist at this location, create it (or both).

HTTP Event Collector is not an app, but it stores its configuration in the `$SPLUNK_HOME/etc/apps/splunk_httpinput/` directory (`%SPLUNK_HOME%\etc\apps\splunk_httpinput\` on Windows) so that its configuration can be easily deployed using built-in app deployment capabilities.

Send data to HTTP Event Collector

You must satisfy all of the following conditions when you send data to HEC:

- HEC must be enabled
- You must have at least one active HEC token available
- You must use an active token to authenticate into HEC
- You must format the data that goes to HEC in a certain way. See [Format events for HTTP Event Collector](#)

There are several options for sending data to HTTP Event Collector:

- You can make an HTTP request using your favorite HTTP client and send your JSON-encoded events.
- As a developer, you can use the Java, JavaScript (`node.js`), and .NET logging libraries in your application to send data to HEC. These libraries are compatible with popular logging frameworks. See [Java](#), [JavaScript \(Node.js\)](#), and [.NET](#) on the Splunk Dev Portal.

Send data to HTTP Event Collector on Splunk Cloud Platform

You must send data using a specific URI for HEC.

The standard form for the HEC URI in Splunk Cloud Platform free trials is as follows:

```
<protocol>://http-inputs-<host>.splunkcloud.com:<port>/<endpoint>
```

The standard form for the HEC URI in Splunk Cloud Platform is as follows:

```
<protocol>://http-inputs-<host>.splunkcloud.com:<port>/<endpoint>
```

The standard form for the HEC URI in Splunk Cloud Platform on Google Cloud is as follows:

`<protocol>://http-inputs.<host>.splunkcloud.com:<port>/<endpoint>`

Where:

- `<protocol>` is either `http` or `https`
- You must add `http-inputs-` before the `<host>` on AWS.
- You must add `http-inputs.` before the `<host>` on GCP.
- `<host>` is the Splunk Cloud Platform instance that runs HEC
- You must add the domain `.splunkcloud.com` after the `<host>`
- `<port>` is the HEC port number
 - ◆ 8088 on Splunk Cloud Platform free trials
 - ◆ 443 by default on Splunk Cloud Platform instances
- `<endpoint>` is the HEC endpoint you want to use. In many cases, you use the `/services/collector/event` endpoint for JavaScript Object Notation (JSON)-formatted events or the `services/collector/raw` endpoint for raw events

If you do not include these prefixes before your Splunk Cloud Platform hostname when you send data, the data cannot reach HEC.

Send data to HTTP Event Collector on Splunk Enterprise

You send data to a specific Uniform Resource Indicator (URI) for HEC.

The standard form for the HEC URI in Splunk Enterprise is as follows:

`<protocol>://<host>:<port>/<endpoint>`

Where:

- `<protocol>` is either `http` or `https`
- `<host>` is the Splunk instance that runs HEC
- `<port>` is the HEC port number, which is 8088 by default, but you can change in the HEC Global Settings
- `<endpoint>` is the HEC endpoint you want to use. In many cases, you use the `/services/collector/event` endpoint for JavaScript Object Notation (JSON)-formatted events or the `services/collector/raw` endpoint for raw events

Example of sending data to HEC with an HTTP request

The following example makes a HTTP POST request to the HEC on port 8088 and uses HTTPS for transport. This example uses the `curl` command to generate the request, but you can use a command line or other tool that better suits your needs.

You can configure the network port and HTTP protocol settings independently of settings for other instances of HEC in your Splunk Enterprise or Splunk Cloud Platform deployment.

The following cURL command uses an example HTTP Event Collector token (B5A79AAD-D822-46CC-80D1-819F80D7BFB0), and uses `https://hec.example.com` as the hostname. Replace these values with your own before running this command.

JSON request and response

When you make a JSON request to send data to HEC, you must specify the "event" key in the command.

The Authorization HTTP header for HEC requires the "Splunk" keyword before the HEC token.

```
curl https://hec.example.com:8088/services/collector/event -H "Authorization: Splunk
B5A79AAD-D822-46CC-80D1-819F80D7BFB0" -d '{"event": "hello world"}'
{"text": "Success", "code": 0}
```

More information on HEC

- For information about defining forwarding output groups, see [Configure forwarders with outputs.conf](#). You can also set up forwarding in Splunk Web, which generates a default output group called `default-autolb-group`.
- For information on indexer acknowledgement, see [HTTP Event Collector indexer acknowledgment](#). Indexer acknowledgment in HTTP Event Collector is not the same indexer acknowledgment capability described in indexer acknowledgment and indexer clusters.

More information on HEC for developers

- For developer content on using HTTP Event Collector, see [HTTP Event Collector Examples](#), as well as [Introduction to Splunk HTTP Event Collector in the Splunk Developer Portal](#).

Set up and use HTTP Event Collector with configuration files

HTTP Event Collector (HEC) stores its settings on a Splunk Enterprise instance in two configuration files: `inputs.conf` and `outputs.conf`. These files are not accessible on Splunk Cloud Platform instances, and you must manage configurations on Splunk Cloud Platform instances through Splunk Web.

Configuring HEC inputs with a configuration file is a slightly different process than configuring other data inputs. In many cases, you edit the files in the `$SPLUNK_HOME/etc/system/local` directory. For HEC, you edit the files in the `$SPLUNK_HOME/etc/apps/splunk_httpinput/local/` directory.

No matter how many `inputs.conf` files a Splunk Enterprise instance has and where they reside, Splunk Enterprise combines all their settings, using the rules of location precedence. See [Configuration file precedence](#) in the *Admin Manual*.

To set up HEC with configuration files, follow these steps:

1. In the `$SPLUNK_HOME/etc/apps/splunk_httpinput` directory, create a `local` directory, if it does not exist.
2. Change to the `$SPLUNK_HOME/etc/apps/splunk_httpinput/local` directory.
3. Create an `inputs.conf` file if it does not exist.
4. Use a text editor to open `inputs.conf` for editing.
5. Specify global and token settings as described in [Token-related settings](#) later in this topic.

The HEC token must be a Globally Unique Identifier (GUID).

6. Save the file and close it.
7. Restart Splunk Enterprise for the changes to take effect.

Token-related settings

HEC stores settings related to token management in the `inputs.conf` configuration file.

You can specify whether settings apply globally to all tokens or only to specific tokens:

- The `[http]` stanza contains global settings that apply to all tokens.
- The `[token_name]` stanzas, where `token_name` indicates the token name as assigned by the user, apply to individual tokens. Settings specified here override settings specified within the `[http]` stanza.

The `inputs.conf` file contains basic explanatory information about each setting.

Global settings

The `[http]` stanza contains global settings that apply to all tokens. To set the HTTP queue size globally, update `server.conf` using the following **case-sensitive** setting:

```
[queue=httpInputQ]
maxSize=10MB
```

Parameter	Description
<code>dedicatedIoThreads</code>	The number of dispatcher threads on the HTTP Event Collector server. The default value is 1. Do not alter this setting unless you are requested to do so by Splunk Support. The value of this parameter must never be more than the number of physical CPU cores on your Splunk Enterprise server.
<code>disabled</code>	Whether tokens are disabled. 1 indicates true, and 0 indicates false. The default value is 1. When set to 1 in the <code>[http]</code> stanza, this parameter disables all tokens.
<code>enableSSL</code>	Whether the HTTP Event Collector server protocol is HTTP or HTTPS. 1 indicates HTTPS is enabled; 0 indicates HTTP. The default value is 1. HTTP Event Collector shares SSL settings with the Splunk Enterprise instance and can't have <code>enableSSL</code> settings that differ from the settings on the Splunk Enterprise instance.
<code>index</code>	The global default index. This parameter can be overridden when set in an individual token stanza, or when the header of event data contains an <code>index</code> parameter set to a different value. You can limit the set of allowed values for this parameter on a per-token basis by using the <code>indexes</code> parameter.
<code>maxSockets</code>	The number of HTTP Event Collector connections, expressed as an integer, that Splunk Enterprise accepts simultaneously. You can limit this number to constrain resource usage. When set to 0, Splunk Enterprise automatically sets it to one-third of the maximum allowable open files on the host. If this number is less than 50, it is set to 50. If this number is greater than 400000, it is set to 400000. If set to a negative number, no limit is enforced. Defaults to 0.
<code>maxThreads</code>	The number of threads, expressed as an integer, that can be used by active HTTP transactions. You can limit this number to constrain resource usage. When set to 0, Splunk Enterprise automatically sets the limit to one-third of the maximum allowable threads on the host. If this number is less than 20, it is set to 20. If this number is greater than 150000, it is set to 150000. If <code>maxSockets</code> is not negative and <code>maxThreads</code> is greater than <code>maxSockets</code> , then Splunk Enterprise sets <code>maxThreads</code> to be equal to <code>maxSockets</code> . If set to a negative number, no limit is enforced. Defaults to 0.
<code>outputgroup</code>	The global default output group. An output group is a group of indexers set up by the Splunk Enterprise administrator to index the data. If there is no output group specified, event data goes to the local indexer. If the given output group is invalid, the data is dropped, and an error message is logged to <code>splunkd.log</code> . For more information about specifying output groups, see Output group-related settings later in this topic.
<code>port</code>	The HTTP Event Collector server port. The default value is 8088. This port number must not already be in use.
<code>sourcetype</code>	The global default source type. This parameter can be overridden either when set in an individual token's stanza or by event data whose header contains a <code>sourcetype</code> parameter set to a different value.
<code>useDeploymentServer</code>	Whether to use the Deployment Server. When set to 1 (true), writes to the location specified by <code>repositoryLocation</code> property in the <code>serverclass.conf</code> file. Defaults to 0 and writes to <code>\$SPLUNK_HOME/etc/apps</code> .

Parameter	Description
-----------	-------------

Per-token settings

The `http://<token_name>` stanzas, where `<token_name>` indicates the token name as assigned by the user, apply to individual tokens. Settings specified here override settings specified within the `[http]` stanza.

Parameter	Description
<code>connection_host</code>	The type of default host for the token. This parameter can be set to any of the following literal values: <ul style="list-style-type: none"> • <code>dns</code> indicates the host value is the reverse DNS entry for the IP address of the system sending the data. • <code>ip</code> indicates the host value is the IP address of the system sending the data. • <code>none</code> sets the host value to the connection host specified in the HTTP host header. This is typically the Splunk platform instance host name.
<code>disabled</code>	Whether the token is disabled. 1 indicates true; 0 indicates false. The default value is 0.
<code>index</code>	The token's default index. This parameter can be overridden by event data whose header contains an <code>index</code> parameter set to a different value.
<code>indexes</code>	A list of allowable indexes to which the data can be indexed. See global settings for including lists.
<code>persistentQueueSize</code>	The maximum size of the persistent queue . The value of this parameter is in the form <code><integer>[KB MB GB]</code> . The default value is 0, which indicates there is no persistent queue. Persistent queues can help prevent loss of transient data by saving data in an input queue to disk. When set, the value of the <code>persistentQueueSize</code> parameter must be more than the value of the <code>queueSize</code> parameter. For more information about persistent queues, see Use persistent queues to help prevent data loss .
<code>source</code>	The token's default source. This parameter can be overridden by event data whose header contains a <code>source</code> parameter set to a different value.
<code>queueSize</code>	The maximum size of the input queue in memory. The value of this parameter is in the form <code><integer>[KB MB GB]</code> . The default value is 500KB.
<code>sourcetype</code>	The token's default source type. This parameter can be overridden by event data whose header contains a <code>sourcetype</code> parameter set to a different value.
<code>token</code>	The HTTP Event Collector token. The token must be a unique GUID.

Output group-related settings

Settings that apply to forwarding and load balancing are stored in `outputs.conf`, including settings for specifying HTTP Event Collector output groups. These settings are the same ones that Splunk Enterprise admins use to manage forwarding and load balancing among indexers.

Specify global settings in the `[tcpout]` stanza, and specify per-output group settings in the `[tcpout:<target_group>]` stanza.

The `outputs.conf` file contains basic explanatory information about each setting. For more information, see [About forwarding and receiving](#) in the *Forwarding Data* manual and [Configure forwarders with outputs.conf](#) in the Splunk Universal Forwarder *Forwarder Manual*.

Global settings

The `[tcpout]` stanza defines the output groups to which the data is forwarded.

Parameter	Description
-----------	-------------

defaultGroup	A comma-separated list of one or more target output group names in the form <target_group>, <target_group>, The names of the output groups are specified later in the outputs.conf file in the [tcpout:<target_group>] stanzas. Data is sent to the specified groups.
--------------	--

Per-output group settings

The [tcpout:target_group] stanza defines the configuration of the target output group indicated by <target_group>. You can have as many target groups as you want. If more than one target group is specified, the forwarder clones the data to each target group.

Parameter	Description
blockWarnThreshold	The output pipeline's send failure count threshold. The default value is 100. After the threshold is met, a failure message is displayed as a banner in Splunk Web. To effectively disable this warning, set this to a very large value, like 2000000.
server	<server name>:<port>, [<ip> <server name>]:<port>, For each mentioned system, you must include the port number, and the IP address or server name.

Set up and use HTTP Event Collector from the CLI

You can use the `http-event-collector` parameter of the Splunk command line interface (CLI) and its options to administer an HTTP Event Collector (HEC) instance on a Splunk Enterprise server.

It's not possible to use HEC on a Splunk Cloud Platform instance from the CLI. If you have a Splunk Cloud Platform instance, log into the instance and manage HEC from Splunk Web instead.

For more information about the CLI, see the following topics in the Splunk Enterprise *Admin Manual*:

- About the CLI
- Use the CLI to administer a remote Splunk server

CLI syntax

There are two syntaxes to use when you administer HEC through the CLI:

- The syntax for all other HEC actions, such as creating, deleting, and showing tokens
- The syntax for sending data to HEC

Use the following syntax for all actions except sending data to HEC:

```
splunk http-event-collector <command> <token-name> [<option2>] [<-parameter1> <value1>] [<-parameter2> <value2>] <data>
```

All HTTP Event Collector commands except for `send` assume that the first option following the command name is the name of the token. In addition, the `create` command assumes that the second option is a description of the token in quotation marks.

Use the following syntax to send data to the HEC:

```
splunk http-event-collector send -uri <uri_value> -name <token-name> <data>
```

If you want to apply the CLI commands to the global configuration, don't include the `-name <token-name>` argument. For example, the following syntax enables HTTP Event Collector:

```
splunk http-event-collector enable -uri <uri_value> <data>
```

Supported CLI commands

The following HTTP Event Collector-specific CLI commands are supported in Splunk Enterprise:

Command	Description
create	Create a new token.
delete	Remove a token.
list	Show all available tokens.
update	Change token properties.
enable	Enable a token.
disable	Disable a token.
help	Show help.
send	Send data to an endpoint.

Supported CLI parameters

HEC supports the following CLI parameters. You must immediately follow a CLI parameter with its value. Enclose any values that contain spaces in quotation marks.

Parameter	Description
-uri	The Uniform Resource Identifier (URI) of the Splunk server that takes the form scheme://host:port. As an alternative to setting this parameter, you can set the <code>\$SPLUNK_URI</code> environment variable instead. The port number to use must be the management port of your Splunk server (8089 by default), and not the HTTP Event Collector port (8088 by default).
-auth	Splunk server user authentication in the form username:password. If this parameter is missing, you are prompted for a username and password.
-name	The name of the token.
-disabled	Whether to disable the token. 1 indicates true and 0 indicates false. You must also update this setting in the global stanza for proper functionality.
-description	A description of the token.
-indexes	A list of indexes accepted by the token.
-index	The token default index. Splunk Enterprise assigns this value to data that doesn't already have an index value set.
-source	The token default source value. Splunk Enterprise assigns this value to data that doesn't already have a source value set.
-sourcetype	The token default sourcetype value. Splunk Enterprise assigns this value to data that doesn't already have a sourcetype value set.
-outputgroup	The token default output group value. An output group is a group of indexers set up by the Splunk software administrator to index the data. Splunk Enterprise assigns this value to data that doesn't already have an output group value set.
-port	The HTTP Event Collector server port. The default value is 8088, but you can change it using this parameter.
-enable-ssl	Whether the HTTP Event Collector server protocol is HTTP or HTTPS. 1 indicates HTTPS and 0 indicates HTTP.
-dedicated-io-threads	The number of dispatcher threads on the HTTP Event Collector server. The default value is 2. Do not alter this setting unless you have been requested to do so by Splunk Support. The value of this parameter can't be more than the number

Parameter	Description
	of physical CPU cores on your Splunk Enterprise server.
-output-format	The output format. <code>txt</code> indicates text and <code>json</code> indicates JSON. The default value is <code>txt</code> .

Example CLI syntax

The following example CLI entry creates a token called `new-token`, assigns it the given URI, gives it a description, sets it to `disabled`, and indicates that the HTTP Event Collector data is to be saved to the `log` index.

```
splunk http-event-collector create new-token -uri https://localhost:8089 -description "this is a new token"
-disabled 1 -index log
```

The following example CLI entry enables a token called `myapp`, assigns it the given URI, and sets the user authentication:

```
splunk http-event-collector enable -name myapp -uri https://localhost:8089 -auth admin:changeme
```

The following example CLI entry sends data to HTTP Event Collector using the given token and URI.

```
splunk http-event-collector send -uri https://localhost:8089 -token new-token {"this is the data to send"}
```

Use cURL to manage HTTP Event Collector tokens, events, and services

You can use the cURL web data transfer application to manage tokens, events, and services for HTTP Event Collector (HEC) on your Splunk Enterprise instance using the Representational State Transfer (REST) API. Using the REST API lets you seamlessly manage HEC objects without having to use Splunk Web or the CLI.

cURL commands differ slightly based on your operating system. For information on how to use cURL commands in Windows, see this [Splunk Community](#) page.

By default, if your token doesn't specify a default index, the data will be under the `main` index. You can specify a default index in your `hec` token configuration.

Manage HTTP Event Collector tokens with cURL

All HEC token operations are available REST using cURL. Splunk Enterprise stores the tokens at the following REST API endpoint, assuming your Splunk Enterprise server management address is as follows:

```
https://localhost:8089/servicesNS/admin/splunk_httpinput/data/inputs/http/
```

List the existing HTTP Event Collector tokens using cURL

You can list the existing tokens in your Splunk Enterprise HEC using cURL. For example, the following example cURL command lists the tokens that exist on the Splunk Enterprise instance at `https://localhost:8089` using the user `admin`:

```
curl -u admin:changeme https://localhost:8089/servicesNS/admin/splunk_httpinput/data/inputs/http
```

Create an HTTP Event Collector token using cURL

To create a token using cURL, use the `name` property. For example, the following example CLI command creates a token called `mytoken` on the Splunk Enterprise instance at `https://localhost:8089` using the user `admin`:

```
curl -u admin:changeme https://localhost:8089/servicesNS/admin/splunk_httpinput/data/inputs/http -d
name=mytoken
```

Edit an HTTP Event Collector token using cURL

You can update any token property except its name or value using cURL. For example, the following example cURL command updates the description of the mytoken token on the Splunk Enterprise instance at <https://localhost:8089> using the user admin:

```
curl -u admin:changeme https://localhost:8089/servicesNS/admin/splunk_httpinput/data/inputs/http/mytoken -d description=abc
```

You can update any of the following parameters:

Parameter	Description
disabled	Whether to disable the token. 1 indicates true; 0 indicates false.
description	A description of the token.
indexes	A list of indexes accepted by the token.
index	The token's default index. Splunk Enterprise assigns this value to data that doesn't already have an index value set.
source	The token's default source value. Splunk Enterprise assigns this value to data that doesn't already have a source value set.
sourcetype	The token's default sourcetype value. Splunk Enterprise assigns this value to data that doesn't already have a sourcetype value set.
outputgroup	The token's default outputgroup value. An output group is a group of indexers set up by the Splunk software administrator to index the data. Splunk Enterprise assigns this value to data that doesn't already have an outputgroup value set.
port	The HTTP Event Collector server port. The default value is 8088, but you can change it using this parameter. For more information on port values for Splunk Cloud Platform instances and free trials, see Send data to HTTP Event Collector on Splunk Cloud Platform . For more information on Splunk Enterprise port values, see Send data to HTTP Event Collector on Splunk Enterprise .
enableSSL	Whether the HTTP Event Collector server protocol is HTTP or HTTPS. 1 indicates HTTPS; 0 indicates HTTP.
dedicatedIoThreads	The number of dispatcher threads on the HTTP Event Collector server. The default value is 2. Do not alter this setting unless you are requested to do so by Splunk Support. The value of this parameter must never be more than the number of physical CPU cores on your Splunk Enterprise instance.
useACK	Returns an acknowledgment when events are indexed. Set to 1 to enable.

Enable or disable an HTTP Event Collector token using cURL

You can enable or disable a token using cURL. Changing the status of one token does not change the status of other tokens. To enable or disable a token, use the HTTP POST command, the token name, and the enable or disable endpoint. For example, the following command disables the token called mytoken on the Splunk Enterprise instance at <https://localhost:8089> using the user admin:

```
curl -X "POST" -u admin:changeme https://localhost:8089/servicesNS/admin/splunk_httpinput/data/inputs/http/mytoken/disable
```

Similarly, the following example enables the token called mytoken on the Splunk Enterprise instance at <https://localhost:8089> using the user admin:

```
curl -X "POST" -u admin:changeme https://localhost:8089/servicesNS/admin/splunk_httpinput/data/inputs/http/mytoken/enable
```

Enable or disable HTTP Event Collector using cURL

You can enable or disable HTTP Event Collector itself by making a bulk change to all tokens using cURL. Do not specify a token name when you use the enable or disable endpoint. To enable or disable HTTP Event Collector, use the HTTP POST command and the enable or disable endpoint. For example, the following example disables HTTP Event Collector on the Splunk Enterprise instance at <https://localhost:8089> using the user admin:

```
curl -X "POST" -u admin:changeme
https://localhost:8089/servicesNS/admin/splunk_httpinput/data/inputs/http/http/disable
```

Delete an HTTP Event Collector token using cURL

To delete a token using cURL, use the HTTP DELETE command and the token name. For example, the following example cURL command deletes the token called mytoken from the Splunk Enterprise instance at <https://localhost:8089> using the user admin:

```
curl -X "DELETE" -u admin:changeme
https://localhost:8089/servicesNS/admin/splunk_httpinput/data/inputs/http/mytoken
```

Manage HEC events and services with cURL

The following commands show how you can send events to and manage HEC services. This list isn't all-inclusive, but it can give you an idea of the things that you can accomplish with HEC.

Send an event to HEC

The following example demonstrates basic HEC usage. It includes the Splunk Enterprise instance address with port and endpoint, the authentication token, and event data and metadata formatted according to the HEC event data format specification.

```
curl "https://http-inputs-mysplunkserver.splunkcloud.com:8088/services/collector" \
-H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
-d '{"event": "Hello, world!", "sourcetype": "manual"}'
```

Send an event to HEC using basic authentication

This example demonstrates basic authentication, which is an alternative to the HTTP Authentication. To use basic authentication, submit a colon-separated username and password pair in the request as the `-u` argument. using any string as the username and the token as the `<password>`: `<user>:<password>`.

```
# Basic auth
curl -u "x:CF179AE4-3C99-45F5-A7CC-3284AA91CF67"
"https://http-inputs-mysplunkserver.splunkcloud.com:8088/services/collector/event" \
-d '{"sourcetype": "mysourcetype", "event": "Hello, world!"}'
```

Send multiple events to HEC in one request

The following example demonstrates sending multiple events in one request. Though you can send multiple events in a single request, you cannot split one event across multiple requests.

```
curl "https://http-inputs-mysplunkserver.splunkcloud.com:8088/services/collector" \
-H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
-d '{"event": "Pony 1 has left the barn"}{"event": "Pony 2 has left the barn"}{"event": "Pony 3 has left the barn", "nested": {"key1": "value1"}{'
```

Send raw text to HEC

The following example demonstrates sending raw text to HEC. Note the use of the raw endpoint, plus the channel identifier and source type specification, both of which are done using URL query parameters.

```
curl
"https://http-inputs-mysplunkserver.splunkcloud.com:8088/services/collector/raw?channel=00872DC6-AC83-4EDE-8AFE-8413C3825C4C&sourcetype=mydata" -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" -d '1, 2, 3... Hello, world!'
```

Send raw batched events to HEC

The following example demonstrates how to send raw batched events to HEC. In this case, the command sends splunkd access logs. It indicates that the indexer must assign these events the source type of `splunkd_access` and specifies that they must be sent to the `main` index.

```
# HEC Raw batching
curl
"https://http-inputs-mysplunkserver.splunkcloud.com:8088/services/collector/raw?channel=00872DC6-AC83-4EDE-8AFE-8413C3825C4C&sourcetype=splunkd_access&index=main" \
  -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
  -d '127.0.0.1 - admin [28/Sep/2016:09:05:26.875 -0700] "GET
/servicesNS/admin/launcher/data/ui/views?count=-1 HTTP/1.0" 200 126721 - - - 6ms
127.0.0.1 - admin [28/Sep/2016:09:05:26.917 -0700] "GET /servicesNS/admin/launcher/data/ui/nav/default
HTTP/1.0" 200 4367 - - - 6ms
127.0.0.1 - admin [28/Sep/2016:09:05:26.941 -0700] "GET
/services/apps/local?search=disabled%3Dfalse&count=-1 HTTP/1.0" 200 31930 - - - 4ms
127.0.0.1 - admin [28/Sep/2016:09:05:26.954 -0700] "GET
/services/apps/local?search=disabled%3Dfalse&count=-1 HTTP/1.0" 200 31930 - - - 3ms
127.0.0.1 - admin [28/Sep/2016:09:05:26.968 -0700] "GET
/servicesNS/admin/launcher/data/ui/views?digest=1&count=-1 HTTP/1.0" 200 58672 - - - 5ms'
```

Send events to HEC with indexer acknowledgement active

The following example demonstrates how to send events to HEC with indexer acknowledgement active. The sole difference between this example and the basic example is the inclusion of a channel identifier. Indexer acknowledgement also works with raw data.

```
# Indexer ack
curl "https://http-inputs-mycompany.splunkcloud.com:8088/services/collector?channel=00872DC6-AC83-4EDE-8AFE-8413C3825C4C" \
  -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
  -d '{"event": "Hello, world!", "sourcetype": "manual"}'
```

Check HEC indexer acknowledgement status

The following example demonstrates how to check the indexing status of a prior HEC request. It sends the request to the `ack` endpoint and includes the `acks` key, which is set to the three acknowledgement identifiers (ackIDs) whose status is queried.

```
# Check ack status
curl
"https://http-inputs-mysplunkserver.splunkcloud.com:8088/services/collector/ack?channel=00872DC6-AC83-4EDE-8AFE-8413C3825C4C" \
  -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
  -d '{"acks": [1,3,4]}'
```

Extract JSON fields from events sent to HEC

The following example demonstrates how to instruct Splunk Enterprise to extract JSON fields from the events sent to HEC.

```
# Extracting JSON fields
curl "https://http-inputs.mysplunkserver.splunkcloud.com:8088/services/collector" \
  -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
  -d '{"sourcetype": "_json", "event": {"a": "value1", "b": ["value1_1", "value1_2"]}}'
```

Extract explicit JSON fields from events sent to HEC

The following example is similar to the previous example, but it explicitly specifies the JSON fields.

```
# Explicit JSON fields
curl "https://mysplunkserver.example.com:8088/services/collector/event" \
  -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
  -d '{"event": "Hello, world!", "sourcetype": "cool-fields", "fields": {"device": "macbook", "users": ["joe", "bob"]}]}'
```

About HTTP Event Collector Indexer Acknowledgment

HTTP Event Collector (HEC) supports **indexer acknowledgment** in Splunk Enterprise only. Splunk Cloud Platform does not offer support for indexer acknowledgment in HEC.

While similar in purpose and identical in name, indexer acknowledgment in HEC is not the same as the indexer acknowledgment capability for forwarding. For information about indexer acknowledgment for forwarding, see protect against loss of in-flight data in the *Splunk Enterprise Forwarding Data* manual. Splunk Cloud Platform also does not offer support for forwarding-based indexer acknowledgment.

Why indexer acknowledgment

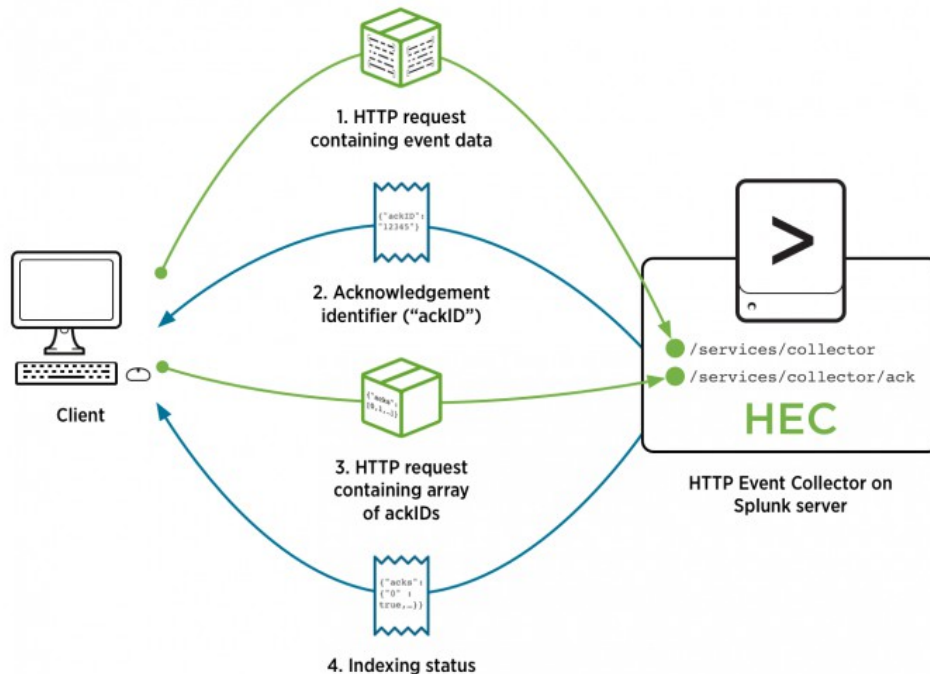
By default, when HEC receives an event successfully, it immediately sends an HTTP Status 200 code to the sender of the data. However, this only means that the event data appears to be valid, and HEC sends the status message before the event data enters the processing pipeline. During processing, there are several places where, because of an outage or a system failure, events can be lost before Splunk Enterprise indexes them. While HEC has precautions in place to prevent data loss, it's impossible to completely prevent such an occurrence, especially in the event of a network failure or hardware crash. This is where indexer acknowledgment comes in.

How indexer acknowledgment works

You can enable indexer acknowledgment on a per-token basis. The indexer acknowledgment process is similar to the following package tracking scenario:

The shipping company issues a tracking number upon shipment of a package. The shipping company updates the status for the tracking number once it's delivered, and then at your convenience, you check whether the package arrived successfully by using the tracking number to retrieve the status.

The following diagram illustrates the indexer acknowledgment process in order from top to bottom. The paragraphs that follow refer to each step by number:



Each time a client sends a request to the HEC endpoint using a token with indexer acknowledgment enabled (1), HEC returns an acknowledgment identifier to the client (2). The response body is a JavaScript Object Notation (JSON) object with the acknowledgment identifier, such as the following:

```
{"ackID": "2"}
```

The client can then query HEC with the identifier to verify whether all the events in the request that corresponds to that identifier have been indexed (3). The client sends the query to a special endpoint (`/services/collector/ack`), and contains JSON-formatted data like the following, where the only key, `"acks"`, is set to an array of the ackIDs whose status you are querying:

```
{"acks": [0, 1, 2]}
```

Next, HEC responds with the status information to the client (4). The body of the reply contains the status of each of the requests that the client queried. A `true` status only indicates that the event that corresponds to that ackID was replicated at the desired replication factor. A `true` status does not guarantee that the event was indexed, because the parsing pipeline might drop events that can't be parsed. A `false` status indicates that there is no status information for that ackID, or that the corresponding event has not been indexed. The corresponding event might not have been indexed yet, the ackID might not have been found, or some other problem might have occurred. For example:

```
{"acks": {"0": true, "1": false, "2": true}}
```

Because a `false` status could indicate any number of problems, only query an ackID during the timeframe in which the request could reasonably be expected to be in transit. Once a client retrieves a `true` status for an ackID, HEC deletes that ackID status information. If you query the same ackID again, HEC will always return `false` for that ackID because its status information can no longer be found. For that reason, avoid querying an ackID again after its status returns as `true`.

Enable indexer acknowledgment for HEC in Splunk Enterprise

You can enable indexer acknowledgment in Splunk Web or in the `inputs.conf` configuration file.

Enable indexer acknowledgment for HEC using Splunk Web

When you create a HEC token in Splunk Web, select the checkbox on the first screen labeled **Enable indexer acknowledgment**. Then continue with the token creation process.

Enable indexer ☐
acknowledgement

For information on creating HEC tokens in Splunk Web, see [Set up and use HTTP Event Collector in Splunk Web](#).

Enable indexer acknowledgment using the `inputs.conf` configuration file

You can enable indexer acknowledgment for existing tokens on Splunk Enterprise by editing the `inputs.conf` configuration file in the HEC app.

1. Open the `inputs.conf` file, which is at the following path:
 - ◆ In *nix: `$SPLUNK_HOME/etc/apps/splunk_httpinput/local/inputs.conf`
 - ◆ In Windows: `%SPLUNK_HOME%\etc\apps\splunk_httpinput\local\inputs.conf`
2. Within the stanza that corresponds to the token for which you want to enable indexer acknowledgment, add the following line: `useACK=true`
3. Save and close the file.

About channels and sending data

Sending events to HEC with indexer acknowledgment active is similar to sending them with the setting off. There is one crucial difference: when you have indexer acknowledgment turned on, you must specify a *channel* when you send events.

The concept of a channel was introduced in HEC primarily to prevent a fast client from impeding the performance of a slow client. When you assign one channel per client, because channels are treated equally on Splunk Enterprise, one client can't affect another.

You must include a matching channel identifier both when sending data to HEC in an HTTP request and when requesting acknowledgment that events contained in the request have been indexed. If you don't, you will receive the error message, "Data channel is missing." Each request that includes a token for which indexer acknowledgment has been enabled must include a channel identifier, as shown in the following example cURL statement, where `<data>` represents the event data portion of the request:

```
curl https://mysplunk.com/services/collector -H "X-Splunk-Request-Channel: FE0ECFAD-13D5-401B-847D-77833BD77131" -H "Authorization: Splunk BD274822-96AA-4DA6-90EC-18940FB2414C" -d '<data>' -v
```

Alternatively, the `X-Splunk-Request-Channel` header field can be sent as a URL query parameter, as shown here:

```
curl https://mysplunk.com/services/collector?channel=FE0ECFAD-13D5-401B-847D-77833BD77131 -H "Authorization: Splunk BD274822-96AA-4DA6-90EC-18940FB2414C" -d '<data>' -v
```

Indexer acknowledgment also works with raw JSON data. In that case, the endpoint to use in requests is `/services/collector/raw`. For more information, see [Format events for HTTP Event Collector](#).

Channels are designed so that you assign a unique channel to each client that sends data to HEC. Each channel has a channel identifier (ID), which must be a Globally Unique Identifier (GUID) but can be randomly generated. You assign channel IDs simply by including them in requests as shown in the examples above. When Splunk Enterprise sees a new channel identifier, it creates a new channel.

Query for indexing status

Once you enable indexer acknowledgment for a token, every request that a client sends to HEC using that token returns the following acknowledgment identifier (ackID) contained in a simple JSON object to the sender, where `<int>` represents a unique integer identifier that corresponds to the request:

```
{"ackID": "<int>"}
```

To verify that the indexer has indexed the event(s) contained in the request, query the following endpoint, where `<host>` and `<port>` represent the hostname and port number of your Splunk platform instance, respectively:

```
https://<host>:<port>/services/collector/ack
```

```
{"acks": [0,1,2]}
```

Following is an example cURL statement that queries Splunk Enterprise for the indexing status of the events contained in the requests with the identifiers "0", "1", "2", and "3":

```
curl https://<host>:<port>/services/collector/ack?channel=FE0ECFAD-13D5-401B-847D-77833BD77131 -H "Authorization: Splunk BD274822-96AA-4DA6-90EC-18940FB2414C"
```

Both the data channel ID (`?channel=FE0ECFAD-13D5-401B-847D-77833BD77131`) and the auth header ("Authorization: Splunk BD274822-96AA-4DA6-90EC-18940FB2414C") are required in this query. For more information, see the previous section [About channels and sending data](#).

The body of the reply contains the status of each of the request(s) for whose status you queried. The following example response indicates that the requests with the ackIDs "0" and "2" were successfully indexed, but the requests with the ackIDs "1" and "3" were not successfully indexed:

```
{"acks": {"0": true, "1": false, "2": true, "3": false}}
```

Channel limits and indexing status expiration

Splunk Enterprise caches acknowledgment IDs and their corresponding status information in memory. To prevent the server from running out of memory, and to prevent malicious or misbehaved clients, several new limit settings have been introduced.

To prevent channels from being overloaded, and to prevent an excessive number of channels from being created, several new settings have been introduced to the `http_input` stanza in the `limits.conf` configuration file:

Setting	Value type	Default value	Description
<code>max_number_of_acked_requests_pending_query_per_ack_channel</code>	integer	1000000	

Setting	Value type	Default value	Description
			Specifies the maximum number of ackIDs and their corresponding status information that are waiting to be queried in each channel. If a client makes many requests with indexer acknowledgment enabled, this setting prevents the client's channel from becoming full of ackIDs and status information and the client from receiving a server busy error.
max_number_of_ack_channel	integer	1000000	Specifies the maximum number of channels that clients can acquire for this Splunk server instance. If a single client tries to acquire more than this number of channels, the request will fail with server busy error. This setting is used to prevent a client from acquiring too many channels.
max_number_of_acked_requests_pending_query	integer	10000000	Specifies the maximum number of ackIDs and their corresponding status information in all channels.

To prevent the likelihood of the limits being reached, Splunk Enterprise can clean up channels that are idle for a period of time and release the memory for those channels. You do this using the following settings, which are set at the global ([http] stanza) level in the inputs.conf configuration file:

- In *nix: \$SPLUNK_HOME/etc/apps/splunk_httpinput/local/inputs.conf
- In Windows: %SPLUNK_HOME%\etc\apps\splunk_httpinput\local\inputs.conf

Parameter	Value type	Default value	Description
ackIdleCleanup	boolean	false	When set to <code>true</code> , this parameter causes the server to remove channels that are idle for the number of seconds set in the <code>maxIdleTime</code> setting.
maxIdleTime	integer	600	Specifies the maximum number of seconds that channels can be idle before they are removed.

Indexer acknowledgment client behavior

To ensure data is successfully ingested into the Splunk platform, configure your clients with the ability to act on response codes that the HEC endpoint returns. If the client can't take an action based on the resulting response code, data loss might occur. For more information, see [Possible error codes](#).

Follow these guidelines to ensure that clients that connect to HEC don't exhibit malicious behavior or end up hitting the limits described earlier in this topic. An indexer acknowledgment client must:

- Create its own GUID to use as its channel identifier.
- Send requests using only that channel.
- Save each acknowledgment identifier (ackID) that HEC returns after a request.
- Continually poll the `/services/collector/ack` REST endpoint at an interval (for instance, every 10 seconds) to confirm that acknowledgment status arrives in a timely manner. Because Splunk Enterprise deletes status information after clients retrieve it, this releases memory on the server.
- Resend any event data for which HEC has not sent an acknowledgment within a certain amount of time, for example, 5 minutes. It is safe to assume that, by that time, the event data has been lost. When you resend the

event data, a good practice is to add some additional data in the event that indicates it may be duplicate data. It's possible the event was previously indexed but the status expired due to the cleanup of the channel, or the HEC status cache might have been cleared.

Scale HTTP Event Collector with distributed deployments

You can use the HTTP event collector (HEC) as part of a distributed Splunk platform deployment. The Splunk software processes HEC data in the same way as it does any other input.

You can configure HEC on Splunk Cloud Platform deployments. When you configure a Splunk Cloud Platform distributed deployment, the indexers in that deployment automatically have HEC configured.

If you want to scale HEC on a Splunk Enterprise distributed deployment, familiarize yourself with distributed Splunk Enterprise deployments before you proceed. See the following related links:

- For more information about distributed deployments, see *Scale your deployment with Splunk Enterprise components* in the *Distributed Deployment Manual* and *Components of a Splunk Enterprise deployment* in the *Capacity Planning Manual*.
- For more information about deployment server, see *About deployment server and forwarder management* in the *Updating Splunk Enterprise Instances* manual.

Where to place HEC

HEC can be placed on indexers or heavy forwarders. HEC, like any data input configuration, must reside on the component when the data enters the system. As a best practice, place HEC on an indexer. HEC is not supported on **universal forwarders**. As with any data input configuration, you can also place HEC directly on a clustered or non-clustered indexer, if necessary.

The deployment server distributes any app that contains your HEC configurations. This configuration includes the following information:

- HTTP Event Collector default values (port, SSL, source type, index)
- SSL settings
- HTTP Event Collector tokens

Each HEC input entry must contain a valid universally unique identifier (UUID) for the token. HEC stores its configurations in the `$SPLUNK_HOME/etc/apps/splunk_httpinput/` directory or `%SPLUNK_HOME%\etc\apps\splunk_httpinput\` on Windows.

Place and distribute HEC on non-clustered indexers

Use HEC with non-clustered indexers in one of two ways:

- Place HEC on heavy forwarders that forward to the indexers by using a deployment server to distribute configurations to the HEC.
- Place HEC directly on the indexers by using a deployment server to distribute configurations to the HEC.

See *About deployment server and forwarder management* in the *Updating Splunk Enterprise Instances* manual to learn more.

For more information about configuring deployment clients, see *Configure deployment clients* in the *Updating Splunk Enterprise Instances* manual.

Place and distribute HEC on indexer cluster peer nodes

Use HEC in a distributed Splunk platform deployment that uses **indexer clustering**.

Use the configuration bundle method to distribute HEC configurations to the peer nodes. Using your HEC port number, preferred protocol (HTTP or HTTPS), SSL settings, and HTTP Event Collector tokens, connect the HECs on your deployment forwarders with the peer nodes of your deployment's indexer cluster. Tokens are managed centrally on the Splunk Enterprise instance running the cluster manager node.

See *Use forwarders to get data into the indexer cluster* and *Update common peer configurations and apps* in the *Managing Indexers and Clusters of Indexers* manual.

Place and distribute HEC on heavy forwarders

Use HEC in a distributed Splunk platform deployment that uses **forwarders**. Use a deployment server to distribute HEC configurations to the heavy forwarders in your Splunk platform deployment.

If you plan to distribute HEC configurations through the deployment server, set the `useDeploymentServer` option in the `[http]` stanza of `inputs.conf` on the deployment server to `1`. When this option is set to `1` and you make UI-based HEC changes on the deployment server, those changes are placed directly in the `$SPLUNK_HOME/etc/deployment-apps/splunk_httpinput/` folder, rather than in `$SPLUNK_HOME/etc/apps/splunk_httpinput/`. See `inputs.conf` for further information.

See *Deploy a heavy forwarder* in the *Forwarding Data* manual to learn more.

Example serverclass.conf file

A **server class** is a group of deployment clients that you can manage as a single unit. You assign the deployment clients you want to use in your HEC deployment to one common server class. Later, when you distribute HEC settings to the deployment clients, only members of that server class receive the configuration settings. The following example `serverclass.conf` file defines a server class `FWD2Local` for HEC.

```
[global]
whitelist.0=*
restartSplunkd=true
stateOnClient = enabled
^
[serverClass:FWD2Local]
whitelist.0=*
[serverClass:FWD2Local:app:splunk_httpinput]
```

For the purposes of deploying HEC settings, you can think of HEC as an app called `splunk_httpinput`. Within the stanzas, you can set client filtering attributes and several non-filtering attributes.

For more information about available client filtering attributes, see *Define filters through serverclass.conf* in the *Updating Splunk Enterprise Instances Manual*. To learn more about available non-filtering attributes, see *Use serverclass.conf to define server classes* in the *Updating Splunk Enterprise Instances Manual*.

See also

For more information about distributed deployment, see the Overview of Splunk Enterprise distributed deployments chapter in the *Distributed Deployment Manual* and Components of a Splunk Enterprise deployment in the *Capacity Planning Manual*.

For more information about distributed deployment, including advanced configuration options and general examples, see the Updating Splunk Instances Manual.

Format events for HTTP Event Collector

The HTTP Event Collector (HEC) receives events from clients in a series of HTTP requests. Each request can contain a HEC token, a channel identifier header, event metadata, or event data, depending on whether your events are raw or have been formatted in accordance with the JavaScript Object Notation (JSON) standard. You can format events for HEC in both Splunk Cloud Platform and Splunk Enterprise.

To learn more about HEC, how it works, and how to set it up, see [Set up and use the HTTP Event Collector in Splunk Web](#).

HEC token

Before the HTTP Event Collector can accept your data for indexing, you must authenticate to the Splunk Cloud Platform or Splunk Enterprise instance on which it runs. You do this using the token you generate when you create a new HEC input. When you use the token management endpoint on the Splunk server to generate a token, it generates the token in the form of a globally unique identifier (GUID). See [Use cURL to manage HTTP Event Collector tokens, events, and services](#) for more information. Using the token management endpoint guarantees that the token is unique.

You have several ways to authenticate to the instance: by HTTP authentication, basic authentication, or a query string.

HTTP authentication

Place the token in the authorization header of each HTTP request as follows:

```
"Authorization: Splunk <hec_token>"
```

The following example shows HTTP authentication in the context of a typical `curl` command that you use to communicate with HEC:

```
curl -H "Authorization: Splunk 12345678-1234-1234-1234-1234567890AB"
https://mysplunkserver.example.com:8088/services/collector/event -d '{"sourcetype": "my_sample_data",
"event": "http auth ftw!"}'
```

Basic authentication

To perform basic authentication into HEC, include a colon-separated user-password pair in the request after `-u`, inserting the HEC token as the `<password>`: `"<user>:<password>"`. The `<user>` can be any string. For example:

```
-u "x:<hec_token>"
```

The following example shows the basic authentication in context of a `curl` command:

```
curl -u "x:12345678-1234-1234-1234-1234567890AB"
https://mysplunkserver.example.com:8088/services/collector/event -d '{"sourcetype": "my_sample_data",
"event": "basic auth ftw!"}'
```

Query string

You can specify the HEC token as a query string in the URL that you specify in your queries to HEC. For example:

```
?token=<hec_token>
```

The following example shows the query string authentication in context of a `curl` command:

```
curl https://mysplunkserver.example.com:8088/services/collector/event?token=12345678-1234-1234-1234-1234567890AB
-d '{"sourcetype": "my_sample_data", "event": "query string ftw!"}'
```

You must also enable query string authentication on a per-token basis. Open a case with Splunk Support to edit the file in the `$SPLUNK_HOME/etc/apps/splunk_httpinput/local/inputs.conf` file. Your tokens appear by name in this file, in the form of `http://<token_name>`.

Within the stanza for each token you want to enable query string authentication, have Splunk Support add or change the following setting:

```
allowQueryStringAuth = true
```

On Splunk Enterprise, you can make these configurations directly on the instance. The changes take effect after you restart the instance.

Channel identifier header

If the request to HEC includes raw events and indexer acknowledgement is enabled for the HEC token, you must include the `X-Splunk-Request-Channel` header field in the request. You must set the header field to a unique channel identifier (GUID). See [About channels and sending data](#) for more information. The following example shows a `cURL` command that constitutes a valid request:

```
curl https://http-inputs-<customer>.splunkcloud.com/services/collector/raw -H "X-Splunk-Request-Channel:
FE0ECFAD-13D5-401B-847D-77833BD77131" -H "Authorization: Splunk BD274822-96AA-4DA6-90EC-18940FB2414C" -d
'<raw data string>' -v
```

Alternatively, you can set the `X-Splunk-Request-Channel` header field as a URL query parameter:

```
curl https://http-inputs-<customer>.splunkcloud.com/services/collector/raw?channel=FE0ECFAD-13D5-401B-847D
-77833BD77131 -H "Authorization: Splunk BD274822-96AA-4DA6-90EC-18940FB2414C" -d '<raw data string>' -v
```

If the token with which you're authenticating to HTTP Event Collector has indexer acknowledgement enabled, you must also include the channel identifier with your indexer status query. For more information, see [Enable indexer acknowledgement](#).

Event metadata

The following table shows keys that you can include in event metadata. These keys are all optional. Any key-value pairs that you don't include in the event are set to values defined for the token on the Splunk platform instance.

Key	Description
"time"	The event time. The default time format is UNIX time format, in the format <code><sec>.<ms></code> and depends on your local timezone. For example, <code>1433188255.500</code> indicates 1433188255 seconds and 500 milliseconds after epoch, or Monday, June 1, 2015, at 7:50:55 PM GMT.
"host"	The host value to assign to the event data. This key is typically the hostname of the client from which you're sending data.

Key	Description
"source"	The source value to assign to the event data. For example, if you're sending data from an app you're developing, set this key to the name of the app.
"sourcetype"	The sourcetype value to assign to the event data.
"index"	The name of the index by which the event data is to be indexed. The index you specify here must be within the list of allowed indexes if the token has the <code>indexes</code> parameter set.
"fields"	The fields key isn't applicable to raw data. This key specifies a JSON object that contains a flat (not nested) list of explicit custom fields to be defined at index time. Requests containing the "fields" property must be sent to the <code>/collector/event</code> endpoint, or else they aren't indexed. For more information, see Indexed field extractions .

With raw events, you can configure metadata at the global level, at the token level, and at the request level using the query string. See [About Event Collector tokens](#) and [Use HEC from the CLI](#) for more information. Metadata specified within a request applies to all events that are extracted from the request.

Event data

Event data can be assigned to the "event" key within the JSON object in the HTTP request, or it can be raw text. The "event" key is at the same level within the JSON event packet as the metadata keys. Within the "event" key-value curly brackets, the data can be in whatever format you want: a string, a number, another JSON object, and so on.

You can batch multiple events in one event packet by combining them within the request. By batching the events, you're specifying that any event metadata within the request is to apply to all of the events contained in the request. Batching events can significantly speed performance when you need to index large quantities of data.

Example 1: Event metadata as a string contained in a JSON

The following example is of a properly formatted event metadata and event data as a string contained within a JSON object:

```
{
  "time": 1426279439, // epoch time
  "host": "localhost",
  "source": "random-data-generator",
  "sourcetype": "my_sample_data",
  "index": "main",
  "event": "Hello world!"
}
```

Example 2: JSON object as event data

The following example is of a JSON object as the event data within a properly formatted event:

```
{
  "time": 1437522387,
  "host": "dataserver992.example.com",
  "source": "testapp",
  "event": {
    "message": "Something happened",
    "severity": "INFO"
  }
}
```

Example 3: Batched data

The following example is of batched data. The batch protocol for HTTP Event Collector involves event objects stacked one after the other, and not in a JSON array. These events, although they contain only the "event" and "time" keys, are still valid:

```
{
  "event": "event 1",
  "time": 1447828325
}

{
  "event": "event 2",
  "time": 1447828326
}
```

Example 4: cURL statement with authorization header

The following example is a simplified "hello world!" cURL statement that includes the authorization header, a destination endpoint, and simple event data. Note that the request is going to the `/services/collector/event` endpoint, which is where all JSON-formatted event requests must go:

```
curl -H "Authorization: Splunk 12345678-1234-1234-1234-1234567890AB"
https://localhost:8088/services/collector/event -d '{"event": "hello world"}'
```

Example 5: cURL statement sending raw data

The following example cURL statement demonstrates sending raw event data. Note the addition of the channel ID, which is required when sending raw event data. Also, the request is going to the `/services/collector/raw` endpoint, which is where all raw event requests must go:

```
curl http://localhost:8088/services/collector/raw -H 'Authorization: Splunk
B5A79AAD-D822-46CC-80D1-819F80D7BFB0' -H 'x-splunk-request-channel: 18654C68-B28B-4450-9CF0-6E7645CA60CA'
-d 'hello world'
```

Example 6: cURL statement as a URL parameter

Alternately, this example cURL statement passes the channel ID as a URL parameter:

```
curl http://localhost:8088/services/collector/raw?channel=18654C68-B28B-4450-9CF0-6E7645CA60CA -H
'Authorization: Splunk B5A79AAD-D822-46CC-80D1-819F80D7BFB0' -d 'hello world'
```

For additional examples on how to format and send event data using cURL, see [Use cURL to manage HTTP Event Collector tokens, events, and services](#).

Event parsing

The HTTP Event Collector endpoint extracts the events from the HTTP request and parses them before sending them to indexers. Because the event data formats are predetermined, the Splunk platform can parse and index your data quickly. This faster parsing results in improved data throughput and reduced event processing time compared to other methods of getting data in.

You can configure extraction rules in the `props.conf` configuration file. To learn more, see [Configure rule-based source type recognition](#).

Raw event parsing

Raw event parsing is available in the current release of Splunk Cloud Platform and Splunk Enterprise 6.4.0 and higher.

HTTP Event Collector can parse raw text and extract one or more events. HEC expects that the HTTP request contains one or more events with line-breaking rules in effect. Once HEC accepts the request, it passes its events into the pipeline, which extracts the fields, such as timestamps. HEC uses a line-breaking strategy that is based on the timestamp, but you can override the strategy by setting a sourcetype in the props.conf configuration file. See [Configure rule-based source type recognition](#).

You must include events within a single HTTP request. They cannot span multiple requests.

To accommodate raw events, use the services/collector/raw endpoint.

This endpoint requires an additional X-Splunk-Request-Channel header field, which you must set to a unique channel identifier (GUID). See [About channels and sending data](#) for more information. You must include a channel identifier with each HTTP request that contains raw events. The following example is of a cURL statement that constitutes a valid request:

```
curl https://http-inputs-<customer>.splunkcloud.com/services/collector/raw -H "X-Splunk-Request-Channel: FE0ECFAD-13D5-401B-847D-77833BD77131" -H "Authorization: Splunk BD274822-96AA-4DA6-90EC-18940FB2414C" -d '<raw data string>' -v
```

Alternatively, the X-Splunk-Request-Channel header field can be sent as a URL query parameter:

```
curl https://http-inputs-<customer>.splunkcloud.com/services/collector/raw?channel=FE0ECFAD-13D5-401B-847D-77833BD77131 -H "Authorization: Splunk BD274822-96AA-4DA6-90EC-18940FB2414C" -d '<raw data string>' -v
```

If the token with which you're authenticating to HTTP Event Collector has indexer acknowledgement enabled, you must also include the channel identifier with your indexer status query. For more information, see [Enable indexer acknowledgement](#).

With raw events, you can configure metadata at the global level, at the token level, and at the request level using the query string. See [About Event Collector tokens](#) and [Use HEC from the CLI](#) for more information. Metadata specified within a request applies to all events that are extracted from the request.

Timestamp extraction rules are enabled at the sourcetype level to extract timestamps. The most common timestamp formats are recognized, such as the "current-time" key, but if no timestamp can be extracted, one is assigned based on the current time. For other metadata, you can configure extraction rules in the props.conf file. See [Configure rule-based source type recognition](#) for more information.

For more examples of cURL requests to the services/collector/raw endpoint, see Input endpoint examples in the Splunk Enterprise *REST API Reference Manual*.

For more information about channels, see the *About channels and sending data* in [Enable indexer acknowledgement](#).

Automate indexed field extractions with HTTP Event Collector

When Splunk Enterprise indexes data, it parses the data stream into a series of events. As part of this process, it adds a number of fields to the event data. These fields include default fields that it adds automatically and any custom fields that you specify. The process of adding fields to events is known as field extraction. There are two types of field extraction: search-time field extraction and indexed field extraction. Indexed fields are incorporated into the index at index time and

become part of the event data.

Indexed field extraction doesn't work with data sent to the **services/collector/raw** endpoint. For more information, see **services/collector/raw** in the Splunk Enterprise *REST API Reference Manual*.

Form HEC requests to trigger indexed field extractions

You can trigger indexed extractions of JavaScript Object Notation (JSON) fields in two ways: as part of the main `event` data or as separate from the `event` data but still associated with the event.

Use nested JSON inside the event property

Assign the `event` property, which is at the top level of the JSON being sent to HEC, to a JSON object that contains the custom fields you want to index as key-value pairs. For example, the following `event` property, from within an HTTP request sent to HEC, specifies two custom fields: `club` and `wins`.

```
"event": {"club": "glee", "wins": ["regionals", "nationals"]}
```

In this example, the `wins` property is a multi-value JSON array. The `wins` field is assigned both the values in the array.

At the same level as the `event` property, you must also include a `sourcetype` property, and set it to a source type that has indexed extraction enabled. You can use any source type that has the `INDEXED_EXTRactions` setting configured to `JSON` in the `props.conf` configuration file, including built-in source types such as `_json`. See the following example:

```
"sourcetype": "_json"
```

Following is an example cURL command that sends an event to HEC on a Splunk Enterprise instance. In this case, the event data contains two custom fields that are extracted at index time:

```
# Extracting JSON fields
curl https://mysplunkserver.example.com:8088/services/collector -H "Authorization: Splunk 12345678-1234-1234-1234-1234567890AB" -d '{"sourcetype": "_json", "event": {"club": "glee", "wins": ["regionals", "nationals"]}}'
```

Add a fields property at the top JSON level

Include the `fields` property at the top level of the JSON that you send to HEC, which is at the same level as the `event` property. Adding this property specifies explicit custom fields that are separate from the main `event` data. This method is useful if you don't want to include the custom fields with the event data, but you want to annotate the data with some extra information, such as where it came from. Using this method is also typically faster than the nested JSON method.

Be aware that you must send HEC requests containing the `fields` property to the **/collector/event** endpoint. Otherwise, they aren't indexed.

Assign the `fields` property to a JSON object that contains the custom fields to be indexed as key-value pairs. For example, the following `fields` property, from within an HTTP request sent to the Splunk platform instance, specifies two custom fields: `club` and `wins`.

```
"fields": {"club": "glee", "wins": ["regionals", "nationals"]}
```

In this example, the `wins` property is set to a multi-value JSON array. The `wins` field is assigned both values in the array.

At the same level as the `event` and `fields` properties, you must also include a `sourcetype` property and set it to a source type that has indexed extractions enabled. You can use any source type that has the `INDEXED_EXTRactions` setting configured to `JSON` in the `props.conf` file, including built-in source types such as `_json`. See the following example:

```
"sourcetype": "_json"
```

Following is an example cURL command that sends an event to HEC on a Splunk Enterprise instance. In this case, the event data contains two custom fields that will be extracted at index time:

```
curl http://mysplunkserver.example.com:8088/services/collector/event -H "Authorization: Splunk 12345678-1234-1234-1234-1234567890AB" -d '{"event": {"my_event": "Hello, McKinley High!"}, "sourcetype": "_json", "fields": {"club": "glee", "wins": ["regionals", "nationals"]}}'
```

Only strings can be used as field values.

Search for index-extracted fields

After the data is indexed, you can search for this event using a double-colon (`::`) indexed extraction notation, as shown here:

```
sourcetype=_json club::glee
```

For more information about using extracted fields to retrieve events, see [Use fields to retrieve events in the Splunk Enterprise *Search Manual*](#).

Send metrics to a metrics index

If you gather metrics data, you can send the data directly to a metrics index using an HTTP Event Collector (HEC).

The most updated instructions for sending data to a metrics index are in the following topics in the Splunk Enterprise *Metrics Manual*:

- [Get metrics in from collectd](#)
- [Get metrics in from other sources](#)

See also

- [Overview of metrics in the Splunk Enterprise *Metrics manual*](#)
- [Create metrics indexes in the Splunk Enterprise *Managing Indexers and Clusters of Indexers manual*](#)
- [/collector and /collector/raw in the Splunk Enterprise *REST API Reference Manual*](#)

HTTP Event Collector REST API endpoints

You can find the complete HTTP Event Collector REST API endpoint reference in the Splunk Enterprise **REST API Reference Manual**. Each endpoint is linked here for your convenience.

REST API endpoint	Description
data/inputs/http	Accesses or updates HTTP Event Collector global configuration tokens and application tokens.
data/inputs/http/{name}	Manages the {name} HTTP Event Collector token. HTTP, as in data/inputs/http/http, indicates global configuration.
data/inputs/http/{name}/disable	Disables the {name} HTTP Event Collector token.
data/inputs/http/{name}/enable	Enables the {name} HTTP Event Collector token.
services/collector	Sends events to HTTP Event Collector using the Splunk platform JavaScript Object Notation (JSON) event protocol.
services/collector/ack	Queries event indexing status.
services/collector/event	<p>Sends timestamped events to the HTTP Event Collector using the Splunk platform JSON event protocol when you set the <code>auto_extract_timestamp</code> argument to <code>true</code> in the <code>/event</code> URL.</p> <ul style="list-style-type: none"> • An example of a timestamp is <code>2017-01-02 00:00:00</code> • If there's a timestamp in the event's JSON envelope, Splunk honors that timestamp first. • If there's no timestamp in the event's JSON envelope, the merging pipeline extracts the timestamp from the event. • If <code>time=xxx</code> is used in the <code>/event</code> URL, then <code>auto_extract_timestamp</code> is disabled.
services/collector/event/1.0	This endpoint works identically to the services/collector/event endpoint but introduces a protocol version for future scalability.
services/collector/health	Checks the health of the HTTP Event Collector. This endpoint is supported in Splunk Cloud Platform and versions 6.6.0 and higher of Splunk Enterprise.
services/collector/mint	Posts data formatted for Splunk MINT to the HTTP Event Collector.
services/collector/mint/1.0	This endpoint works identically to the receivers/token/mint endpoint but introduces a protocol version for future scalability.
services/collector/raw	Sends raw data directly to the HTTP Event Collector.
services/collector/raw/1.0	This endpoint works identically to the services/collector/raw endpoint but introduces a protocol version for future scalability.
services/collector/s2s	This endpoint receives Splunk TCP data over HTTP from the Splunk Universal Forwarder. Compatible with Splunk Enterprise 8.1.0 and higher.

HTTP Event Collector examples

The HTTP Event Collector (HEC) input has a myriad of use cases. The following examples show how you can use HEC to index streams of data. They also show how you must send data to the HEC input. You can use these examples to model how to send your own data to HEC in either Splunk Cloud Platform or Splunk Enterprise.

The examples on this page use the `curl` command. Typically, the example commands use the following arguments:

Argument	Description
<code>-d</code>	Use this argument to supply events to HEC. You can send raw text or text in JSON format to HEC.
<code>-u</code>	Use this argument to specify a user. This argument is required when you use basic authentication.
<code>-H</code>	Use this argument to specify a header. You must supply a header to submit events to HEC whether you use HTTP authentication or basic authentication. The header is how you include the HEC token.

The `-k` argument is insecure, so don't use it to check security certificates. Don't use this argument in a production environment or where security is necessary.

There's no requirement to use the `curl` command to submit events to HEC. You can use any tool or application that is compatible with the HTTP and REST specifications.

Example 1: Basic example

This example demonstrates basic HEC usage. It includes the Splunk platform instance address, port, and REST endpoint, as well as the authentication token, event data, and metadata. The example is formatted according to the HEC event data format specification.

```
curl "https://mysplunkserver.example.com:8088/services/collector" \
  -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
  -d '{"event": "Hello, world!", "sourcetype": "manual"}'
```

Example 2: Send multiple events to HEC

This example demonstrates how to send multiple events in one request. While you can send multiple events in a single request, you can't split one event across multiple requests.

```
curl "https://mysplunkserver.example.com:8088/services/collector" \
  -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
  -d '{"event": "Pony 1 has left the barn"}{"event": "Pony 2 has left the barn"}{"event": "Pony 3 has left the barn", "nested": {"key1": "value1"}}'
```

Example 3: Send raw text to HEC

This example demonstrates sending raw text to HEC. To send raw text, you must use the **raw** endpoint, plus the channel identifier and source type specification. You submit both of these settings using URL query parameters.

```
curl "https://mysplunkserver.example.com:8088/services/collector/raw?channel=00872DC6-AC83-4EDE-8AFE-8413C3825C4C&sourcetype=mydata" -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" -d '1, 2, 3... Hello, world!'
```

Example 4: Send multiple raw text events to HEC

This example demonstrates how to send raw, batched events to HEC. In this case, the command sends splunkd access logs. The command indicates that the indexer is to assign these events the source type of `splunkd_access`, and specifies that they are to go into the **main** index.

```
curl "https://mysplunkserver.example.com:8088/services/collector/raw?channel=00872DC6-AC83-4EDE-8AFE-8413C3825C4C&sourcetype=splunkd_access&index=main" \
  -H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
  -d '127.0.0.1 - admin [28/Sep/2016:09:05:26.875 -0700] "GET /servicesNS/admin/launcher/data/ui/views?count=-1 HTTP/1.0" 200 126721 - - - 6ms
127.0.0.1 - admin [28/Sep/2016:09:05:26.917 -0700] "GET /servicesNS/admin/launcher/data/ui/nav/default HTTP/1.0" 200 4367 - - - 6ms
127.0.0.1 - admin [28/Sep/2016:09:05:26.941 -0700] "GET /services/apps/local?search=disabled%3Dfalse&count=-1 HTTP/1.0" 200 31930 - - - 4ms
127.0.0.1 - admin [28/Sep/2016:09:05:26.954 -0700] "GET /services/apps/local?search=disabled%3Dfalse&count=-1 HTTP/1.0" 200 31930 - - - 3ms
127.0.0.1 - admin [28/Sep/2016:09:05:26.968 -0700] "GET /servicesNS/admin/launcher/data/ui/views?digest=1&count=-1 HTTP/1.0" 200 58672 - - - 5ms'
```

Example 5: Send multiple metrics at once using HEC

This example demonstrates how to send JSON-formatted events with multiple metrics using HEC. See the [Get Metrics](#) page in the Splunk Enterprise manual for more information.

```
{
  "time": 1486683865,
  "event": "metric",
  "source": "metrics",
  "sourcetype": "perflog",
  "host": "host_1.splunk.com",
  "fields": {
    "region": "us-west-1",
    "datacenter": "dc2",
    "rack": "63",
    "os": "Ubuntu16.10",
    "arch": "x64",
    "team": "LON",
    "service": "6",
    "service_version": "0",
    "service_environment": "test",
    "path": "/dev/sda1",
    "fstype": "ext3",
    "metric_name:cpu.usr": 11.12,
    "metric_name:cpu.sys": 12.23,
    "metric_name:cpu.idle": 13.34
  }
}
```

Example 6: Indexer acknowledgement of HEC event data

This example demonstrates how to send events to HEC with indexer acknowledgement of incoming HEC data. The difference between this example and the basic example is the inclusion of a channel identifier. Indexer acknowledgement also works with raw data.

```
curl
"https://mysplunkserver.example.com:8088/services/collector?channel=00872DC6-AC83-4EDE-8AFE-8413C3825C4C" \
-H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
-d '{"event": "Hello, world!", "sourcetype": "manual"}'
```

Example 7: Check indexer acknowledgement status

This example demonstrates how to check the indexing status of a previous HEC request. The command sends the request to the **ack** REST endpoint, and includes the `acks` key, which you set to be equal to the three acknowledgement identifiers whose status you want to see.

```
curl
"https://mysplunkserver.example.com:8088/services/collector/ack?channel=00872DC6-AC83-4EDE-8AFE-8413C3825C4C" \
-H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \
-d '{"acks": [1,3,4]}'
```

Example 8: Extract JSON fields

This example demonstrates how to instruct the Splunk platform to extract JSON fields from the events you send to HEC.

```
curl "https://mysplunkserver.example.com:8088/services/collector" \  
-H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \  
-d '{"sourcetype": "_json", "event": {"a": "value1", "b": ["value1_1", "value1_2"]}}'
```

Example 9: Explicit JSON fields

This example is similar to the previous example, but it explicitly specifies the JSON fields.

```
curl "https://mysplunkserver.example.com:8088/services/collector/event" \  
-H "Authorization: Splunk CF179AE4-3C99-45F5-A7CC-3284AA91CF67" \  
-d '{"event": "Hello, world!", "sourcetype": "cool-fields", "fields": {"device": "macbook", "users":  
["joe", "bob"]}}'
```

Example 10: Basic authentication

This example demonstrates basic authentication, which is an alternative to the HTTP authentication used in the previous examples. To use basic authentication, use the `-u` argument to include a colon-separated user-password pair in the request. You can use anything for the `<user>` string and the token is the `<password>`.

```
curl -u "x:CF179AE4-3C99-45F5-A7CC-3284AA91CF67"  
"https://mysplunkserver.example.com:8088/services/collector/event" \  
-d '{"sourcetype": "mysourcetype", "event": "Hello, world!"}'
```

Troubleshoot HTTP Event Collector

You can troubleshoot HTTP Event Collector (HEC) by viewing error logs. You can also set up logging using configuration files, investigate instance performance with dashboards included in the Monitoring Console, and detect other scaling problems.

Logging

HTTP Event Collector saves usage data about itself to log files. You can search these usage metrics using Splunk Cloud Platform or Splunk Enterprise to explore usage trends system-wide, per token, per source type, and more, as well as to evaluate HEC performance. Metrics are logged whenever HEC is active. HEC is disabled by default, so it does not log data until you enable it.

You can also view HEC error logs in the `splunkd.log` log file on Splunk Enterprise. See *Enable debug logging in the Troubleshooting Manual* for how to enable debugging on your Splunk Enterprise instance.

Log file location and management

Splunk Enterprise writes HTTP Event Collector metrics to the `$SPLUNK_HOME/var/log/introspection/splunk/http_event_collector_metrics.log` file.

The Splunk platform creates a new `http_event_collector_metrics.log` file when you log off of and back on to Splunk Cloud Platform or start your Splunk Enterprise instance. Any existing file with that name is renamed.

You configure the logging frequency of HTTP Event Collector metrics in the `limits.conf` configuration file. 60 seconds is the default frequency. HEC continues logging system-level metrics even when there is no data input activity. When there is no

activity, you can expect about 200 kilobytes (KB) of metrics log data to be produced every 24 hours. The maximum size of a metrics log file is 25 megabytes (MB). If a log file reaches that limit, the Splunk platform renames the log file and creates a new file. Up to five metrics log files can be stored at a time.

The props.conf configuration file defines parameters for reading and indexing the metrics log file.

Searching HTTP Event Collector metrics data

The Splunk platform puts HEC metrics data into the `_introspection` index. To search the accumulated HEC metrics with the Splunk platform, use the following search command:

```
index="_introspection" token
```

Metrics log data format

The Splunk platform records HEC metrics data to the log in JSON format. This means that the log is both human-readable and consistent with other Splunk Cloud Platform or Splunk Enterprise log formats. A single entry consists of both input summary metrics (`series = http_event_collector`) and per-token metrics (`series = http_event_collector_token`), as shown in the following example:

```
{
  "datetime": "09-01-2016 19:21:19.014 -0700",
  "log_level": "INFO",
  "component": "HttpEventCollector",
  "data": {
    "series": "http_event_collector",
    "transport": "http",
    "format": "json",
    "total_bytes_received": 0,
    "total_bytes_indexed": 0,
    "num_of_requests": 0,
    "num_of_events": 0,
    "num_of_errors": 0,
    "num_of_parser_errors": 0,
    "num_of_auth_failures": 0,
    "num_of_requests_to_disabled_token": 0,
    "num_of_requests_to_incorrect_url": 0,
    "num_of_requests_in_mint_format": 0,
    "num_of_ack_requests": 0,
    "num_of_requests_acked": 0,
    "num_of_requests_waiting_ack": 0
  }
}

{
  "datetime": "08-22-2016 12:38:04.854 -0700",
  "log_level": "INFO",
  "component": "HttpEventCollector",
  "data": {
    "token_name": "test",
    "series": "http_event_collector_token",
    "transport": "http",
    "format": "json",
    "total_bytes_received": 57000,
    "total_bytes_indexed": 44000,
    "num_of_requests": 1000,
    "num_of_events": 1000,
  }
}
```

```

    "num_of_errors":0,
    "num_of_parser_errors":0,
    "num_of_requests_to_disabled_token":0,
    "num_of_requests_in_mint_format":0
  }
}

```

HEC summary metrics

The Splunk platform accumulates system-wide summary metrics even if there is no input activity. These metrics are identified by "series": "http_event_collector".

See the following table for a description of the fields for HEC summary metrics:

Field	Description	Value
component	HTTP Event Collector metrics data identifier.	HttpEventCollector
data:format	HTTP Event Collector data format.	json
data:num_of_auth_failures	Total number of authentication failures due to invalid token.	unsigned integer
data:num_of_errors	Total number of per-token errors, which include the following options: <ul style="list-style-type: none"> • Nad data format • No authorization • Bad authorization • Connectivity problems 	unsigned integer
data:num_of_events	Total number of per-token events received by the HTTP Event Collector endpoint.	unsigned integer
data:num_of_parser_errors	Total number of per-token parser errors due to incorrectly formatted event data.	unsigned integer
data:num_of_requests	Total number of valid per-token individual HTTP or HTTPS requests received by an HTTP Event Collector endpoint. Each request can have one or more data events.	unsigned integer
data:num_of_ack_requests	Total number of HEC request indexer status queries received.	unsigned integer
data:num_of_requests_acked	Total number of HEC requests that Splunk successfully indexed and acknowledged.	unsigned integer
data:num_of_requests_waiting_ack	Total number of HEC requests received with indexer acknowledgements enabled.	unsigned integer
data:num_of_requests_to_incorrect_url	Total number of requests to an incorrect URL.	unsigned integer
data:num_of_requests_in_mint_format	Total number of requests from Splunk MINT.	unsigned integer
data:num_of_requests_to_disabled_token	Total number of per-token requests to disable token.	unsigned integer
data:series	Metrics data type.	http_event_collector
data:total_bytes_indexed	Total amount of per-token data sent to the indexer.	unsigned integer
data:total_bytes_received	Total amount of per-token data received by calling the <code>receive/token</code> endpoint.	unsigned integer
data:transport	Data transport protocol for HTTP Event Collector data.	http
datetime	Date and time associated with the data. Takes the following format: MM-DD-YYYY HH:MM:SS.SSS +/-GMTDELTA	string
log_level	Log severity level.	INFO

Field	Description	Value

Per-token metrics

In contrast to the system-wide summary metrics, the Splunk platform accumulates per-token metrics only when HEC is active. These metrics are identified by "series": "http_event_collector_token".

The `[http_input]` stanza in the `limits.conf` configuration file defines the logging interval and maximum number of tokens logged for these metrics.

See the following table for a description of the fields for per-token metrics:

Field	Description	Value
component	HTTP Event Collector metrics data identifier.	HttpEventCollector
data:format	HTTP Event Collector data format. Always JSON format for metrics logging.	json
data:num_of_errors	Number of errors, which include the following: <ul style="list-style-type: none"> • Bad data format • No authorization • Bad authorization • Connectivity problems 	unsigned integer
data:num_of_events	Number of events received by the HTTP Event Collector endpoint.	unsigned integer
data:num_of_parser_errors	Number of parser errors due to incorrectly formatted event data.	unsigned integer
data:num_of_requests	Number of valid individual HTTP or HTTPS requests received by an HTTP Event Collector endpoint. Each request can have one or more data events.	unsigned integer
data:num_of_requests_in_mint_format	Total number of requests from Splunk MINT.	unsigned integer
data:num_of_requests_to_disabled_token	Number of requests to a disabled token.	unsigned integer
data:series	Metrics data type.	http_event_collector_token
data:token_name	Token name.	string
data:total_bytes_indexed	Total amount of data sent to the indexer.	unsigned integer
data:total_bytes_received	Total amount of data received by calling the <code>receive/token</code> endpoint.	unsigned integer
data:transport	Data transport protocol for HTTP Event Collector data.	http
datetime	Date and time associated with the data. Takes the following format: <code>MM-DD-YYYY HH:MM:SS.SSS +/-GMTDELTA</code>	string
log_level	Log severity level.	INFO

Logging with configuration files

The `limits.conf` and `props.conf` files control metrics data logging and indexing behavior.

limits.conf

The `[http_input]` stanza in the `$SPLUNK_HOME/etc/system/default/limits.conf` file controls HTTP Event Collector metrics data logging.

For information about all HTTP Event Collector-related parameters, including those not related to metrics, see the [http_input] stanza documentation on limits.conf in the Splunk Enterprise Admin Manual.

Limits.conf takes the following parameters:

Parameter	Default value	Description
max_number_of_tokens	10000	An unsigned integer that represents the maximum number of tokens reported by HTTP Event Collector metrics.
metrics_report_interval	60	An unsigned integer that represents the number of seconds in an HTTP Event Collector metrics report interval.

props.conf

The [http_event_collector_metrics] stanza in the \$SPLUNK_HOME/etc/system/default/props.conf file controls reading and indexing the HTTP Event Collector log files.

See the following example:

```
[source::.../http_event_collector_metrics.log(.\\d+)?]
sourcetype = http_event_collector_metrics
```

...

```
[http_event_collector_metrics]
SHOULD_LINEMERGE = false
TIMESTAMP_FIELDS = datetime
TIME_FORMAT = %m-%d-%Y %H:%M:%S.%l %z
INDEXED_EXTRACTIONS = json
KV_MODE = none
JSON_TRIM_BRACES_IN_ARRAY_NAMES = true
```

Props.conf takes the following parameters:

Parameter	Default	Description
SHOULD_LINEMERGE	false	Specifies layout of events per line. Setting to true allows multiple events in the same line. Setting to false puts multiple events in separate lines.
TIMESTAMP_FIELDS	datetime	Log entry time field name.
TIME_FORMAT	%m-%d-%Y %H:%M:%S.%l %z	Log entry time field format.
INDEXED_EXTRACTIONS	json	Metrics log format. Always in JSON format for metrics logging.
KV_MODE	none	Key-value data indicator. Setting to none means no key-value data. Always none for metrics logging.
JSON_TRIM_BRACES_IN_ARRAY_NAMES	true	Whether to trim brace characters from JSON array names.

Possible error codes

The following status codes have particular meaning for all HTTP Event Collector endpoints:

Status code	HTTP status code ID	HTTP status code	Status message
-------------	---------------------	------------------	----------------

Status code	HTTP status code ID	HTTP status code	Status message
0	200	OK	Success
1	403	Forbidden	Token disabled
2	401	Unauthorized	Token is required
3	401	Unauthorized	Invalid authorization
4	403	Forbidden	Invalid token
5	400	Bad Request	No data
6	400	Bad Request	Invalid data format
7	400	Bad Request	Incorrect index
8	500	Internal Error	Internal server error
9	503	Service Unavailable	Server is busy
10	400	Bad Request	Data channel is missing
11	400	Bad Request	Invalid data channel
12	400	Bad Request	Event field is required
13	400	Bad Request	Event field cannot be blank
14	400	Bad Request	ACK is disabled
15	400	Bad Request	Error in handling indexed fields
16	400	Bad Request	Query string authorization is not enabled

To ensure data is successfully ingested into the Splunk platform, configure your clients with the ability to act on response codes returned by the HEC endpoint. If the client can't take an action based on the resulting response code, data loss might occur.

Investigate instance performance with the Monitoring Console

The **Monitoring Console** provides pre-built dashboards for HEC that you can use to investigate your instance performance. See the following topics for more information:

- For Splunk Cloud Platform, see the Monitor your Splunk Cloud Platform Deployment chapter in the *Splunk Cloud Platform Admin Manual*.
- For Splunk Enterprise, see the About the Monitoring Console chapter in the *Monitoring Splunk Enterprise* manual.

The Monitoring Console provides a pre-built dashboard to monitor HTTP Event Collector. See Indexing: Inputs: HTTP Event Collector in the *Monitoring Splunk Enterprise* manual.

Detect scaling problems

If you are experiencing performance slowdowns or want to speed up your HTTP Event Collector deployment, the following factors can affect performance.

HTTP and HTTPS

Sending data over HTTP results in a significant performance improvement compared to sending data over HTTPS.

Batching

If you batch multiple events into single requests, it can speed up data transmission. Because the request metadata applies to all events in the request, less data is sent overall. For more information about how event data is packaged, see [Format events for HTTP Event Collector](#).

HTTP Keep-alive

Setting keep-alive on your connection can improve performance. As long as the client sending the data supports HTTP 1.1 and is set up to support HTTP persistent connection, you can optimize performance with keep-alive.

Persistent queues

Persistent queuing slows down performance by storing data in an input queue to disk. For more information, see [Use persistent queues to help prevent data loss](#).

Get other kinds of data in

Monitor First In, First Out (FIFO) queues

You can configure a First In, First Out (FIFO) input by editing the `inputs.conf` configuration file on a Splunk Enterprise instance. If you use Splunk Cloud Platform, use a heavy forwarder to read, index, and forward FIFO queues. Splunk Web doesn't support the definition of FIFO inputs.

Data that you send over FIFO queues doesn't remain in computer memory and can be an unreliable method for data sources. To ensure data integrity, use the **monitor** input instead. For more information on the **monitor** input, see [Monitor files and directories](#).

Add a FIFO input to `inputs.conf`

If you haven't worked with configuration files before, read [About Configuration Files](#) in the Splunk Enterprise *Admin Manual* before you begin.

To add a FIFO input, edit the `inputs.conf` file and add a FIFO input stanza. Add the stanza to the `inputs.conf` file in the `$SPLUNK_HOME/etc/system/local/` directory, or in your own custom application directory in `$SPLUNK_HOME/etc/apps/`. You might need to create the file if it doesn't already exist.

This input stanza configures Splunk Enterprise to read from a FIFO queue at the specified path:

```
[fifo://<path>]
<setting1> = <val1>
<setting2> = <val2>
...
```

You can use the following settings with FIFO input stanzas:

Setting	Description	Default
<code>host = <string></code>	The host key or field to a static value for this stanza. The <code><string></code> is prepended with <code>host::</code> . This setting sets the host key's initial value. This key is used during parsing and indexing to set the host field. It also uses the host field at search time.	The IP address or fully qualified domain name of the host where the data originated.
<code>index = <string></code>	The index where events from this input are stored. The <code><string></code> is prepended with <code>index::</code> .	The <code>main</code> index or whatever you have set as your default index.
<code>sourcetype = <string></code>	The sourcetype key or field for events from this input. This setting explicitly declares the source type for this data, as opposed to letting it be determined automatically. Declaring the source type is important both for searchability and for applying the relevant formatting for this type of data during parsing and indexing.	Splunk software picks a source type based on various aspects of the data. There is no hard-coded default.

Setting	Description	Default
	<p>This setting sets the sourcetype key's initial value. This value is used during parsing and indexing to set the source type field. It is also the source type field used at search time.</p> <ul style="list-style-type: none"> • The <code><string></code> is prepended with <code>sourcetype::</code>. • For more information about source types, see Why source types matter. 	
<code>source = <string></code>	<p>Sets the source key or field for events from this input. The <code><string></code> is prepended with <code>source::</code>.</p> <p>Don't override the source field unless absolutely necessary. The input layer provides a more accurate string to aid in problem analysis and investigation, accurately recording the file from which the data was retrieved. Consider using source types, tagging, and search wildcards before overriding this value.</p>	The input file path.
<code>queue = [parsingQueue indexQueue]</code>	<p>Where the input processor deposits the events that it reads.</p> <p>Set to <code>parsingQueue</code> to apply the props.conf file and other parsing rules to your data. Set to <code>indexQueue</code> to send your data directly into the index.</p>	Defaults to <code>parsingQueue</code> .

Monitor changes to your file system

This feature is deprecated.
<p>This feature has been deprecated as of Splunk Enterprise version 5.0. This means that although it continues to function in the current version of the Splunk platform, it might be removed in a future version. As an alternative, you can:</p> <ul style="list-style-type: none"> • Learn how to monitor file system changes on Windows systems. • Use the auditd daemon on *nix systems and monitor output from the daemon. <p>For a list of all deprecated features, see the topic <i>Deprecated features</i> in the <i>Release Notes</i>.</p>

The Splunk platform **file system change monitor** tracks changes in your file system. The monitor watches a directory you specify and generates an event when that directory undergoes a change. It can detect when a file on the system is edited, deleted, or added. It detects changes on any file, including files that are not Splunk platform-specific files.

For example, you can configure the file system change monitor to watch the `/etc/sysconfig/` directory and alert you any time the system configurations change.

To learn how to monitor file system changes on Windows with built-in Microsoft auditing tools, see [Monitor file system changes](#).

The file system change monitor works with on-premises versions of the Splunk platform only. If you use Splunk Cloud Platform, you must use a universal or heavy forwarder to send file system change data to the Splunk Cloud Platform.

instance.

How the file system change monitor works

The file system change monitor detects changes on the *nix file system by using the following attributes:

- modification date/time
- group ID
- user ID
- file mode (read/write attributes, etc.)
- an optional Secure Hash Algorithm-256 (SHA256) hash of file contents

You can configure the following features of the file system change monitor:

- allow list using regular expressions
 - ◆ specify files that will be checked, no matter what
- deny list using regular expressions
 - ◆ specify files to skip
- directory recursion
 - ◆ including symbolic link traversal
 - ◆ scanning multiple directories, each with their own polling frequency
- cryptographic signing
 - ◆ creates a distributed audit trail of file system changes
- indexing entire file as an event on add/change
 - ◆ size cutoffs for sending entire file and/or hashing
- all change events indexed by, and searchable through, the Splunk platform

The file system change monitor generates **audit events** whenever any process changes, deletes, or adds to the contents of the `$SPLUNK_HOME/etc/` directory. When you start an on-premises Splunk instance for the first time, it generates an audit event for each file in the `$SPLUNK_HOME/etc/` directory and all its subdirectories. Afterward, any change in configuration, regardless of origin, generates an audit event for the affected file.

The file system change monitor sends data to various indexes depending on how you configure the file system monitoring input. If you have configured the `signedaudit` setting for the input, the instance sends the file system change to the **audit index**. If you have not configured `signedaudit`, then the instance writes the events to the **main** index, unless you specify another index.

The file system change monitor only tracks that a change has occurred. It does not track the user name of the account that made the change. For user-level monitoring, consider using built-in operating system audit tools, which have access to this information.

Do not configure the file system change monitor to monitor your root file system. Doing so can be very resource-intensive if you enable directory recursion.

Prerequisites to configuring the file system change monitor

- If you use Splunk Cloud Platform, you must configure a universal or heavy forwarder to connect to your Splunk Cloud Platform instance to send file system change data there.
- You must use configuration files to configure the file system change monitor.

Configure the file system change monitor

To use the file system change monitor to watch any directory, add or edit an `[fschange]` stanza to the `inputs.conf` configuration file in the `$SPLUNK_HOME/etc/system/local/` directory, or your own custom application directory in `$SPLUNK_HOME/etc/apps/` on the machine from which you want to collect file system change information. For information on configuration files in general, see [About configuration files in the Admin manual](#).

You must restart the Splunk platform any time you make changes to the `[fschange]` stanza in the `inputs.conf` file.

1. On the Splunk platform instance that is to send data to Splunk Cloud Platform, use a text editor to open the `inputs.conf` configuration file.
2. Add `[fschange:<directory>]` stanzas to specify files or directories that the Splunk platform should monitor for changes.
3. Save the `inputs.conf` file and close it.
4. Restart the Splunk platform instance. File system change monitoring begins immediately.

File system change monitor syntax

Here is the syntax for the `[fschange]` stanza:

```
[fschange:<directory or file to monitor>]
<setting1> = <val1>
<setting2> = <val2>
...
```

The Splunk platform monitors all adds, updates, and deletes to the directory and its subdirectories. Any change generates an event that the Splunk platform indexes. `<directory or file to monitor>` defaults to `$SPLUNK_HOME/etc/`.

Configuration file settings for the file system change monitor

All settings for the file system change monitor are optional. Here is the list of available settings:

Setting	Description	Default
<code>index=<indexname></code>	The index where the Splunk platform stores all events it generates.	main , unless you have turned on audit event signing, then _audit .
<code>recurse=<true false></code>	Whether or not to recurse all directories within the directory specified in <code>[fschange]</code> . Set to <code>true</code> to recurse all subdirectories and <code>false</code> to specify only the current directory.	<code>true</code>
<code>followLinks=<true false></code>	Whether or not the file system change monitor follows symbolic links. Set to <code>true</code> to follow symbolic links and <code>false</code> to not follow symbolic links.	<code>false</code> <div>File system loops can occur if you set <code>followLinks</code> on a directory that has symbolic links to files in the same directory.</div>

`pollPeriod=N` Check this directory for changes every N seconds. 3600 seconds

If you make a change to a file in the monitored directory, the file system audit events could take anywhere between 1 and 3600 seconds to be generated and become available in audit search.

`hashMaxSize=N` Calculate a SHA1 hash for every file that is less than or equal to N size in bytes.

This hash can be used as an additional method for detecting change in the file/directory.

-1, The splunk platform does not calculate a hash for change detection)
`signedaudit=<true | false>` Send cryptographically signed add, update, or delete events.

Set to true to generate events in the `_audit` index. Set to false if you're configuring the `index` setting. When setting `signedaudit` to true, confirm auditing is enabled in the `audit.conf` configuration file.

`false`
`fullEvent=<true | false>` Send the full event if an add or update change is detected.

Further qualified by the `sendEventMaxSize` attribute.

`false`
`sendEventMaxSize=N` Only send the full event if the size of the event is less than or equal to N bytes.

This limits the size of indexed file data.

-1 (unlimited)
`sourcetype = <string>` Set the source type for events from this input.

"sourcetype::" is prepended to `<string>`.

`audittrail` (if `signedaudit=true`) or `fs_notification` (if `signedaudit=false`)
`filesPerDelay = <integer>` Injects a delay specified by `delayInMills` after processing `<integer>` files. This setting throttles file system monitoring so it does not consume as much CPU.
`n/adelayInMills = <integer>` The delay, in milliseconds, to use after processing every `<integer>` file as specified in `filesPerDelay`. This setting is used to throttle file system monitoring so it does not consume as much CPU.
`filters=<filter1>,<filter2>,...<filterN>` Each of these filters applies from left to right for each file or directory that is found during the monitors poll cycle.
`n/a`

Define a filter for the file system change monitor

To define a filter to use with the `filters` attribute, add a `[filter...]` stanza as follows:

```
[filter:blacklist:backups]
regex1 = .*bak
regex2 = .*bk
[filter:whitelist:code]
regex1 = .*\.c
regex2 = .*\.h
```

```
[fschange:/etc]
filters = backups,code
```

The following list describes how the Splunk platform handles `fschange` allow list and deny list logic:

- The events go through the list of filters until they reach their first match.
- If the first filter to match an event is an allow list, then the Splunk platform indexes the event.
- If the first filter to match an event is a deny list, the filter prevents the event from getting indexed.
- If an event reaches the end of the chain with no matches, then the Splunk platform indexes the event.

To default to a situation where the Splunk platform does not index events if they don't match an allow list explicitly, end the chain with a deny list that matches all remaining events.

For example:

```
...
filters = <filter1>, <filter2>, ... terminal-blacklist
```

```
[filter:blacklist:terminal-blacklist]
regex1 = .?
```

If you exclude a directory including a terminal deny list at the end of a series of allow lists, then the Splunk platform excludes all its subfolders and files because they do not pass any allow list. To accommodate this scenario, include all desired folders and subfolders explicitly before the deny list items in your filters.

Example of explicit inclusion and terminal exclusion

This configuration monitors files in the specified directory with the extensions .config, .xml, .properties, and .log and ignores all others.

In this example, the file system change monitor could exclude a directory. If this is the case, the Splunk platform excludes all of its subfolders and files as well. Only files in the specified directory would be monitored.

```
[filter:whitelist:configs]
regex1 = .*\.config
regex2 = .*\.xml
regex3 = .*\.properties
regex4 = .*\.log

[filter:blacklist:terminal-denylist]
regex1 = .?
```

```
[fschange:/var/apache]
index = sample
recurse = true
followLinks = false
signedaudit = false
fullEvent = true
sendEventMaxSize = 1048576
delayInMills = 1000
filters = configs,terminal-denylist
```

Use the file system change monitor with a universal forwarder

To forward file system change monitor events from a universal forwarder, you must configure the `fschange` stanza with the `signedaudit = false` and `index=_audit` settings.

```
[fschange:<directory or file to monitor>]
signedaudit = false
index=_audit
```

With this workaround, the Splunk platform indexes file system change monitor events into the **_audit** index with a sourcetype of `fs_notification` and a source of `fschangemonitor`, instead of a source and sourcetype of `audittrail`.

Get data from APIs and other remote data interfaces through scripted inputs

Splunk Cloud Platform, through a universal or heavy forwarder, can accept events from scripts that you provide.

Scripted input is useful combined with some Windows and *nix command-line tools, such as `ipconfig`, `iostat`, `netstat`, `top`, and so on. You can use scripted input to get data from APIs, other remote data interfaces, and message queues. You can then use commands like `vmstat` and `iostat` on that data to generate metrics and status data. On Windows platforms, you can enable text-based scripts, such those in Perl and Python, with an intermediary Windows batch (.bat) or

PowerShell (.ps1) file.

You can configure scripted inputs from the Settings menu in Splunk Web on Splunk Enterprise, or by editing the `inputs.conf` configuration file on a universal or heavy forwarder.

When a scripted input launches a script, that script inherits the Splunk Enterprise or universal forwarder environment. The only environment variable that might cause problems with scripts and script output generation is the library path, most commonly known as `LD_LIBRARY_PATH` on Linux, Solaris, and FreeBSD. When you use scripted inputs on a Splunk platform instance, clear any environment variables that can affect the operation of a script.

Splunk Enterprise and the universal forwarder log any messages that scripted inputs send to the `stderr` I/O channel to `splunkd.log`.

Prerequisites

To add a scripted input, you must first write an input. To learn how to write scripted inputs, see Scripted input examples for Splunk Cloud Platform or Splunk Enterprise on the Splunk developer portal.

Add a scripted input in Splunk Web

On a Splunk Enterprise or heavy forward instance, follow these high-level steps to add a scripted input in Splunk Web:

1. Go to the Add New page.
2. Select the input source.
3. (Optional) Specify input settings.
4. Review your choices.

Go to the Add Data page

To get to the Add Data page using Settings, follow these steps:

1. In Splunk Web, click **Settings**.
2. Click **Data Inputs**.
3. Click **Scripts**.
4. Click **New** to add an input.

To get to the Add Data page using the Splunk Web homepage, follow these steps:

1. In Splunk Web, click the **Add Data**.
2. Click **Monitor** to monitor a script on the local machine, or click **Forward** to forward data from a script on a remote machine.
Splunk Web displays the **Add Data - Select Source** page.
3. In the left pane, select **Scripts**.

Forwarding data from scripted inputs requires additional setup.

Select the input source

1. In the **Script Path** drop-down list, select the path where the script resides.
Splunk Web updates the page to include the **Script Name** drop-down list.

2. In the **Script Name** drop-down list, select the script that you want to run.
Splunk Web updates the page to populate the **Command** field with the script name.
3. In the **Command** field, add any arguments needed to invoke the script.
4. In the **Interval** field, enter the amount of time, in seconds, that Splunk Enterprise waits before invoking the script.
5. (Optional) In the **Source Name Override** field, enter a new source name to override the default source value, if necessary.
6. Click **Next**.

Specify input settings

You can specify the application context, default host value, and index on the **Input Settings** page. All of these parameters are optional. For more about setting the host value, see [About hosts](#).

Setting the **Host** on this page sets the **host** field only in the resulting events. It does not direct Splunk Enterprise to look on a specific host on your network.

1. Select the source type for the script. Click **Select** to pick from the list of available source types on the local machine, or click **Manual** to enter the name of a source type.
2. Select the appropriate **Application context** for this input.
3. Set the **Host** name. You have several choices for this setting.
4. Set the **Index** that Splunk Enterprise will send data to. Unless you defined multiple indexes to handle different types of events, leave the value as **default**. In addition to indexes for user data, Splunk Enterprise has multiple utility indexes, which also appear in this drop-down list.
5. Click **Review**.

Review your choices

After specifying all your input settings, review your selections. Splunk Web lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.
2. If they do not match what you want, click the left angle bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

Splunk Web displays the Success page.

Add a scripted input with the inputs.conf configuration file

You add a scripted input in the inputs.conf file by adding a `[script]` stanza within that file. You can do this on Splunk Enterprise or the universal forwarder, and then forward that information to Splunk Cloud Platform.

Syntax

The syntax for the `[script]` stanza appears as follows, where `$SCRIPT` is the full path to the location of the script:

```
[script://$SCRIPT]
<attribute1> = <val1>
<attribute2> = <val2>
...
```

`$SCRIPT` can also be a file path that ends in the `.path` suffix. This special suffix lets you use the stanza to point to another command or script that exists anywhere on the host file system. See [Use the .path suffix to reference external scripts](#). The

file that you refer to in the stanza must follow the location restrictions described in the following section, Where to place the scripts for scripted inputs.

Where to place the scripts for scripted inputs

The script that you refer to in `$$SCRIPT` can reside in only one of the following places on the host file system:

- `$$SPLUNK_HOME/etc/system/bin`
- `$$SPLUNK_HOME/etc/apps/<your_app>/bin`
- `$$SPLUNK_HOME/bin/scripts`

As a best practice, put your script in the `bin/` directory that is nearest to the `inputs.conf` file that calls your script on the host file system. For example, if you configure `$$SPLUNK_HOME/etc/system/local/inputs.conf`, place your script in `$$SPLUNK_HOME/etc/system/bin/`. If you work on an application in `$$SPLUNK_HOME/etc/apps/$APPLICATION/`, put your script in `$$SPLUNK_HOME/etc/apps/$APPLICATION/bin/`.

Attributes

All attributes are optional. Here is the list of available attributes:

Attribute	Description	Default
<code>interval = <number> <cron schedule></code>	<p>How often to run the specified command. Specify either an integer value representing seconds or a valid cron schedule.</p> <p>When you specify a <code>cron schedule</code>, the script does not run at start-up, but rather at the times that the cron schedule defines.</p> <p>Splunk Enterprise keeps one invocation of a script per instance. Intervals are based on when the script completes. If you configure a script to run every 10 minutes and the script takes 20 minutes to complete, the next run occurs 30 minutes after the first run.</p> <p>For constant data streams, enter 1 or a value smaller than the script interval. For one-shot data streams, enter -1. Setting <code>interval</code> to -1 causes the script to run each time at start-up.</p>	60 seconds
<code>index = <string></code>	<p>The index where events from this input are stored. Splunk Enterprise prepends the <code><string></code> with <code>index::</code>.</p> <p>For more information about the index field, see How indexing works in the <i>Managing Indexers and Clusters of Indexers</i> manual.</p>	<code>main</code> , or whatever you have set as your default index.
<code>sourcetype = <string></code>	Sets the <code>sourcetype</code> field for events from this input. The <code><string></code> is prepended with <code>sourcetype::</code> .	There is no hard-coded default. Splunk Enterprise picks a source type based on various aspects of the data.

Attribute	Description	Default
	<p>Explicitly declares the source type for this data, as opposed to letting it be determined automatically. This is important both for searchability and for applying the relevant formatting for this type of data during parsing and indexing.</p> <p>Sets the <code>sourcetype</code> key initial value. Splunk Enterprise uses this key during parsing and indexing, particularly to set the <code>sourcetype</code> field during indexing. It also uses the <code>sourcetype</code> field at search time.</p>	
<code>source = <string></code>	<p>Sets the <code>source</code> key for events from this input.</p> <div> <p>Do not override the source key unless absolutely necessary. Typically, the input layer provides a more accurate string to aid in problem analysis and investigation, accurately recording the file from which the data was retrieved. Consider use of source types, tagging, and search wildcards before overriding this value.</p> </div>	

Splunk Enterprise prepends `<string>` with `source::`.

The input file `pathdisabled = <true | false>` Whether or not the input will run. Set to true if you want to disable the input.
`false`

Run scripts continuously

If you want the script to run continuously, write the script to never exit and set it on a short interval. This helps to ensure that if a problem occurs, the script restarts. Splunk Enterprise keeps track of scripts it spawned and shuts them down on exit.

Use a wrapper script

As a best practice, write a wrapper script for scripted inputs that use commands with arguments. In some cases, the command can contain special characters that the scripted input escapes when it validates text that you enter in Splunk Web. Updates to a previously configured input will then fail to save.

When validating text, Splunk Enterprise escapes characters that can't be in paths, such as the equal sign (=) and semicolon (;). For example, the following scripted input is not correctly saved when you edit it in Splunk Web because the scripted input escapes the equal (=) sign in the parameter to the `myUtil.py` utility:

```
[script://$SPLUNK_HOME/etc/apps/myApp/bin/myUtil.py file=my_data.csv]
disabled = false
```

To avoid this problem, write a wrapper script that contains the scripted input, or use the special `.path` argument for the scripted input stanza name. For information on writing wrapper scripts, see *Scripted inputs overview* in the *Developing Views and Apps for Splunk Web* manual.

When you update scripted inputs by editing `inputs.conf` directly, this validation does not occur.

Use the .path suffix to reference external scripts

As an alternative to writing a wrapper script, you can configure the scripted input to reference a script or executable that is anywhere on the host file system.

The script that you refer to can have a single line that calls the script or executable that you want. You can use this file to call a runtime environment that is outside of the Splunk Enterprise environment. For example, if you have both Splunk Enterprise, which comes with Python, and a second installation of Python on the same host, you can use the `.path` method to refer to the second Python installation.

Follow these steps to refer to external scripts with the `.path` suffix:

1. Use Splunk Web or edit `inputs.conf` and specify a scripted input stanza with a script name that ends in `.path`. For example:

```
[script://myfile.path]
disabled = 0
```

2. Place the file that you reference in the stanza in the appropriate directory, as described in [Where to place the scripts for scripted inputs](#).
3. Edit the file to specify the script or executable you want. For example:

```
/path/to/myscript -arg1 arg -arg2 arg
```

Examples of scripted inputs with inputs.conf

The following examples configure various scripted inputs with `inputs.conf`.

Unix top command

This example shows the use of the UNIX `top` command as a data input source:

1. Create a new application directory. This example uses `scripts/`.

```
$ mkdir $SPLUNK_HOME/etc/apps/scripts
```

2. Create a `bin/` directory. All scripts must be run out of a `bin/` directory inside your application directory.

```
$ mkdir $SPLUNK_HOME/etc/apps/scripts/bin
```

3. Create a script within the `bin/` directory. This example uses a small shell script `top.sh`.

```
$ #!/bin/sh
top -bn 1 # linux only - different OSes have different parameters
```

4. Make the script executable.

```
chmod +x $SPLUNK_HOME/etc/apps/scripts/bin/top.sh
```

5. Test that the script works by running it with the shell.

```
$SPLUNK_HOME/etc/apps/scripts/bin/top.sh
```

The script sends one `top` output.

6. Add the script entry to `inputs.conf` in `$SPLUNK_HOME/etc/apps/scripts/local/`.

```
[script:///opt/splunk/etc/apps/scripts/bin/top.sh]
interval = 5 # run every 5 seconds
sourcetype = top # set sourcetype to top
source = script:///bin/top.sh # set source to name of script
```

7. By default, Splunk Enterprise breaks the single `top` entry into multiple events, so you might need to modify `props.conf` to fix this issue. If necessary, edit `props.conf` and configure the server to break only before something

that doesn't exist in the output.

For example, adding the following to `$SPLUNK_HOME/etc/apps/scripts/default/props.conf` forces all lines into a single event:

```
[top]
BREAK_ONLY_BEFORE = <stuff>
```

Since there is no timestamp in the `top` output, you must tell Splunk Enterprise to use the current time. Set the following parameter in `props.conf`:

```
DATETIME_CONFIG = CURRENT
```

Reference an external script with the `.path` stanza

The following example uses the special `.path` stanza setting to reference an external build of Python to run a script on your host.

1. Edit `inputs.conf`.

```
[script://loglogs.path]
disabled = 0
```

2. Place or create `loglogs.path` in `$SPLUNK_HOME/etc/system/bin`.
3. Edit `loglogs.path` to reference the external version of Python.

```
/usr/bin/python logit.py --source /opt/files/my_files --target /opt/files/my_files/processed
--logfile /opt/src/my_sources/logfiles
```

Set interval attribute to cron schedule

In the previous example, you can also set the `interval` attribute to a cron schedule by specifying strings.

For example, the following string means the script runs once an hour at the top of the hour:

```
0 * * * *
```

The following string means the script runs every 15 minutes from 9 AM until 5 PM, Monday to Friday.

```
*/15 9-17 * * 1-5
```

The following string means the script runs at 15, 35, and 55 minutes after the hour between midnight and 7 AM and again between 8 PM and midnight, on the first of every even-numbered month, such as February, April, June, and so on.

```
15,35,55 0-6,20-23 1 */2 *
```

For more information about setting cron schedules, see `CRONTAB(5)` in <https://crontab.org> on the Crontab website.

Get data with the Journald input

The `journald` input is a modular input that collects logs that Linux `journald` system logging produces for:

- 64-bit Intel x86 (x86-64) chipsets
- Linux on ARM architecture machines.

The journald process runs on versions of the Linux operating system that use systemd as the system management service. The `systemd-journald.service` writes journal entries to the journald database, and they are read with the **journalctl** utility.

Your Splunk platform deployment accepts journald data from **universal forwarders** that capture the data and send it to other parts of your Splunk platform deployment. Both Splunk Cloud Platform and Splunk Enterprise accept journald data.

How the journald reader works

The journald input takes its configurations from defined stanzas in the `inputs.conf` file. Each defined stanza contains the settings that define what data your Splunk Universal Forwarder collects. The number of defined stanzas determine the number of **journalctl** processes that run. For example, if you define five stanzas, the **journalctl** utility runs five times. If you configure five identical defined stanzas, the **journalctl** utility reports identical data to the Splunk software five times.

Every setting you define in a journald stanza is a filter for your data. If you do not configure filters, journald ingests the full contents of the journal, starting with the oldest entry. To avoid this, you define filters to refine your data ingestion. Journald configuration supports both prescriptive and restrictive filters. If one or more `FIELD=VALUE` match arguments are passed, the output is filtered accordingly. For example, `_SYSTEMD_UNIT=httpd.service` refers to the components of a structured journal entry.

Version and field compatibility

The journald input collector uses the systemd utility `journalctl`. The older versions (versions 236 and lower) of `journalctl` do not support the `journalctl-include-fields` configuration parameter.

To determine your version of `journalctl`, enter the following line into your command line interface (CLI):

```
journalctl --version
```

Journalctl version	Supported fields
Versions 236 and higher	<code>journalctl-include-fields</code> , <code>journalctl-exclude-fields</code> , <code>journalctl-filter</code> , <code>journalctl-grep</code> , <code>journalctl-user-unit</code>
Versions 236 and lower.	<code>journalctl-exclude-fields</code> , <code>journalctl-filter</code> , <code>journalctl-grep</code> , <code>journalctl-user-unit</code>

Configure the journald input

Use the `inputs.conf` file to configure your input.

1. On your Splunk Universal Forwarder, navigate to `splunkforwarder/etc/apps/journald_input/default/`.
2. Copy the `inputs.conf` file.
3. Navigate to `splunkforwarder/etc/apps/journald_input/local/`.
4. Paste the copy of the `inputs.conf` file.
5. Open the `inputs.conf` file with a text editor.
6. Navigate to the `journald` stanza, and configure the stanza with the parameters that include or exclude the data fields that you want.

```
[journald://my-stanza]
# The following fields are included by default, and required: MESSAGE, _REALTIME_TIMESTAMP, _CURSOR
journalctl-include-fields = PRIORITY,CMD,EXE
```

```
journalctl-exclude-fields =
journalctl-filter = _SYSTEMD_UNIT=my.service,_PID=232+_SYSTEMD_UNIT=sshd
journalctl-grep =^WARN.*disk,.*errno=\d+\S+restarting
journalctl-user-unit =unit1,unit2
```

For a full list of parameters, see the **Reference** section in this topic.

7. Verify that your field configurations begin with underscores. The journald input stanza configuration requires underscores
8. Save your changes.
9. Restart your Splunk Universal Forwarder.
10. (Optional) Use a deployment server to push the changes to your settings to other forwarders in your Splunk platform deployment. For more information, see [Use forwarder management to manage apps](#) topic in the *Updating Splunk Enterprise Instances* manual.

Reference

For a full list of available parameters, see the following table:

Parameter	Description
journalctl-include-fields = <string>	Filter which fields to send to the Splunk platform instance.
journalctl-exclude-fields = <string>	Restrict the fields that sent to the Splunk platform instance.
journalctl-filter = <string>	Filters fields using the matches concept in journalctl. For example, <code>_SYSTEMD_UNIT=avahi-daemon.service _PID=28097 + _SYSTEMD_UNIT=dbus.service</code> will show all messages from the Avahi service process with a PID of 28097, plus all messages from the D-Bus service.
journalctl-unit = <string>	Show messages for the specified systemd unit.
journalctl-identifier = <string>	Show messages for the specified syslog identifier <code>SYSLOG_IDENTIFIER</code> .
journalctl-priority = <string>	Filter output by message priorities or priority ranges.
journalctl-boot = <string>	Messages from a specific boot.
journalctl-facility = <string>	Filter output by syslog facility.
journalctl-grep = <string>	Filter output to entries where the <code>MESSAGE=</code> field matches the specified regular expression.
journalctl-user-unit = <string>	Show messages for the specified user session unit.
journalctl-dmesg = <string>	Show only kernel messages.
journalctl-quiet = <string>	Suppresses all informational messages.
journalctl-freetext = <string>	Reserved for future use.

Troubleshoot the journald input

Best practices for the journald input

- Check for errors in your `splunkd.log` file.
- Start with a simple configuration before you build something more complex.
- For more information on configurations, see the spec file for this product.

Verify data

To verify the data that your journald input is collecting, enter the following into your command line interface (CLI):

```
ps aux | grep journalctl.
```

You need to define at least one stanza in your `inputs.conf` for data to be collected.

Data duplication

When configuring journald, Each stanza will run a journal reader. Duplicate stanza definitions will result in data duplication.

Data collection efficiency

The `include-fields` is more efficient than using `exclude-fields`, `include-fields` filtering happens at the journald level, instead of the Splunk Universal Forwarder level.

Fields are not available

Review your `journalctl` version; the `include-fields` and `grep` are not available on earlier versions.

For deployment server scenarios where some machines use a newer version of `journalctl` while some machines use the older version, the Splunk deployment server will push the same configurations to the new and older machines, and the older machines will fail to collect data.

Troubleshoot missing field options

The journald input uses an externally-developed utility to function. Because of this, older versions of `journalctl` might not have all configuration options available. The following options might be missing in CentOS 7 filtering configurations:

- The `journalctl-include-fields` option lets you declare which journald fields you want to retrieve. Without this option, you can only specify which fields you do not want to include. If, for example, you want to retrieve only the `MESSAGE` field, you must list every field you do not want your Splunk platform instance to ingest. This method can generate multiple lengthy data fields.
- The `grep` option lets you apply a regular expression filter before data reaches the indexer.

journald logs unable to open cursor file for the first start on the Splunk Universal Forwarder

If, when starting the journald modular input for the first time on the universal forwarder, you receive the following journald log error:

```
09-04-2020 03:50:49.221 +0000 ERROR ExecProcessor - message from
"/home/ec2-user/splunkforwarder/bin/splunkd journald-modinput '$@'" journald-modinput - unable to open
cursor file /home/ec2-user/splunkforwarder/var/lib/splunk/modinputs/journald/s1.checkpoint for reading (No
such file or directory)
09-04-2020 03:50:49.221 +0000 INFO ExecProcessor - message from
"/home/ec2-user/splunkforwarder/bin/splunkd journald-modinput '$@'" journald-modinput - starting (mode =
Data Collection) Config - serverHost=ip-172-31-49-163.us-west-2.compute.internal
serverUri=https://127.0.0.1:8089
checkpointDir=/home/ec2-user/splunkforwarder/var/lib/splunk/modinputs/journald stanza=journald://s1
PropertiesMap: {host -> '$decideOnStartup' index -> 'default' interval -> '30' journalctl-exclude-fields ->
'__MONOTONIC_TIMESTAMP,__SOURCE_REALTIME_TIMESTAMP' journalctl-include-fields ->
'PRIORITY,__SYSTEMD_UNIT,__SYSTEMD_CGROUP,__TRANSPORT,__PID,__UID,__MACHINE_ID,__GID,__COMM,__EXE' journalctl-quiet
-> 'true'}
```

The cursor file might not be available the first time you run journald. This is where the Splunk software keeps track of the data that has been ingested from the journal. This does not affect journald input functionality. The data injection is normal and generates this error log response.

Journald ID is missing checkpoint file error

If you run the journald input for the first time and receive the following error:

```
10-03-2020 18:29:35.241 -0600 ERROR ExecProcessor - message from "/opt/splunkforwarder/bin/splunkd
journald-modinput '$@'" journald-modinput - unable to open cursor file
/opt/splunkforwarder/var/lib/splunk/modinputs/journald/all.checkpoint for reading (No such file or
directory)
```

When the journald reader starts, it first checks whether the Splunk software needs to resume reading after a previous stop. The first time this process runs, no checkpoint file exists, so the journalid reader will fail to find the checkpoint file, and will write the error to the log.

No additional actions need to be taken, the Splunk software restarts after the ID in this error.

Configure event processing

Overview of event processing

The Splunk platform indexes events, which are records of activity that reside in machine data. Events provide information about the systems that produce the machine data. The term **event data** refers to the contents of a Splunk platform index.

Here is a sample event:

```
172.26.34.223 - - [01/Jul/2017:12:05:27 -0700] "GET /trade/app?action=logout HTTP/1.1" 200 2953
```

When Splunk software indexes events, it does the following tasks:

Task	Link
Configures character set encoding	Configure character set encoding
Configures line breaking for multi-line events	Configure event line breaking
Identifies event timestamps and applies timestamps to events if they don't exist	Configure event timestamps
Extracts a set of useful standard fields, such as <code>host</code> , <code>source</code> , and <code>sourcetype</code>	About default fields
Segments events	About event segmentation
Dynamically assigns metadata to events, if specified	Assign default fields dynamically
Anonymizes data, if specified	Anonymize data

For an overview of the indexing process, see the Indexing overview chapter of the *Managing Indexers and Clusters of Indexers* manual.

Configure character set encoding

You can configure **character set encoding** for your data sources. Splunk software has built-in character set specifications to support internationalization of your **deployment**. Splunk software supports many languages, including some that don't use Universal Coded Character Set Transformation Format - 8-bit (UTF-8) encoding.

Splunk software attempts to apply UTF-8 encoding to your sources by default. If a source doesn't use UTF-8 encoding or is a non-ASCII file, Splunk software tries to convert data from the source to UTF-8 encoding unless you specify a character set to use by setting the `CHARSET` key in the `props.conf` file.

You can retrieve a list of the valid character encoding specifications by using the `iconv -l` command on most *nix systems. A port for `iconv` on Windows is available.

Supported character sets

Splunk software supports a wide range of character sets, including the following key character sets:

- UTF-8

- UTF-16LE
- Latin-1
- BIG5
- SHIFT-JIS`

The following table shows a short list of common supported character sets and the languages they correspond to.

Language	Code
Arabic	CP1256
Arabic	ISO-8859-6
Armenian	ARMSCII-8
Belarus	CP1251
Bulgarian	ISO-8859-5
Czech	ISO-8859-2
Georgian	Georgian-Academy
Greek	ISO-8859-7
Hebrew	ISO-8859-8
Japanese	EUC-JP
Japanese	SHIFT-JIS
Korean	EUC-KR
Russian	CP1251
Russian	ISO-8859-5
Russian	KOI8-R
Slovak	CP1250
Slovenian	ISO-8859-2
Thai	TIS-620
Ukrainian	KOI8-U
Vietnamese	VISCII

For more supported character sets, see the Comprehensive list of supported character sets section later in this topic.

Manually specify a character set

To manually specify a character set, you need to edit the props.conf file. If you have a Splunk Cloud Platform deployment, edit your props.conf file on your forwarder. If you have a Splunk Enterprise deployment, you can edit this file in your Splunk Enterprise deployment.

To manually specify a character set to apply to an input, set the `CHARSET` key in the props.conf file:

```
[spec]
CHARSET=<string>
```

For example, if you have a host that generates data in Greek and uses ISO-8859-7 encoding, set `CHARSET=ISO-8859-7` for

that host in the props.conf file. The host is called "GreekSource" in this example:

```
[host::GreekSource]
CHARSET=ISO-8859-7
```

Splunk software parses only character encodings that have UTF-8 mappings.

Automatically specify a character set

Splunk software can automatically detect languages and proper character sets using its character set encoding algorithm.

To configure Splunk software to automatically detect the proper language and character set encoding for a particular input, set `CHARSET=AUTO` for the input in the props.conf file. If you have a Splunk Cloud Platform deployment, you can edit this file on your forwarder. If you have a Splunk Enterprise deployment, you can edit this file in your Splunk Enterprise deployment.

For example, to automatically detect character set encoding for the host "my-foreign-docs", set `CHARSET=AUTO` for that host in the props.conf file:

```
[host::my-foreign-docs]
CHARSET=AUTO
```

Train Splunk Enterprise to recognize a character set

If you have Splunk Cloud Platform and want to add a character set encoding to your Splunk deployment, file a Splunk Support ticket. If you have a Splunk Enterprise deployment, you can train Splunk software to recognize the character set.

You can train Splunk Enterprise to recognize the character set by adding a sample file to the following path and restarting Splunk Enterprise:

```
$SPLUNK_HOME/etc/ngram-models/_<language>-<encoding>.txt
```

For example, if you want to use the "vulcan-ISO-12345" character set, copy the specification file to the following path:

```
/SPLUNK_HOME/etc/ngram-models/_vulcan-ISO-12345.txt
```

After the sample file is added to the specified path, Splunk software recognizes sources that use the new character set and automatically converts them to UTF-8 format at index time.

Comprehensive list of supported character sets

The common character sets described earlier in the Supported character sets section are a small subset of what the `CHARSET` attribute can support. Splunk software also supports a long list of character sets. Of the character sets that the Splunk platform supports, it also supports their aliases. identical to the list supported by the *nix `iconv` utility.

Splunk software ignores punctuation and case when matching `CHARSET`. For example, utf-8, UTF-8, and utf8 are all considered identical.

The following list shows all supported character sets with their aliases indicated in parentheses:

- utf-8 (CESU-8, ANSI_X3.4-1968, ANSI_X3.4-1986, ASCII, CP367, IBM367, ISO-IR-6, ISO646-US ISO_646.IRV:1991, US, US-ASCII, CSASCII)
- utf-16le (UCS-2LE, UNICODELITTLE)
- utf-16be (ISO-10646-UCS-2, UCS-2, CSUNICODE, UCS-2BE, UNICODE-1-1, UNICODEBIG, CSUNICODE11, UTF-16)
- utf-32le (UCS-4LE)
- utf-32be (ISO-10646-UCS-4, UCS-4, CSUCS4, UCS-4BE, UTF-32)
- utf-7 (UNICODE-1-1-UTF-7, CSUNICODE11UTF7)
- c99 (java)
- utf-ebcdic
- latin-1 (CP819, IBM819, ISO-8859-1, ISO-IR-100, ISO_8859-1:1987, L1, CSISOLATIN1)
- latin-2 (ISO-8859-2, ISO-IR-101, ISO_8859-2:1987, L2, CSISOLATIN2)
- latin-3 (ISO-8859-3, ISO-IR-109, ISO_8859-3:1988, L3, CSISOLATIN3)
- latin-4 (ISO-8859-4, ISO-IR-110, ISO_8859-4:1988, L4, CSISOLATIN4)
- latin-5 (ISO-8859-9, ISO-IR-148, ISO_8859-9:1989, L5, CSISOLATIN5)
- latin-6 (ISO-8859-10, ISO-IR-157, ISO_8859-10:1992, L6, CSISOLATIN6)
- latin-7 (ISO-8859-13, ISO-IR-179, L7)
- latin-8 (ISO-8859-14, ISO-CELTIC, ISO-IR-199, ISO_8859-14:1998, L8)
- latin-9 (ISO-8859-15, ISO-IR-203, ISO_8859-15:1998)
- latin-10 (ISO-8859-16, ISO-IR-226, ISO_8859-16:2001, L10, LATIN10)
- ISO-8859-5 (CYRILLIC, ISO-IR-144, ISO_8859-5:1988, CSISOLATINCYRILLIC)
- ISO-8859-6 (ARABIC, ASMO-708, ECMA-114, ISO-IR-127, ISO_8859-6:1987, CSISOLATINARABIC, MACARABIC)
- ISO-8859-7 (ECMA-118, ELOT_928, GREEK, GREEK8, ISO-IR-126, ISO_8859-7:1987, ISO_8859-7:2003, CSISOLATINGREEK)
- ISO-8859-8 (HEBREW, ISO-8859-8, ISO-IR-138, ISO8859-8, ISO_8859-8:1988, CSISOLATINHEBREW)
- ISO-8859-11
- roman-8 (HP-ROMAN8, R8, CSHPROMAN8)
- KOI8-R (CSKOI8R)
- KOI8-U
- KOI8-T
- GEORGIAN-ACADEMY
- GEORGIAN-PS
- ARMSII-8
- MACINTOSH (MAC, MACROMAN, CSMACINTOSH)

These MAC* character sets are for MacOS 9. Higher versions, like macOS X, use unicode.

- MACGREEK
- MACCYRILLIC
- MACUKRAINE
- MACCENTRALEUROPE
- MACTURKISH
- MACCROATIAN
- MACICELAND
- MACROMANIA
- MACHEBREW
- MACTHAI
- NEXTSTEP
- CP850 (850, IBM850, CSPC850MULTILINGUAL)
- CP862 (862, IBM862, CSPC862LATINHEBREW)
- CP866 (866, IBM866, CSIBM866)
- CP874 (WINDOWS-874)

- CP932
- CP936 (MS936, WINDOWS-936)
- CP949 (UHC)
- CP950
- CP1250 (MS-EE, WINDOWS-1250)
- CP1251 (MS-CYRL, WINDOWS-1251)
- CP1252 (MS-ANSI, WINDOWS-1252)
- CP1253 (MS-GREEK, WINDOWS-1253)
- CP1254 (MS-TURK, WINDOWS-1254)
- CP1255 (MS-HEBR, WINDOWS-1255)
- CP1256 (MS-ARAB, WINDOWS-1256)
- CP1257 (WINBALTRIM, WINDOWS-1257)
- CP1258 (WINDOWS-1258)
- CP1361 (JOHAB)
- BIG-5 (BIG-FIVE, CN-BIG5, CSBIG5)
- BIG5-HKSCS(BIG5-HKSCS:2001)
- CN-GB (EUC-CN, EUCCN, GB2312, CSGB2312)
- EUC-JP (EXTENDED_UNIX_CODE_PACKED_FORMAT_FOR_JAPANESE, CSEUCPKDFMTJAPANESE)
- EUC-KR (CSEUCKR)
- EUC-TW (CSEUCTW)
- GB18030
- GBK
- GB_1988-80 (ISO-IR-57, ISO646-CN, CSISO57GB1988, CN)
- HZ (HZ-GB-2312)
- GB_2312-80 (CHINESE, ISO-IR-58, CSISO58GB231280)
- SHIFT-JIS (MS_KANJI, SJIS, CSSHIFTJIS)
- ISO-IR-87 (JIS0208 JIS_C6226-1983, JIS_X0208 JIS_X0208-1983, JIS_X0208-1990, X0208, CSISO87JISX0208, ISO-IR-159, JIS_X0212, JIS_X0212-1990, JIS_X0212.1990-0, X0212, CSISO159JISX02121990)
- ISO-IR-14 (ISO646-JP, JIS_C6220-1969-RO, JP, CSISO14JISC6220RO)
- JISX0201-1976 (JIS_X0201, X0201, CSHALFWIDTHKATAKANA)
- ISO-IR-149 (KOREAN, KSC_5601, KS_C_5601-1987, KS_C_5601-1989, CSKSC56011987)
- VISCII (VISCII1.1-1, CSVISCII)
- ISO-IR-166 (TIS-620, TIS620-0, TIS620.2529-1, TIS620.2533-0, TIS620.2533-1)
- UCS-2-INTERNAL, UCS-2-SWAPPED, UCS-4-INTERNAL, UCS-4-SWAPPED

Configure event line breaking

Some events consist of more than one line. The Splunk platform handles most multiline events correctly by default. If you have multiline events that the Splunk platform doesn't handle properly, you can configure it to change its line breaking behavior.

If you use Splunk Cloud Platform, you must forward any data where you need to configure event-line breaking, because there is no way to configure event-line breaking in the Splunk Web interface. You must use a heavy forwarder that you have configured to send data to your Splunk Cloud Platform instance to break incoming data into lines and subsequently merge them as you want into events.

If you use Splunk Enterprise, you can configure the settings and follow the procedures in this topic on any instance that indexes the incoming data stream.

How the Splunk platform determines event boundaries

The Splunk platform determines event boundaries in two phases:

1. Line breaking, which uses the `LINE_BREAKER` setting to split the incoming stream of data into separate lines. By default, the `LINE_BREAKER` value is any sequence of newlines and carriage returns. In regular expression format, this is represented as the following string: `([\r\n]+)`. You don't normally need to adjust this setting, but in cases where it's necessary, you must configure it in the `props.conf` configuration file on the forwarder that sends the data to Splunk Cloud Platform or a Splunk Enterprise indexer. The `LINE_BREAKER` setting expects a value in regular expression format.
2. Line merging, which uses the `SHOULD_LINEMERGE` setting to merge previously separated lines into events. By default, the Splunk platform performs line merging, and the value for `SHOULD_LINEMERGE` is `true`. You don't normally need to adjust this setting, but in cases where it is necessary, you must configure this setting in the `props.conf` configuration file on the forwarder that sends the data to Splunk Cloud Platform. If you configure the Splunk platform to not perform line merging by setting the `SHOULD_LINEMERGE` attribute to `false`, then the platform splits the incoming data into lines according to what the `LINE_BREAKER` setting determines.

Line breaking is relatively efficient for the Splunk platform, while line merging is relatively slow. Using the `LINE_BREAKER` setting can produce the results you want in the line breaking phase. This is valuable if a significant amount of your data consists of multiline events.

There are additional configuration settings that help you break your incoming data stream into events, such as line-breaking.

How to configure event boundaries

Many event logs have a strict one-line-per-event format, but others don't. The Splunk platform can often recognize the event boundaries, but if event boundary recognition doesn't occur, or happens incorrectly, you can set custom rules in the `props.conf` configuration file to establish event boundaries.

Requirements for configuring event boundaries

Before you attempt to configure event boundaries for your events, confirm that you have the following:

- An understanding of regular expressions. The `LINE_BREAKER` setting uses a regular expression to determine what the boundary of an event is.
- One of the following, depending on whether you use Splunk Cloud Platform or Splunk Enterprise:
 - ◆ A heavy forwarder that has been configured to send data to your Splunk Cloud Platform instance. You can download the Splunk Cloud Platform universal forwarder credentials package that comes with your Splunk Cloud Platform instance and install it on a Splunk heavy forwarder.
 - ◆ A Splunk Enterprise indexer or heavy forwarder, if you use Splunk Enterprise.
- A file that represents the data stream where you want to configure custom line breaking.

Edit the props.conf configuration file to configure multiline events

1. Examine the file that you want to index to determine its event format.
2. In the file, look for a pattern in the events to set as the start or end of an event.
3. Using a text editor, on the forwarder you have configured to send data to Splunk Cloud Platform, edit the `$SPLUNK_HOME/etc/system/local/props.conf` configuration file.
4. In the `props.conf` configuration file, add the necessary line breaking and line merging settings to configure the forwarder to perform the correct line breaking on your incoming data stream.

5. Save the file and close it.
6. Restart the forwarder to commit the changes.

There are two ways to handle multiline events:

- Break and reassemble the data stream into events.
- Break the data stream directly into real events with the `LINE_BREAKER` setting.

Break and reassemble the data stream into events

This method oftentimes simplifies the configuration process, as it gives you access to several settings that you can use to define line-merging rules.

You must perform these steps on the heavy forwarder that you have designated to send data to your Splunk Cloud Platform instance.

1. On the forwarder that is to send data to your Splunk Cloud Platform instance, use a text editor to open `$SPLUNK_HOME/etc/system/local/props.conf` for editing.
2. In this file, specify a stanza in the props.conf configuration file that represents the stream of data you want to break and reassemble into events.
3. In that stanza, configure the `LINE_BREAKER` setting with a regular expression that breaks the data stream into multiple lines.
4. Add the `SHOULD_LINEMERGE` setting to the stanza, and set its value to `true`.
5. Configure additional line-merging settings, such as `BREAK_ONLY_BEFORE` and others, to specify how the forwarder is to reassemble the lines into events. For more information on the line-merging settings, see [Attributes that apply only when the SHOULD_LINEMERGE setting is true](#) later in this topic.

If your data conforms well to the default `LINE_BREAKER` value, which is any number of newlines and carriage returns, you don't need to change the `LINE_BREAKER` setting. Instead, set `SHOULD_LINEMERGE=true` and use the line-merging settings to reassemble the data.

Break the data stream directly into real events with the LINE_BREAKER setting

Using the `LINE_BREAKER` setting to define event boundaries might increase your indexing speed, but is somewhat more difficult to work with. If you find that indexing is slow and a significant amount of your data consists of multiline events, this method can provide significant improvement.

1. Specify a stanza in props.conf that represents the stream of data you want to break directly into events.
2. Under this stanza, configure the `LINE_BREAKER` setting with a regular expression that matches the boundary that you want to use to break up the raw data stream into events.
3. Add the `SHOULD_LINEMERGE` setting, and configure it to `false`.

Line breaking general settings

The following tables list the settings in the props.conf file that affect line breaking.

Attribute	Description	Default
<code>TRUNCATE = <non-negative integer></code>	Changes the default maximum line length, in bytes. Although this setting is a byte measurement, the Splunk platform rounds down line length when this attribute	10000

Attribute	Description	Default
	<p>would otherwise land mid-character for multibyte characters.</p> <p>Set to 0 if you never want truncation. However, very long lines are often a sign of garbage data.</p>	
<code>LINE_BREAKER = <regular expression></code>	<p>A regular expression that determines how the Splunk platform breaks the raw text stream into initial events, before any line merging takes place. This setting is dependent upon the <code>SHOULD_LINEMERGE</code> setting, described later.</p> <p>The expression must contain a capturing group, which is a pair of parentheses that defines an identified subcomponent of the match.</p> <p>Wherever the expression matches, the Splunk platform considers the start of the first capturing group to be the end of the previous event, and considers the end of the first capturing group to be the start of the next event.</p> <p>The platform discards the contents of the first capturing group. This content will not be present in any event, as the platform considers this text to come between lines.</p> <p>You can realize a significant boost to processing speed when you use the <code>LINE_BREAKER</code> setting to delimit multiline events as opposed to using <code>SHOULD_LINEMERGE</code> to reassemble individual lines into multiline events. Consider using this method if a significant portion of your data consists of multiline events.</p> <p>See the <code>props.conf</code> specification file for information on how to use <code>LINE_BREAKER</code> with branched expressions and additional information.</p>	<code>([\\r\\n]+)</code> The Splunk platform breaks data into an event for each line, delimited by any number of carriage return (<code>\\r</code>) or newline (<code>\\n</code>) characters.
<code>LINE_BREAKER_LOOKBEHIND = <integer></code>	<p>When there is leftover data from a previous raw chunk, <code>LINE_BREAKER_LOOKBEHIND</code> indicates the number of characters before the end of the raw chunk, with the next chunk concatenated, where the Splunk platform applies the <code>LINE_BREAKER</code> regular expression. You might want to increase this value from its default if you are dealing with especially large or multiline events.</p>	100
<code>SHOULD_LINEMERGE = [true false]</code>	<p>When set to <code>true</code>, the Splunk platform combines several input lines into a single event, with configuration based on the settings described in the next section.</p>	true

Attributes that apply only when the `SHOULD_LINEMERGE` setting is true

When you set `SHOULD_LINEMERGE` to the default of `true`, use these additional settings to define line breaking behavior.

Attribute	Description	Default
-----------	-------------	---------

BREAK_ONLY_BEFORE_DATE = [true false]	When set to true, the Splunk platform creates a new event if it encounters a new line with a date.	true If you configure the DATETIME_CONFIG setting to CURRENT or NONE, this attribute is not meaningful, because in those cases, the Splunk platform doesn't identify timestamps.
---------------------------------------	--	---

BREAK_ONLY_BEFORE = <regular expression>When set, the Splunk platform creates a new event if it encounters a new line that matches the regular expression.empty stringMUST_BREAK_AFTER = <regular expression>When set, and the regular expression matches the current line, the Splunk platform always creates a new event for the next input line. The platform might still break before the current line if another rule matches.empty stringMUST_NOT_BREAK_AFTER = <regular expression>When set, and the current line matches the regular expression, the Splunk platform doesn't break on any subsequent lines until the MUST_BREAK_AFTER expression matches.empty stringMUST_NOT_BREAK_BEFORE = <regular expression>When set and the current line matches the regular expression, the Splunk platform doesn't break the last event before the current line.empty stringMAX_EVENTS = <integer>Specifies the maximum number of input lines that the Splunk platform adds to any event. The software breaks the event after it reads the specified number of lines.256 lines

Examples of configuring event line breaking

Specify event breaks

The following example configures the Splunk platform to identify any line that consists of only digits as the start of a new event for any data whose source type is set to my_custom_sourcetype.

```
[my_custom_sourcetype]
BREAK_ONLY_BEFORE = ^\d+\s*$
```

Merge multiple lines into a single event

The following log event contains several lines that are part of the same request. The differentiator between requests is "Path".

```
{{"2006-09-21, 02:57:11.58", 122, 11, "Path=/LoginUser
Query=CrmId=ClientABC&ContentItemId=TotalAccess&SessionId=3A1785URH117BEA&Ticket=646A1DA4STF896EE&
amp;SessionTime=25368&ReturnUrl=http://www.clientabc.com, Method=GET, IP=209.51.249.195, Content=", ""}}
```

```
{{"2006-09-21, 02:57:11.60", 122, 15, "UserData:<User CrmId="clientabc"
UserId="p12345678"><EntitlementList></EntitlementList></User>", ""}}
{{"2006-09-21, 02:57:11.60", 122, 15, "New Cookie:
SessionId=3A1785URH117BEA&Ticket=646A1DA4STF896EE&CrmId=clientabc&UserId=p12345678&AccountId=&
amp;AgentHost=man&AgentId=man, MANUser:
Version=1&Name=&Debit=&Credit=&AccessTime=&BillDay=&Status=&Language=&Country=
&Email=&EmailNotify=&Pin=&PinPayment=&PinAmount=&PinPG=&PinPGRate=&PinMenu=&", ""}}
```

To index this multiline event properly, use the Path differentiator in your configuration. Add the following to your \$SPLUNK_HOME/etc/system/local/props.conf file.

```
[source::source-to-break]
SHOULD_LINEMERGE = True
BREAK_ONLY_BEFORE = Path=
```

This code configures the Splunk platform to merge the lines of the event, and only break before the term Path=.

Multiline event line breaking and segmentation limitations

The Splunk platform applies line breaking and segmentation limitations to extremely large events:

Limitation	Description
Events over <code>MAX_EVENTS</code> lines	If the platform encounters a multiline event that exceeds the number of lines that you specified in <code>MAX_EVENTS</code> , it breaks the event at that limit, sets the <code>BREAK_ONLY_BEFORE_DATE</code> setting to <code>false</code> if it is true, and then drops any <code>MUST_NOT_BREAK_BEFORE</code> or <code>MUST_NOT_BREAK_AFTER</code> rules. This can result in events not being line broken as you would expect. To work around the problem, you can raise the <code>MAX_EVENTS</code> setting, but you might get better results by changing the <code>SHOULD_LINEMERGE</code> setting to <code>false</code> and by specifying the event boundary with the <code>LINE_BREAKER</code> setting.
Lines that exceed 10,000 bytes in length.	The Splunk platform uses the <code>LINE_BREAKER</code> and <code>TRUNCATE</code> settings to evaluate and break events over 10kB into multiple lines of 10kB each. It appends the <code>meta::truncated</code> field to the end of any truncated line. If you have also configured <code>SHOULD_LINEMERGE</code> to <code>true</code> , the platform evaluates any additional event data using the <code>props.conf</code> rules until it can create a complete event.
Segmentation for events over 100,000 bytes	In search results, Splunk Web displays the first 100,000 bytes of an event. Segments after those first 100,000 bytes of a very long line are still searchable, however.
Segmentation for events over 1,000 segments	In search results, Splunk Web displays the first 1,000 segments of an event as segments separated by whitespace and highlighted on mouseover. It displays the rest of the event as raw text without interactive formatting.

Configure event timestamps

You can control how the Splunk platform handles timestamps in both Splunk Cloud Platform and Splunk Enterprise.

Examine this event example:

```
172.26.34.223 - - [01/Jul/2017:12:05:27 -0700] "GET /trade/app?action=logout HTTP/1.1" 200 2953
```

The time information in the event, `[01/Jul/2017:12:05:27 -0700]`, is a **timestamp**.

The Splunk platform uses timestamps to correlate events by time, create the histogram in Splunk Web, and set time ranges for searches. Most events contain timestamps, and in cases where an event doesn't have timestamp information, the Splunk platform attempts to assign a timestamp value to the event at index time.

In most cases, the Splunk platform extracts timestamps correctly, but there are situations where you might need to configure timestamp handling yourself. For example, when dealing with distributed deployments, you might need to reconfigure timestamp recognition and formatting.

See the [Configure timestamps](#) chapter of this manual for specific instructions on how to configure timestamps.

Configure indexed field extraction

Splunk software extracts various fields at index time. You can configure and modify how the software performs this field extraction.

Splunk software can extract the following fields at index time:

- Default fields
- Custom fields

- File header fields

Splunk software always extracts a set of default fields for each event. You can configure it to extract custom fields and, for some data, file header fields.

For more information on indexed field extraction, see the [Configure indexed field extraction](#) chapter.

Anonymize data

You might need to anonymize, or mask, sensitive personal information from the data that you index into Splunk Cloud Platform, such as credit card or Social Security numbers. You can anonymize parts of confidential fields in events to protect privacy while providing enough remaining data for use in event tracking.

To anonymize data with Splunk Cloud Platform, you must configure a Splunk Enterprise instance as a heavy forwarder and anonymize the incoming data with that instance before sending it to Splunk Cloud Platform.

There are two ways to anonymize data with a heavy forwarder:

- **Use the `SEDCMD` setting.** This setting exists in the `props.conf` configuration file, which you configure on the heavy forwarder. It acts like a `sed` *nix script to do replacements and substitutions. This method is more straightforward, takes less time to configure, and is slightly faster than a regular expression transform. But there are limits to how many times you can invoke the `SEDCMD` setting and what it can do. For instructions on this method, see [Anonymize data with a sed script](#).
- **Use a regular expression (regex) transform.** This method takes longer to configure, but less complex to modify after the initial configuration. You can also assign this method to multiple data inputs more flexibly. For instructions on this method, see [Anonymize data with a regular expression transform](#).

Both of these options are also available in Splunk Enterprise, where you can complete the configuration on either a heavy forwarder or an indexer.

Prerequisites to anonymize data

Before you can anonymize data, you must select a set of events to anonymize.

- First, you select the events to anonymize
- Then, you either:
 - ♦ Use the `props.conf` configuration file to anonymize the events with a sed script
 - ♦ Use the `props.conf` and `transforms.conf` configuration files to anonymize the events with a regular expression transform

Select events to anonymize

You can anonymize event data based on whether the data comes from a specific source or host, or whether the data is tagged with a specific source type. You must specify which method to select the data in the `props.conf` configuration file. The stanza name that you specify in the `props.conf` file determines how the Splunk platform selects and processes events for anonymization.

Refer to the following stanza specifications:

- The `[host::<host>]` stanza matches events that contain the specified host
- The `[source::<source>]` stanza matches events with the specified source
- The `[<sourcetype>]` stanza matches events with the specified source type
 - ♦ As a best practice, you must subsequently specify the source type in the `inputs.conf` file for this stanza type to work. This option is a Splunk best practice.

Replace strings in events with a sed script

You can use a sed script and the `SEDCMD` method to replace strings or substitute characters. Refer to the following syntax for a sed-style replacement:

```
SEDCMD-<class> = s/<regex>/<replacement>/flags
```

The `SEDCMD` setting has the following components:

- `regex` is a regular expression written in the Perl programming language. It represents what you want to replace.
- `replacement` is the string you want to replace whatever the regular expression matches.
- `flags` can be either the letter `g` to replace all matches or a number to replace a specified match.

Substitute characters in events with a sed script

Refer to the following syntax for a sed character substitution:

```
SEDCMD-<class> = y/<string1>/<string2>/
```

This substitutes each occurrence of the characters in `string1` with the characters in `string2`.

Use a regular expression transform with transforms.conf to anonymize events

Each stanza in the `transforms.conf` configuration file defines a transform class that you can reference from the `props.conf` file for a given source type, source, or host.

Transforms have several settings and variables that let you specify what changes and where, but the following settings are the most important:

- The `REGEX` setting specifies the regular expression that points to the string in the event that you want to anonymize
- The `FORMAT` setting specifies the masked values
- The `%1` variable represents the text of the event before the regular expression that represents the string in the event that you want to mask
- The `%2` variable represents the text of the event after the regular expression
- `DEST_KEY = _raw` writes the value from `FORMAT` to the raw value in the log. This anonymizes the event.

The regular expression processor does not handle multiline events. In cases where events span multiple lines, specify that the event is multiline by placing the string `(?m)` before the regular expression in the `transforms.conf` file.

Anonymize data with a sed script

You can anonymize data by using a sed script to replace or substitute strings in events.

`sed` is a *nix utility that reads a file and modifies the input based on commands that you use within or arguments that you supply to the utility. Many *nix users use the utility for its versatility and fast transformation of incoming data. You can use a sed-like syntax in the `props.conf` file to script the masking of your data in the Splunk platform.

The following is an example of how you would mask files.

Suppose you have a log file called `accounts.log` that contains Social Security and credit card numbers:

```
...
ss=123456789, cc=1234-5678-9012-3456
ss=123456790, cc=2234-5678-9012-3457
ss=123456791, cc=3234-5678-9012-3458
ss=123456792, cc=4234-5678-9012-3459
...
```

You want to mask the fields, so that they appear like this:

```
...
ss=XXXXXX6789, cc=XXXX-XXXX-XXXX-3456
ss=XXXXXX6790, cc=XXXX-XXXX-XXXX-3457
ss=XXXXXX6791, cc=XXXX-XXXX-XXXX-3458
ss=XXXXXX6792, cc=XXXX-XXXX-XXXX-3459
...
```

You can use the `inputs.conf` and `props.conf` configuration files to change the data that comes in from the `accounts.log` file as the Splunk platform accesses it. These configuration files reside in the `$SPLUNK_HOME/etc/system/local/` directory on a heavy forwarder for Splunk Cloud Platform or on a Splunk Enterprise indexer.

Requirements for anonymizing data with a sed script

You must meet the following requirements to anonymize data with a `sed` script:

- Have data that you want to anonymize
- Have an understanding of how regular expressions work
- Have an `inputs.conf` configuration file that points to where the data you want to anonymize is located
- Have a `props.conf` configuration file that references the `sed` script that anonymizes the data

Configure the inputs.conf file to use a sed script

In this example, you create the source type `SSN-CC-Anon` and assign it to the data input for the `accounts.log` file. The transform that you create uses this source type to know what data to transform. While there are other options available for using `SEDCMD` to transform incoming data from a log file, as best practice, create a source type, then assign the transform to that source type in the `props.conf` file.

1. On the machine that runs the heavy forwarder, create an `inputs.conf` file in the `$SPLUNK_HOME/etc/system/local` directory if it doesn't already exist.
2. Open `$SPLUNK_HOME/etc/system/local/inputs.conf` with a text editor.
3. Add the following stanza to reference the `accounts.log` file and assign a source type to the `accounts.log` data.

```
[monitor:///opt/appserver/logs/accounts.log]
sourcetype = SSN-CC-Anon
```

4. Save the file and close it.

Define the sed script in props.conf

In this example, `props.conf` uses the `SEDCMD` setting to perform the transformation directly.

The `-Anon` clause after the `SEDCMD` stem can be any string that helps you identify what the transformation script does. The clause must exist because it and the `SEDCMD` stem form the class name for the script. The text after the equal sign (=) is the

regular expression that invokes the transformation.

1. On the machine that runs the heavy forwarder, create a props.conf file in the `$SPLUNK_HOME/etc/system/local` directory if it doesn't already exist.
2. Open `$SPLUNK_HOME/etc/system/local/props.conf` with a text editor.
3. Add the following stanza to reference the source type that you created in the inputs.conf file to do the masking transformation.

```
[SSN-CC-Anon]
SEDCMD-Anon = s/ss=\d{5}(\d{4})/ss=xxxxx\1/g s/cc=(\d{4})-\{3}(\d{4})/cc=xxxx-xxxx-xxxx-\2/g
```

4. Save the file and close it.
5. Restart the heavy forwarder.

Anonymize data with a regular expression transform

You can mask data by creating a **transform**. Transforms take incoming data and change it based on configurations you supply. In this case, the transformation is the replacement of portions of the data with characters that obscure the real, sensitive data, while retaining the original data format.

Suppose you have an application server log file called `MyAppServer.log` that contains events like the following:

```
"2006-09-21, 02:57:11.58", 122, 11, "Path=/LoginUser Query=CrmId=ClientABC&
ContentItemId=TotalAccess&SessionId=3A1785URH117BEA&Ticket=646A1DA4STF896EE&
SessionTime=25368&ReturnUrl=http://www.clientabc.com, Method=GET, IP=209.51.249.195,
Content=", ""
"2006-09-21, 02:57:11.60", 122, 15, "UserData:<User CrmId="clientabc"
UserId="p12345678"><EntitlementList></EntitlementList></User>", ""
"2006-09-21, 02:57:11.60", 122, 15, "New Cookie: SessionId=3A1785URH117BEA&
Ticket=646A1DA4STF896EE&CrmId=clientabcUserId=p12345678&AccountId=&AgentHost=man&
AgentId=man, MANUser: Version=1&Name=&Debit=&Credit=&AccessTime=&BillDay=&Status=
&Language=&Country=&Email=&EmailNotify=&Pin=&PinPayment=&PinAmount=&PinPG=
&PinPGRate=&PinMenu=&", ""
```

You want to change the data so that the `sessionID` and `Ticket` fields are masked and the events appear as follows:

```
"2006-09-21, 02:57:11.58", 122, 11, "Path=/LoginUser Query=CrmId=ClientABC&
ContentItemId=TotalAccess&SessionId=#####7BEA&Ticket=#####96EE&
SessionTime=25368&ReturnUrl=http://www.clientabc.com, Method=GET, IP=209.51.249.195,
Content=", ""
```

You can use the inputs.conf, props.conf, and transforms.conf files to change the data that comes in from the `MyAppServer.log` file as the Splunk platform accesses it. All of these configuration files reside in the `$SPLUNK_HOME/etc/system/local/` directory on a heavy forwarder for Splunk Cloud Platform or on a Splunk Enterprise indexer.

Requirements for anonymizing data with a regular expression transform

To mask sensitive data, you must meet the following requirements:

- Have data that you want to anonymize
- Have an understanding of how regular expressions work
- Have an inputs.conf configuration file that points to where this data is located
- Have a transforms.conf configuration file that does the data masking
- Have a props.conf configuration file that references the transforms.conf file for the data that you want to mask

Configure inputs.conf

In this example, you create the `MyAppServer-Anon` source type. The transform you create uses this source type to know what data to transform. You can choose from other options for selecting the data to transform.

Follow these steps to configure the `inputs.conf` file for this example:

1. On the machine that runs the heavy forwarder, create an `inputs.conf` file in the `$SPLUNK_HOME/etc/system/local` directory if the file doesn't already exist.
2. Open `$SPLUNK_HOME/etc/system/local/inputs.conf` with a text editor.
3. Add the following stanza to reference the `MyAppServer.log` file and assign a source type to the `MyAppServer.log` data.

```
[monitor:///opt/MyAppServer/logs/MyAppServer.log]
sourcetype = MyAppServer-Anon
```

4. Save the file and close it.

Configure the transforms.conf file

The Splunk platform uses the `transforms.conf` file to perform the transformation of the data. Follow these steps to configure the `transforms.conf` file for this example:

1. On the machine that runs the heavy forwarder, create a `transforms.conf` file in the `$SPLUNK_HOME/etc/system/local` directory if the file doesn't already exist.
2. Open `$SPLUNK_HOME/etc/system/local/transforms.conf` with a text editor.
3. Add the following text to define the transform that anonymizes the `sessionID` field so that only the last four characters in the field are exposed:

```
[session-anonymizer]
REGEX = (?m)^(.*)SessionId=\w+(\w{4}[\&"].*)$
FORMAT = $1SessionId=#####$2
DEST_KEY = _raw
```

4. Add the following text directly underneath the `session-anonymizer` stanza to define the transform for the `Ticket` field, similar to the `sessionID` field:

```
[ticket-anonymizer]
REGEX = (?m)^(.*)Ticket=\w+(\w{4}&.*)$
FORMAT = $1Ticket=#####$2
DEST_KEY = _raw
```

5. Save the file and close it.

Configure the props.conf configuration file

`Props.conf` specifies the transforms to use to anonymize your data. It references one or more transform classes that you define in a `transforms.conf` file.

In this example, `session-anonymizer` and `ticket-anonymizer` are the transform class names that you defined in the `transforms.conf` file.

Follow these steps to configure the `props.conf` file for this example:

1. On the machine that runs the heavy forwarder, create a `props.conf` file in the `$SPLUNK_HOME/etc/system/local` directory if the file doesn't already exist.
2. Open `$SPLUNK_HOME/etc/system/local/props.conf` with a text editor.

3. Add the following stanza to reference the transforms that you created in the transforms.conf file to do the masking transformation.

```
[MyAppServer-Anon]
TRANSFORMS-anonymize = session-anonymizer, ticket-anonymizer
```

4. Save the file and close it.
5. Restart the heavy forwarder.

Example of substitution using a sed/SEDCMD script

Suppose you want to index the file `abc.log`, and you want to substitute the capital letters "A", "B", and "C" for every lowercase "a", "b", or "c" in your events.

Add the following stanza and settings to your `props.conf` file:

```
[source::.../abc.log]
SEDCMD-abc = y/abc/ABC/
```

The Splunk platform substitutes "A" for each "a", "B" for each "b", and "C" for each "c". When you search for `source="*/abc.log"`, the lowercase letters "a", "b", and "c" do not appear in your data.

Caveats for anonymizing data

Anonymizing data can come with the following caveats.

Restrictions for using the sed script to anonymize data

If you use the `SEDCMD` method to anonymize the data, the following restrictions apply:

- The `SEDCMD` script applies only to the `_raw` field at index time. With the regular expression transform, you can apply changes to other fields.

Restrictions for using the regular expression transform to anonymize data

If you use the regular expression transform to anonymize data, include the `LOOKAHEAD` setting when you define the transform and set it to a number that is larger than the largest expected event. Otherwise, anonymization might fail.

Splunk Cloud Platform and Splunk Enterprise indexers do not parse structured data

When you forward structured data to Splunk Cloud Platform or a Splunk Enterprise indexer, the platform does not parse it, even if you configured a `props.conf` file on that indexer with the `INDEXED_EXTRactions` setting. Forwarded data skips the following processing queues on the indexer, which precludes data parsing:

- parsing
- aggregation
- typing

The forwarder must parse the data before it sends that data onward to Splunk Cloud Platform or the Splunk Enterprise indexer. To achieve this, you must set up a `props.conf` file on the forwarder that sends the data. This includes configuring the `INDEXED_EXTRactions` setting and any other parsing, filtering, anonymizing, and routing rules.

Universal forwarders can only parse structured data. See [Forward data extracted from structured data files](#).

Configure timestamps

How timestamp assignment works

Timestamp processing is a key step in **event processing**. Splunk software uses **timestamps** in the following ways:

- Correlating **events** by time
- Creating the timeline histogram in Splunk Web
- Setting time ranges for searches

Splunk software adds timestamps to events at **index time**. It assigns timestamp values automatically by using information that it finds in the raw event data. If there is no explicit timestamp in an event, Splunk software attempts to assign a timestamp value through other means. For some data, you might need to help Splunk software learn to recognize the timestamps. You do this by configuring timestamp extraction. To configure timestamp extraction, see the Configuring timestamp extraction section later in this topic.

Splunk software stores timestamp values in the `_time` field using Coordinated Universal Time (UTC) format.

If you have Splunk Cloud Platform and need to modify timestamp extraction, use a heavy forwarder to ingest the data and send it to the Splunk Cloud Platform instance. You can configure timestamp extraction on the heavy forwarder. If you have Splunk Enterprise and need to modify timestamp extraction, you can modify the configuration files on your Splunk Enterprise instance.

For more information on event processing, see the [Configure event processing](#) chapter.

How Splunk software assigns timestamps

Splunk software uses the following precedence rules to assign timestamps to events:

1. The software looks for a time or date in the event itself using an explicit `TIME_FORMAT`, if provided. You configure the `TIME_FORMAT` attribute in the `props.conf` file.
2. If no `TIME_FORMAT` is configured for the data, Splunk software attempts to automatically identify a time or date in the event itself. It uses the source type of the event, which includes `TIME_FORMAT` information, to try to find the timestamp.
3. If an event has a time and date, but not a year, Splunk software determines the year and builds the timestamp from that date. See [How Splunk software determines timestamps with no year](#).
4. If no events in the source have a date, Splunk software tries to find a date in the source name or file name. Time of day is not identified in file names, so the events must have a time, even if they don't have a date.
5. For file sources, if no date can be identified in the file name, Splunk software uses the file modification time.
6. The software attempts to automatically identify a time or date in the event itself using the advanced timestamp recognition configured in `datetime.xml`.
7. As a last resort, Splunk software sets the timestamp to the current system time when indexing each event.

Splunk software can extract only dates from a source, not times. If you need to extract a time from a source, use a transform configuration. See [Create custom fields at index time](#).

How Splunk software determines timestamps with no year

If Splunk software discovers a timestamp within an event that does not have a year element, it uses the following logic to determine the year:

1. It identifies the current date by using either the date of the event it last parsed or the current clock time.
2. It then uses the year from that date as a base and runs the year through several tests:
 1. If the date in the new event is December 31 and the current date is January 1, it decrements the base year.
 2. If the date in the new event is January 1 and the current date is December 31, it increments the base year.
 3. If the date in the new event is February 29, it determines if the current year is a leap year.
 4. If the current year is a leap year, it uses that year as the base year. If it is not, it uses the previous leap year.
3. If none of the previous tests results in a successful base year determination, the software uses the following procedure to determine the year:
 1. It determines the day of the year of the new event by calculating the number of days from January 1.
 2. If the date information of the previous event is available, and the day of the year of that event is greater than the day of the year of the new event plus 4 days, then it increments the base year.
 3. If the date information of the previous event is not available, and the day of the year of the new event is greater than the current day of the year plus 2 days, then it decrements the base year.
4. The software then assigns the base year to the timestamp for the event. The timestamp must still pass the time range check for the timestamp to be valid.

Example 1

If Splunk software encounters `26 Jun` in a new event on May 26, 2017, and it is not able to determine the year in the previous events, it follows these steps to determine the event timestamp:

1. Since it was not able to determine the year in the previous event, it sets a base year of `2017` as that is the year of the current date.
2. The software checks if the date is December 31 or January 1. The December 31 and January 1 tests fail, so the base year remains `2017`.
3. The software checks if the date occurs in a leap year. The leap year test fails, as the date is not February 29. The base year remains `2017`.
4. Splunk software calculates the day of the year for June 26 as Day `177`.
5. Since the software can't determine the year in the previous event, it adds 2 to this number to arrive at `179`.
6. It then compares `179` to the day of the year of the current date, May 26, 2017, which is Day `147`.
7. Since 179 is greater than 147, the software decrements the year from `2017` to `2016`.
8. The software then builds the new timestamp: `26 Jun 2016`.
9. If the new timestamp falls within the time range that has been set, the software adds the timestamp to the event.

Example 2

If Splunk software encounters `10 Apr` in a new event on May 26, 2017, and it determined the year `2017` in previous events, it follows these steps to determine the event timestamp:

1. Since the software determined the year in the previous event, it sets `2017` as the base year.
2. The software checks if the date is December 31 or January 1. The December 31 and January 1 tests fail, so the base year remains `2017`.
3. The software checks if the date occurs in a leap year. The leap year test fails, as the date is not February 29. The base year remains `2017`.

4. Splunk software calculates the day of the year for April 10 as Day 100.
5. Since the year information in the previous event is available, it adds 4 to this number to arrive at 104.
6. It then compares 104 to the day of the year of the current date, May 26, 2017, which is Day 147.
7. Since 104 is less than 147, the software increments the year from 2017 to 2018.
8. The software then builds the new timestamp: 10 Apr 2018.
9. By default, this new timestamp is not legal, since it falls outside the default `MAX_DAYS_HENCE` setting, which limits valid timestamps to 2 days into the future. The software uses the current date of 26 May 2017 as the timestamp and applies that timestamp to the event.

Configuring timestamp extraction

Most events do not require special timestamp handling. Splunk software automatically recognizes and extracts their timestamps. For some sources and distributed deployments, you might need to configure how timestamps are extracted, so that they format properly.

There are two ways to configure timestamp extraction:

- Use the **Set Source Type** page in Splunk Web to interactively adjust timestamps on sample data. Once you are happy with the results, save the changes to a new source type and then apply that source type to your data inputs. See [Assign the correct source types to your data](#).
- Edit the `props.conf` configuration file directly. See [Configure timestamp recognition](#).

You can also configure timestamp extraction for the following purposes:

- Apply time zone offsets. See [Specify time zones for timestamps](#).
- Extract the correct timestamp from events with more than one timestamp. See [Configure timestamp assignment for events with multiple timestamps](#).
- Improve indexing performance. See [Tune timestamp recognition for better indexing performance](#).

Considerations when adding data from new inputs

If you index data from a new input and then discover that you need to adjust the timestamp extraction process, you must reindex that data after you make the configuration changes.

Consider previewing your data to prevent the need to reindex your data. Alternatively, you can test new data inputs on a separate index on your production Splunk Cloud Platform environment. If you use Splunk Enterprise, you can create a separate test Splunk Enterprise deployment before adding data to your production instance. Creating a separate test deployment allows you to delete and reindex until you get the results you expect.

Configure timestamp recognition

Most events do not require special timestamp handling. The Splunk platform recognizes and extracts timestamps correctly. However, with some sources and distributed deployments, you might need to configure how the Splunk platform extracts timestamps to ensure that the timestamps have the proper format.

You can configure timestamp extraction in these ways:

- Use the Set Source Type page in Splunk Web to interactively adjust timestamps on sample data. Once you're happy with the results, you can save the changes to a new source type and then apply that source type to your data inputs. This option is available on Splunk Cloud Platform only if you upload a file directly to the instance. See [Assign the correct source types to your data](#).
- Use the props.conf configuration file. This option is available on both Splunk Cloud Platform and Splunk Enterprise under the following conditions:
 - ◆ If you have Splunk Cloud Platform and need to modify timestamp extraction, use a heavy forwarder to ingest the data and send it to the Splunk Cloud Platform instance. The heavy forwarder lets you specify configurations that extract the timestamps. You cannot make these configurations with the universal forwarder or with Splunk Cloud Platform directly.
 - ◆ On Splunk Enterprise instances, if you need to modify timestamp extraction, specify the configuration on the indexers. In cases where you have to forward data, you must configure a heavy forwarder to handle these changes.

The timestamp processor

On a Splunk Enterprise instance, you can find the timestamp processor at `$SPLUNK_HOME/etc/datetime.xml` by default. You do not need to edit this file normally, unless you work with unusual custom timestamps. You cannot edit this file on a Splunk Cloud Platform instance because you do not have access to the Splunk Cloud Platform file system.

If you need to configure timestamp recognition, you can make changes by editing timestamp settings in the props.conf configuration file, as described in this topic.

If you have a custom timestamp that can't be handled by configuring the props.conf file, substitute your own timestamp processor with the `DATETIME_CONFIG` setting. This setting specifies the file that the Splunk platform uses for timestamp processing.

Edit timestamp properties in the props.conf configuration file

You can edit timestamp properties on a heavy forwarder to ensure that Splunk Cloud Platform sees and uses the proper timestamps, or you can edit them directly on a Splunk Enterprise instance.

To configure how the Splunk platform recognizes timestamps, edit the props.conf configuration file. With this file, you can control how the Splunk platform processes the timestamps that it sees in incoming events. See the props.conf specification file in the *Admin Manual*.

The props.conf file has several settings for timestamp processing. For example, you can use the `TIME_FORMAT` setting to specify a format for the timestamp that is based on the `strptime()` string to time-structure conversion function.

You can also configure where to look in an event for a timestamp, what time zone to use for events, or how to deal with timestamps of varying currency.

Follow these steps to configure timestamp recognition:

1. For Splunk Cloud Platform instances or on Splunk Enterprise instances that receive data from forwarders, install a new Splunk Enterprise instance and configure it as a heavy forwarder.
2. If you use Splunk Cloud Platform, install the Splunk Cloud Platform universal forwarder credentials package on the forwarder.

3. On the forwarder, use a text editor to edit the props.conf file in the \$SPLUNK_HOME/etc/system/local/ directory or in your own custom application directory in \$SPLUNK_HOME/etc/apps/.
4. Specify timestamp recognition settings that suit your needs, fit your incoming data, and comply with the syntax. See Syntax overview in this topic for more information.
5. Save the changes and restart the forwarder.

The forwarder begins processing timestamps based on the new configuration that you provided.

If you haven't worked with configuration files before, see About configuration files in the *Admin Manual*.

Syntax overview

When you configure timestamp recognition, it is very important that your configuration file and its settings follow the appropriate syntax. Syntax errors can result in the incorrect timestamping and ingestion of the events.

See the following overview of the syntax:

```
[<spec>]
DATEIME_CONFIG = <filename relative to the $SPLUNK_HOME directory>
TIME_PREFIX = <regular expression>
MAX_TIMESTAMP_LOOKAHEAD = <integer>
TIME_FORMAT = <strptime-style format>
TZ = <POSIX time zone string>
MAX_DAYS_AGO = <integer>
MAX_DAYS_HENCE = <integer>
MAX_DIFF_SECS_AGO = <integer>
MAX_DIFF_SECS_HENCE = <integer>
```

In this syntax, <spec> can have any of the following meanings:

- <sourcetype>, the source type of an event.
- host::<host>, where <host> is the host value for an event.
- source::<source>, where <source> is the source value for an event.

If an event contains data that matches the value of <spec>, then the timestamp rules specified in the stanza apply to that event. You can have multiple stanzas to handle different <spec> values.

DATEIME_CONFIG values of NONE or CURRENT must be listed in all caps. Otherwise, a parsing error may occur that could cause data loss.

Timestamp validity settings and their impact on events

By default, Splunk Enterprise indexes all events unless you specifically filter out events through other means.

When you set the MAX_DAYS_AGO, MAX_DAYS_HENCE, MAX_DIFF_SECS_AGO, or MAX_DIFF_SECS_HENCE settings in a stanza, and event timestamps fall outside of the parameters of the settings, then Splunk Enterprise uses the following algorithm to determine the timestamp:

- The software uses the timestamp of the previous event to assign the timestamp of the current event.
- If the timestamp of the previous event can't be determined, then the software uses the current index time to assign a timestamp to the event.

Splunk Enterprise does not drop events if they fall outside of the parameters of these settings.

Timestamp settings

The following timestamp configuration settings are available:

Setting	Description	Default
<code>DATETIME_CONFIG = <filename relative to \$SPLUNK_HOME></code>	<p>Specify a file to use to configure the Splunk timestamp processor.</p> <p>Normally, you don't need to create your own timestamp processor file or modify the default <code>datetime.xml</code> file. The other <code>props.conf</code> settings are usually sufficient. However, if your data has a custom timestamp format, you might need your own version of the <code>datetime.xml</code> file.</p> <p>Set <code>DATETIME_CONFIG = NONE</code> to prevent the timestamp processor from running. When timestamp processing is off, Splunk Enterprise does not look at the text of the event for the timestamp and instead uses the event <code>time of receipt</code>, the time the event arrives through its input. For file-based inputs, the event timestamp is taken from from the modification time of the input file.</p> <p>Set <code>DATETIME_CONFIG = CURRENT</code> to assign the current system time to each event as Splunk Enterprise indexes it.</p> <div><p>Both <code>CURRENT</code> and <code>NONE</code> as values for <code>DATETIME_CONFIG</code> explicitly disable timestamp identification, so the default event boundary detection (<code>BREAK_ONLY_BEFORE_DATE = true</code>) might not work as desired. When using these settings, use <code>SHOULD_LINEMERGE</code>, <code>BREAK_ONLY_*</code>, <code>MUST_BREAK_*</code>, or both settings to control event merging.</p></div>	

`$SPLUNK_HOME/etc/datetime.xml`
`TIME_PREFIX = <regular expression>` When set, Splunk Enterprise uses the specified regular expression to look for a match before trying to extract a timestamp. The timestamp algorithm looks only for a timestamp in the event text that follows the end of the first regular expression match.

Use a regular expression that points exactly before your event timestamp. For example, if the timestamp follows the phrase `abc123` in your events, set `TIME_PREFIX` to `abc123`.

If the `TIME_PREFIX` isn't found in the event text, the timestamp isn't extracted.

`empty string`
`MAX_TIMESTAMP_LOOKAHEAD = <integer>` Specify how far, or how many characters into, an event that the Splunk platform looks for a timestamp.

This constraint is applied starting from the location positioned by `TIME_PREFIX`.

For example, if `TIME_PREFIX` positions a location 11 characters into the event and `MAX_TIMESTAMP_LOOKAHEAD` is set to 10, timestamp extraction will be constrained to characters 11 to 20.

If set to 0 or -1, the length constraint for timestamp recognition is disabled. This can have negative performance implications, which scale with the length of input lines or with event size when `LINE_BREAKER` is redefined for event splitting.

128 characters `TIME_FORMAT = <strftime-style format>` Specifies a `strftime()` format string to extract the timestamp.

`strftime()` is a Unix standard for designating time formats. For more information, see the section [Enhanced strftime\(\) support](#).

`TIME_FORMAT` starts reading after the `TIME_PREFIX` or directly at the start of the event, if there is no `TIME_PREFIX` setting. If you use a `TIME_PREFIX`, it must match up to and including the character before the timestamp begins. If you don't set `TIME_PREFIX` but you do set `TIME_FORMAT`, the timestamp must appear at the very start of each event. If not, `strftime()` will fail and Splunk displays a "Failed to parse timestamp" warning message. Splunk will also write the message onto `splunkd.log`, but NOT tied to each specific event.

For best results, use `<strftime-style format>` to describe the day of the year and the time of day.

If `<strftime-style format>` contains an hour component, but no minute component, `TIME_FORMAT` ignores the hour component. It treats the format as an anomaly and considers the precision to be date-only.

empty string `TZ = <timezone_identifier>` The time zone of an event is determined as follows:

- If the event has a time zone in its raw text such as `UTC` or `-08:00`, use that.
- Otherwise, if `TZ` is set to a valid time zone string, use that. Specify a time zone setting using a value from the TZ database of time zones. See [1] on Wikipedia.
- If an event that arrives at an indexer originates from a forwarder and both indexer and forwarder are version 6.0 or higher, then use the time zone that the forwarder provides.
- Otherwise, use the time zone of the system that runs `splunkd`.

For more details and examples, see [Specify time zones of timestamps](#).

empty string `TZ_ALIAS = <key=value>[,<key=value>]...` Provides admin-level control over how time zone strings extracted from events are interpreted. For example, `EST` can mean Eastern (US) Standard Time or Eastern (Australian) Standard Time. There are many other three letter time zone acronyms with multiple expansions.

There is no requirement to use `TZ_ALIAS` if the default mappings for these values work as expected. For example, `EST` maps to the Eastern US by default.

Has no effect on the `TZ` value. It affects only time zone strings from event text, either from any configured `TIME_FORMAT` or from pattern-based guess fallback.

The setting is a list of `key=value` pairs, separated by commas.

The key is matched against the text of the time zone specifier of the event, and the value is the time zone specifier to use when mapping the timestamp to UTC/GMT.

The value is another `TZ` specifier that expresses the desired offset.

For example: `TZ_ALIAS = EST=GMT+10:00`

See the `props.conf` specification file in the *Admin Manual* for more examples.

not set `MAX_DAYS_AGO = <integer>` Specifies the maximum number of days in the past, from the current date, that an extracted date is valid.

For example, if `MAX_DAYS_AGO = 10`, the Splunk platform ignores dates older than 10 days from the current date and instead either uses the timestamp of the previous event or the current index time of the event if it can't determine a timestamp in the previous event.

The maximum number of days you can set in the past is 10951.

2000 days

If you have data that is more than 2000 days old, increase this setting.

`MAX_DAYS_HENCE = <integer>` Specifies the maximum number of days in the future from the current date that an extracted date is valid.

For example, if `MAX_DAYS_HENCE = 3`, the Splunk platform ignores dates that are more than 3 days in the future and instead either uses the timestamp of the previous event or uses the current index time of the event if it can't determine a timestamp from the previous event.

False positives are less likely with a tighter window. Change this setting with caution.

If your servers have the wrong date set or are in a time zone that is one day ahead, set this value to at least 3. This allows timestamp extractions that are up to a day in the future.

The maximum number of days you can set is 10950.

2 days `MAX_DIFF_SECS_AGO = <integer>` If the event timestamp is more than `<integer>` seconds before the previous timestamp, the Splunk platform accepts it only if it has the same time format as the majority of timestamps from the source.

If your timestamps are wildly out of order, consider increasing this value.

The maximum number of seconds you can set is 2147483646.

3600 seconds (1 hour) `MAX_DIFF_SECS_HENCE = <integer>` If the event timestamp is more than `<integer>` seconds after the previous timestamp, the Splunk platform accepts it only if it has the same time format as the majority of timestamps from the source.

If your timestamps are wildly out of order, or if you have logs that are written less than once a week, consider increasing this value.

The maximum number of seconds you can set is 2147483646.

604800 seconds (1 week)

Enhanced `strptime()` support

Use the `TIME_FORMAT` setting in the `props.conf` file to configure timestamp parsing. This setting takes a `strptime()` format string, which it uses to extract the timestamp.

the Splunk platform implements an enhanced version of Unix `strptime()` that supports additional formats, allowing for microsecond, millisecond, any time width format, and some additional time formats for compatibility. The following additional formats are included:

Format	Description
%N	For GNU date-time nanoseconds. Specify any sub-second parsing by providing the width: %3N = milliseconds, %6N = microseconds, %9N = nanoseconds. Sub-second parsing depends on the setting <code>ADD_EXTRA_TIME_FIELDS</code> . See <code>props.conf</code> in the <i>Admin Manual</i> .
%Q,%q	For milliseconds, microseconds for Apache Tomcat. %Q and %q can format any time resolution if the width is specified.
%l	For hours on a 12-hour clock format. If %l appears after %S or %s like %H:%M:%S.%l, it takes on the log4cpp meaning of milliseconds.
%+	For standard Unix date format timestamps.
%v	For BSD and OSX standard date format.
%Z	The time zone abbreviation. Nothing if there is no time zone information.
%z, %:z, %::z	The time zone offset designator in International Organization for Standardization (ISO) 8601 format. For example, -0800 for PST, +0000 for GMT, or nothing if the time zone can't be determined. Use %:z if the timestamp offset contains hours and minutes, like -08:00. Use %::z if the timestamp offset contains hours, minutes, and seconds, like -08:00:00.
%o	For AIX timestamp support. %o is used as an alias for %Y.
%p	The locale equivalent of AM or PM. There might be no equivalent.
%s	Epoch (10 digits).

A `strptime()` expression that ends with a literal dot and subsecond specifier such as %Q, %q, %N treats the terminal dot and conversion specifier as optional. If the .subseconds portion is absent from the text, the timestamp is still extracted.

strptime() format expression examples

Here are some sample date formats, with the `strptime()` expressions that handle them:

Date format	Expression
1998-12-31	%Y-%m-%d
98-12-31	%y-%m-%d
1998 years, 312 days	%Y years, %j days
Jan 24, 2003	%b %d, %Y
January 24, 2003	%B %d, %Y
1397477611.862	%s.%3N

The Splunk platform doesn't recognize non-English month names in timestamps. If you have an app that writes non-English month names to log files, reconfigure the app to use numerical months, if possible.

Examples

Your data might contain an easily recognizable timestamp, like the following example:

```
...FOR: 04/24/07 PAGE 01...
```

To extract that timestamp, add this stanza in the props.conf file:

```
[host::foo]
TIME_PREFIX = FOR:
TIME_FORMAT = %m/%d/%y
```

You might also have a timestamp that includes time zone information:

```
â |Valid_Until=Thu Dec 31 17:59:59 GMT-06:00 2020
```

To extract the timestamp, add this stanza to the props.conf file:

```
[host::bar]
TIME_PREFIX = Valid_Until=
TIME_FORMAT = %a %b %d %H:%M:%S %Z%:z %Y
```

Your data might contain other information that is parsed as timestamps. Take the following example:

```
...1989/12/31 16:00:00 Wed May 23 15:40:21 2007...
```

The Splunk platform extracts the date as Dec 31, 1989, which is not useful. In this case, configure the props.conf file to extract the correct timestamp from events from `host::foo`:

```
[host::foo]
TIME_PREFIX = \d{4}/\d{2}/\d{2} \d{2}:\d{2}:\d{2} \w+\s
TIME_FORMAT = %b %d %H:%M:%S %Y
```

This configuration assumes that all timestamps from `host::foo` are in the same format. Configure your props.conf stanza to be as granular as possible to avoid potential timestamping errors.

For more information on extracting the correct timestamp from events containing multiple timestamps, see [Configure timestamp assignment for events with multiple timestamps](#).

Configure timestamps for specific needs

You can use the settings described in this topic to configure the timestamp extraction processor for some specialized purposes. For example, you can configure timestamps for the following needs:

- Apply time zone offsets. See [Specify time zones for timestamps](#).
- Pull the correct timestamp from events with more than one timestamp. See [Configure timestamp assignment for events with multiple timestamps](#).
- Improve indexing performance. See [Tune timestamp recognition for better indexing performance](#).

Configure how timestamps appear in search results

You can use your browser locale setting to configure how Splunk Web displays timestamps in search results. For information on setting the browser locale, see [Configure user language and locale in the Admin Manual](#).

Reconfigure how timestamps appear in raw data

Even though the Splunk platform uses the browser locale to configure how timestamps appear in search results, the raw data still remains in its original format. You might want to change this so that the data format is standardized in both raw data and search results. Do this with the `props.conf` and `transforms.conf` files.

For example, assume the timestamp data in the raw event looks like this:

```
06/07/2011 10:26:11 PM
```

However, you want it to correspond with how it appears in search results:

```
07/06/2011 10:26:11 PM
```

You can use the `props.conf` and `transforms.conf` files to transform the timestamp in the raw event.

In the `transforms.conf` file, add this stanza:

```
[resortdate]
REGEX = ^(\d{2})\/(\d{2})\/(\d{4})\s([^/]+)
FORMAT = $2/$1/$3 $4
DEST_KEY = _raw
```

In `props.conf`, add this stanza, where `<spec>` qualifies your data:

```
[<spec>]
TRANSFORMS-sortdate = resortdate
```

Configure timestamp assignment for events with multiple timestamps

If an event contains more than one timestamp, you can specify which timestamp the event is to use for indexing. Configuring the timestamp is especially useful when you are indexing events that contain syslog host-chaining data.

Configure positional timestamp extraction by editing the `props.conf` configuration file. While there is limited ability to configure timestamp extraction in Splunk Web, you can achieve the best results by using configuration files on a heavy forwarder. See `props.conf` for more information on this configuration file.

On a Splunk Cloud Platform instance, you must configure timestamps on a heavy forwarder after you configured that forwarder to send data to the Splunk Cloud Platform instance. For general information on editing the `props.conf` file for timestamps, see [Configure timestamp recognition](#).

If you use Splunk Enterprise and need to modify timestamp extraction, perform the configuration on your indexer machines or, if you are forwarding data, use heavy forwarders and perform the configuration on the machines where the heavy forwarders run.

Configure positional timestamp extraction

To specify the position of the timestamp you want extracted, follow these steps:

1. Add `TIME_PREFIX` and `MAX_TIMESTAMP_LOOKAHEAD` settings to a stanza in the `props.conf` file. By setting a regular expression for `TIME_PREFIX`, you specify the pattern of characters that indicates the point to start looking for the timestamp.

2. Set a value for the `MAX_TIMESTAMP_LOOKAHEAD` setting to specify how far into an event past the `TIME_PREFIX` location to look for the timestamp.

By constraining the amount of time to look ahead, you can improve both the accuracy and performance in determining and extracting the timestamp.

When you set `TIME_PREFIX` setting, the Splunk platform scans the event text for a match to its regular expression before it tries to extract a timestamp. The timestamp algorithm looks for a timestamp in the text following the end of the first regular expression match. For example, if the `TIME_PREFIX` setting is set to `abc123`, only the text following the first occurrence of `abc123` is used for timestamp extraction.

`TIME_PREFIX` also sets the start point for the `MAX_TIMESTAMP_LOOKAHEAD` setting. The lookahead starts after it finds the matched portion of text in the `TIME_PREFIX` regular expression. For example, if `TIME_PREFIX` matches text through the first 11 characters of the event and the timestamp you want to extract is always within the next 30 characters, you can set `MAX_TIMESTAMP_LOOKAHEAD=30`. Timestamp extraction is then limited to text starting with character 12 and ending with character 41.

Example

Examine this example event:

```
1989/12/31 16:00:00 Wed May 23 15:40:21 2007 ERROR UserManager - Exception thrown
Ignoring unsupported search for eventtype: /doc sourcetype="access_combined"
NOT eventtypetag=bot
```

To identify the timestamp as the second string of time information, `May 23 15:40:21 2007`, configure the `props.conf` file like this:

```
[source::/Applications/splunk/var/spool/splunk]
TIME_PREFIX = \d{4}\/\d{2}\/\d{2} \d{2}:\d{2}:\d{2} \w+\s
MAX_TIMESTAMP_LOOKAHEAD = 21
```

This configuration instructs the Splunk platform to locate events that match the first timestamp construction, but to ignore that timestamp in favor of another timestamp that occurs within the following 21 characters, a number it gets from the `MAX_TIMESTAMP_LOOKAHEAD` setting. The Splunk platform finds the second timestamp because it always occurs within that 21-character limit.

You can optimize the speed of timestamp extraction by setting the value of `MAX_TIMESTAMP_LOOKAHEAD` to look only as far into an event as you need for the timestamp you want to extract. In this example, `MAX_TIMESTAMP_LOOKAHEAD` is optimized to look just 21 characters into the event past the regular expression value.

Configure advanced timestamp recognition with `datetime.xml`

The Splunk platform uses the `datetime.xml` timestamp recognition file to extract dates and timestamps from events as it indexes them. The file contains regular expressions that describe how the Splunk platform is to perform those extractions from the raw event data.

In nearly all cases, you do not need to make modifications to the `datetime.xml` file. In those cases where you do make modifications to the file, you must take care in ensuring the its structure remains intact and that there are no typos, as this can cause significant problems with timestamp recognition.

If you need to modify this file on a Splunk Cloud Platform instance, file a support ticket. It is not possible to modify the file on a Splunk Cloud Platform instance directly. Instead, consider whether or not you need to modify the file at all. Configure the file, if necessary, on a universal or heavy forwarder on the machine that contains the data that you want to send to Splunk Cloud Platform.

On Splunk Enterprise, consider using the props.conf configuration file to configure timestamp recognition

In most cases, you do not need to make changes to the datetime.xml timestamp recognition file on Splunk Enterprise instances. The props.conf configuration file is responsible for most timestamp configuration changes.

When you configure timestamp recognition with the props.conf file, Splunk Enterprise uses the datetime.xml file to configure its timestamp processor and extract timestamps out of the events for the source, source type, or host information in your data. If the software can't process the timestamps in your event data, you can configure Splunk Enterprise to extract the timestamps by making a custom version of the datetime.xml file.

Structure of the datetime.xml file

The datetime.xml file has the following parts:

- Code blocks that define individual elements of a time stamp
- Code blocks with other elements defined within the file
- Extraction pattern code blocks

Each definition code block has one or more `<text>` definitions that contain a regular expression that Splunk Enterprise uses to extract the timestamp element.

Code blocks that define individual elements of a time stamp

These individual elements can contain information such as year, month, day, hour, and minute. The following example code block defines the regular expression that Splunk Enterprise uses to extract a literal month element (for example, Jan, Mar) out of event data:

```
<define name="_litmonth" extract="litmonth">
<text><![CDATA[(?![\d\w]) (jan|\x{3127}\x{6708}|feb|\x{4E8C}\x{6708}|mar|\x{4E09}\x{6708}|apr|\x{56DB}\x{6708}|may|\x{4E94}\x{6708}|jun|\x{516D}\x{6708}|jul|\x{4E03}\x{6708}|aug|\x{516B}\x{6708}|sep|\x{4E5D}\x{6708}|oct|\x{5341}\x{6708}|nov|\x{5341}\x{3127}\x{6708}|dec|\x{5341}\x{4E8C}\x{6708}) [a-z,\.;]*)]></text>
</define>
```

Code blocks with other elements defined within the file

The following example code block defines the `_time` element, which extracts hours, minutes, seconds, subseconds, period of day, and time zone:

```
<define name="_time" extract="hour, minute, second, subsecond, ampm, zone">
<text><![CDATA[(?![\d])]]></text>
  <use name="_hour"/>
  <text><![CDATA[:]]></text>
  <use name="_minute"/>
  <text><![CDATA[:]]></text>
  <use name="_second"/>
  <text><![CDATA[(?:\d{4})?[:,\.\.](\d+)? {0,2}]]></text>
  <use name="_ampm"/>
```

```

<text><![CDATA[ {0,2}]]></text>
<use name="_zone"/>
<text><![CDATA[(?!:\d)]]></text>
</define>

```

Extraction pattern code blocks

Extraction pattern code blocks define the order in which to attempt extracting times and dates from incoming event data. In general, the `timePatterns` block defines the order in which the Splunk platform attempts to extract a timestamp under most conditions, and the `datePatterns` block defines how to extract dates.

While extraction code blocks in general define when the Splunk platform attempts timestamp extraction, they do not strictly dictate when timestamp extraction occurs. If, for example, there are multiple matches for a timestamp, the platform uses heuristics that favor matches that contain more information or that occur earlier in the event to determine when to perform an extraction.

Create or modify a custom datetime.xml file

In nearly all cases, you do not need to modify the `datetime.xml` file. Instead, configure the `props.conf` configuration file for timestamp extractions. See the section [Edit timestamp properties in the props.conf configuration file](#) of [Configure timestamp recognition](#) for instructions.

If Splunk Enterprise does not extract dates and times properly with the `props.conf` file, you might need to modify or substitute `datetime.xml` with a custom version. You can use the `splunk train` CLI command to sample the timestamp data and generate code that you can use to create a custom `datetime.xml` that extracts your timestamp.

The `splunk train` CLI command is deprecated, but is still available to help you create patterns for `datetime.xml` based on your sample timestamp data.

After you create a pattern file with `splunk train`, you can make a copy of the default `datetime.xml` file and add your modifications to it, or you can create a new `datetime.xml` that only contains your custom timestamp definitions.

Never make modifications directly to `$SPLUNK_HOME/etc/datetime.xml`. Splunk Enterprise overwrites this file any time you upgrade, and any errors in the file that occur as the result of your changes can cause serious, lasting problems with data ingestion for both your custom source type and all other source types on the instance. If you want to make changes to the default file, save a copy to `$SPLUNK_HOME/etc/system/local` and make the changes there.

To create or modify a custom `datetime.xml` file, follow these high-level steps:

1. [Create a sample timestamp pattern file.](#)
2. [Run the splunk train CLI command against the file.](#)
3. [Use the output to create a custom datetime.xml file.](#)
4. [Reference the custom datetime.xml file in your timestamp configuration.](#)

Create a sample timestamp pattern file

1. From a prompt or PowerShell window, create a text file.
2. Paste the sample of your timestamp data into this file.
3. Save the file and close it.
4. Change to the `$SPLUNK_HOME/bin` directory.

Run the splunk train CLI command against the file

1. Change to the \$SPLUNK_HOME/bin directory:

```
cd $SPLUNK_HOME/bin
```

2. Run the splunk train CLI command:

```
./splunk train dates
```

3. After the software asks the action you want to perform, type `L`, `l`, or `learn` to perform the "learn" action.
4. Enter the path to the file that contains the timestamp sample.
Splunk Enterprise displays the first line of your sample and prompts you to enter values that represent the timestamp:

```
-----  
Interactively learning date formats.  
-----
```

```
INSTRUCTIONS: If a sample line does not have a timestamp, hit Enter.  
If it does have a timestamp, enter the timestamp separated by commas  
in this order: month, day, year, hour, minute, second, ampm, timezone.  
Use a comma as a placeholder for missing values. For example, for a  
sample line like this "[Jan/1/08 11:56:45 GMT] login", the input  
should be: "Jan, 1, 08, 11, 56, 45, , GMT" (note missing AM/PM).  
Spaces are optional.
```

```
SAMPLE LINE 1:
```

```
      Tue Jul 10 21:23:06 PDT 2007 Received Trade 330 with detail user: user3456 date: date:  
10Jul200721:  
      23:06 action: sell 3583 MNAG @ 42
```

```
-----  
Enter timestamp values as: month, day, year, hour, minute, second, ampm, timezone.  
>
```

5. Enter values for month, day, year, hour, minute, second, period of day (am/pm), and time zone.
If the values are sufficient, Splunk software displays the following message to show it remembered the pattern:

```
Learned pattern.
```

```
-----  
If you are satisfied that the timestamps formats have been learned, hit control-c.  
-----
```

6. If Splunk Enterprise correctly learned the timestamp formats, press Ctrl+C.
Splunk software displays text similar to the following:

```
Patterns Learned.
```

```
It is highly recommended that you make changes to a copy of the default datetime.xml file.  
For example, copy "/Applications/splunk/etc/datetime.xml" to  
"/Applications/splunk/etc/system/local/datetime.xml", and work with that file.  
In that custom file, add the below timestamp definitions, and add the pattern names  
to timePatterns and datePatterns list.  
For more details, see http://www.splunk.com/doc/latest/admin/TrainTimestampRecognition
```

```
-----  
<define name="mycustom_date" extract="day,litmonth,year,">  
<text><![CDATA[:\d+\s(w+)\s(\d+)\s(w+)\s(\d+)]]></text>  
</define>  
<define name="mycustom_time" extract="hour,minute,second,ampm,">  
<text><![CDATA[(\d+):(\d+):(\d+)\s(w+)]></text>  
</define>
```

```
-----  
What operation do you want to perform? (default=learn)  
-----
```

Enter choice: [Learn]/Test/Quit >

7. Review the pattern definitions in the output. If the definition for your timestamp sample looks the way that you want, quit the `splunk train` session by typing `Q`, `q`, or `quit`. Otherwise, type in `L`, `l`, or `learn` again to attempt the training operation again.

Use the output to create a custom datetime.xml file

After you successfully train Splunk Enterprise to understand your custom timestamp, you must add the definition that `splunk train` generated to a custom version of `datetime.xml`.

You can create this file using the following options:

- Add your timestamp definitions to an existing `datetime.xml`. This is the preferred method.
- Create a new `datetime.xml` file that contains only your customized timestamp definitions. This option is better when the source type for your data is in a very strict format and Splunk Enterprise was incorrectly choosing a broader default format.

Never make edits to `$(SPLUNK_HOME)/etc/datetime.xml`. Always make a copy of this file and add your custom timestamp patterns to the copy.

1. Make a copy of `datetime.xml` in the `$(SPLUNK_HOME)` directory.

```
cd $(SPLUNK_HOME)/etc
cp datetime.xml system/local/
```

2. Open `$(SPLUNK_HOME)/etc/system/local/datetime.xml` for editing.
3. On its own line, copy the block of code that the `splunk train` command generated and that begins with `define name` into the file. This code block can go anywhere between the `<datetime>` and `<timePatterns>` entries.
4. Within the `<timePatterns>` block, add a line that references the definition line you added earlier in the `datetime.xml` file.
5. Within the `<datePatterns>` code block, add the same line you added in the previous step.
6. Save the custom `datetime.xml` file and close it.

See Examples of custom `datetime.xml` configuration later in this topic for examples.

Reference the custom datetime.xml file in your timestamp configuration

After you build your custom `datetime.xml` file, you can reference it in `props.conf` to extract your custom timestamps. You can set a custom timestamp extraction pattern for any host, source, or source type.

1. In `$(SPLUNK_HOME)/etc/system/local`, create `props.conf` if it does not already exist.
2. Open `props.conf` in `$(SPLUNK_HOME)/etc/system/local` for editing.
3. Add a stanza for the host, source, or source type that requires the custom timestamp extraction, if it does not already exist.
4. Within this stanza, add a `DATETIME_CONFIG` setting that points to the custom `datetime.xml`, relative to the `$(SPLUNK_HOME)` directory. For example:

```
[mysourcetype]
DATETIME_CONFIG = /etc/system/local/datetime.xml
MAX_TIMESTAMP_LOOKAHEAD = 128
MAX_DAYS_AGO = 28
```

5. Repeat the previous steps as necessary for other hosts, sources, or source types that require the custom extraction.
6. Save props.conf and close it.
7. Restart the Splunk platform.
8. Confirm that timestamps are being extracted properly for the events that match the host, source, or source type that contains the custom timestamp extraction pattern.

Examples of custom datetime.xml configuration

The following blocks of code are examples of how to properly configure a custom datetime.xml file.

For example, suppose that the `splunk train` command generated the following code:

```
<define name="mycustom_date" extract="day,litmonth,year,">
<text><![CDATA[:\d+\s\w+\s(\d+)\s(\w+)\s(\d+)]]></text>
</define>
<define name="mycustom_time" extract="hour,minute,second,ampm,">
<text><![CDATA[(\d+):(\d+):(\d+)\s(\w+)]]></text>
</define>
```

See the Example 1a and Example 1b sections for ways you can proceed from this code. Then, see the Example 2 section for the next step.

Example 1a: Modification of existing datetime.xml

Proceeding the previous example, you can then add these definition blocks to an existing datetime.xml in `$SPLUNK_HOME/etc/system/local` that you copied previously:

```
<datetime>

<define name="mycustom_date" extract="day,litmonth,year,">
<text><![CDATA[:\d+\s\w+\s(\d+)\s(\w+)\s(\d+)]]></text>
</define>

<define name="mycustom_time" extract="hour,minute,second,ampm,">
<text><![CDATA[(\d+):(\d+):(\d+)\s(\w+)]]></text>
</define>

<... existing configurations removed for clarity ...>

<timePatterns>
  <use name="_time"/>
  <use name="_hmtime"/>
  <use name="_hmtime"/>
  <use name="_dottime"/>
  <use name="_combdatetime"/>
  <use name="_utcePOCH"/>
  <use name="_combdatetime2"/>
  <use name="mycustom_time"/>
</timePatterns>

<datePatterns>
<use name="_usdate1"/>
  <use name="_usdate2"/>
  <use name="_isodate"/>
  <use name="_eurodate1"/>
  <use name="_eurodate2"/>
```

```

    <use name="_bareurlitdate"/>
    <use name="_orddate"/>
    <use name="_combdatetime"/>
    <use name="_masheddate"/>
    <use name="_masheddate2"/>
    <use name="_combdatetime2"/>
    <use name="mycustom_date"/>
</datePatterns>

</datetime>

```

Example 1b: New datetime.xml with only your timestamp configuration

Instead of the example shown in the Example 1a section, you can also create a new datetime.xml file in \$SPLUNK_HOME/etc/system/local, as follows:

```

<datetime>

<define name="mycustom_date" extract="day,litmonth,year,">
<text><![CDATA[:\d+\s\w+\s(\d+)\s(\w+)\s(\d+)]]></text>
</define>

<define name="mycustom_time" extract="hour,minute,second,ampm,">
<text><![CDATA[(\d+):(\d+):(\d+)\s(\w+)]]></text>
</define>

<timePatterns>
    <use name="mycustom_time"/>
</timePatterns>

<datePatterns>
    <use name="mycustom_date"/>
</datePatterns>

</datetime>

```

Example 2: Reference of new datetime.xml in props.conf for your custom source type

After completing the previous examples, you can then reference the custom datetime.xml file in the configuration for your source type in props.conf, as follows:

```

$SPLUNK_HOME/etc/system/local/props.conf

[my_custom_sourcetype]
DATETIME_CONFIG=/etc/system/local/datetime.xml
SHOULD_LINEMERGE=false
NO_BINARY_CHECK=true

```

Specify time zones for timestamps

If you index data from different time zones, you can use time zone offsets to check that they correlate correctly when you search. You can configure time zones based on the host, source, or source type of an event.

To modify timestamp extraction, your Splunk Cloud Platform architecture must include a heavy forwarder and you must edit the props.conf file on the heavy forwarder. Perform the configuration on the machines where your heavy forwarders run.

If you change the time zone setting of the host machine, you must restart Splunk Enterprise or the forwarder for the software to detect the change.

For general information on editing timestamps in the `props.conf` file, see [Configure timestamp recognition](#).

If you have Splunk Enterprise and need to modify timestamp extraction, perform the configuration on your indexer machines or, if you are forwarding data, use heavy forwarders and perform the configuration on the machines where the heavy forwarders run.

How Splunk software determines time zones

To determine the time zone to assign to a timestamp, Splunk software uses the following logic in order of precedence:

1. Use the time zone specified in raw event data (for example, PST, -0800), if present.
2. Use the `TZ` attribute set in `props.conf`, if the event matches the host, source, or source type that the stanza specifies.
3. If the forwarder and the receiving indexer are version 6.0 or higher, use the time zone that the forwarder provides.
4. Use the time zone of the host that indexes the event.

If Splunk has multiple specified time zones, it will use the one higher in precedence.

Specify time zones in `props.conf`

To configure time zone settings, edit the `props.conf` file in `$FORWARDER_HOME/etc/system/local/` or in your own custom application directory in `$FORWARDER_HOME/etc/apps/`. For information on configuration files in general, see [About configuration files in the Splunk Enterprise Admin Manual](#).

Configure time zones by adding a `TZ` attribute to the appropriate stanza in the `props.conf` file. The `TZ` attribute recognizes zone info TZ IDs. Inside the stanza for a host, source, or source type, set the `TZ` attribute to the TZ ID for the desired time zone. Make sure that the time zone of the events you enter is the time zone coming from that host, source, or source type.

To view a list of all the time zone TZ IDs, see https://en.wikipedia.org/wiki/List_of_tz_database_time_zones.

You do not configure the time zone for the indexer on the Splunk Platform, but instead in the underlying operating system. As long as the time is set correctly on the host system of the indexer, the offsets to event time zones are calculated correctly.

Examples of time zone specification in `props.conf`

The following are examples of how to specify time zones in `props.conf`.

In the first example, events come into the forwarder from New York City in the U.S./Eastern time zone and Mountain View, California in the U.S./Pacific time zone. To correctly handle the timestamps for these two sets of events, you must set the time zone for the `props.conf` for the forwarder to be specified as U.S./Eastern and U.S./Pacific respectively.

The first example sets the time zone to U.S./Eastern for any events coming from hosts whose names match the regular expression `nyc.*`:

```
[host::nyc*]  
TZ = US/Eastern
```

The second example sets the time zone to U.S./Pacific for any events coming from sources in the path `/mnt/ca/...`:

```
[source::/mnt/ca/...]  
TZ = US/Pacific
```

zoneinfo (TZ) database time zone values

The zoneinfo database is a publicly maintained database of time zone values. The location and content of the TZ database depend on your operating system. The following list shows the locations of the TZ database for several common operating systems.

- UNIX versions of Splunk software rely on a TZ database included with the UNIX distribution you're running on. Most UNIX distributions store the database in the `/usr/share/zoneinfo` directory.
- Solaris versions of Splunk software store TZ information in the `/usr/share/lib/zoneinfo` directory.
- Windows versions of Splunk software ship with a copy of the TZ database.

See list of tz database time zones for a list of tz database time zones.

Map timezone strings extracted from event data

Use the `TZ_ALIAS` attribute in `props.conf` to change how Splunk software interprets the timezone acronym string occurring in event data. For example, "EST" means Eastern (U.S.) Standard Time by default, but your event data might be using that value instead to designate Eastern (Australian) Standard Time. To change the meaning of "EST" to the latter, set the attribute using the following syntax:

```
TZ_ALIAS = EST=GMT+10:00
```

Then, when Splunk software encounters "EST" in event data, it will interpret it as "GMT+10:00", rather than the default of "GMT- 5:00".

As this example shows, you can map a timezone string to an existing string plus its offset value. You can also map one TZ string directly to another.

When mapping timezone strings, be sure to handle both summer and winter versions of the time zones. For example, if you map Eastern Standard Time, you must also map Eastern Daylight Time. Test your software to see what timezone strings it produces.

The syntax for `TZ_ALIAS` is:

```
TZ_ALIAS = <key=value>[,<key=value>]...
```

For more information about editing the `props.conf` file, including examples, see the `props.conf` specification in the configuration file reference chapter of the Splunk Enterprise *Admin Manual*.

Set the time zone for a user's search results

When you add or edit users, you can set a user time zone. Search results for that user appear in the specified time zone. This setting, however, does not change the actual event data, whose time zone is determined at index time. For information on setting this value, see Create and manage users with Splunk Web in the *Securing the Splunk Platform* manual.

Tune timestamp recognition for better indexing performance

To speed up indexing, you can use the `props.conf` configuration file to adjust how far ahead into events you want the timestamp processor to look. You can even turn off the timestamp processor altogether.

If you use Splunk Cloud Platform and need to modify timestamp extraction, use a heavy forwarder and perform the configuration on the machines where the heavy forwarders run.

If you use Splunk Enterprise and need to modify timestamp extraction, perform the configuration on your indexer machines. If you are forwarding data, use heavy forwarders and perform the configuration on the machines where the heavy forwarders run. If you use Splunk Cloud Platform and need to modify timestamp extraction, use a heavy forwarder and perform the configuration on the machines where the heavy forwarders run. For information on editing the `props.conf` configuration file for timestamps, see [Configure timestamp recognition](#).

Adjust timestamp lookahead

Timestamp lookahead determines how many characters into an event the timestamp processor looks for a timestamp. Adjust how far the timestamp processor looks by adjusting the `MAX_TIMESTAMP_LOOKAHEAD` setting.

The default number of characters that the timestamp processor looks into an event is 128. You can set the `MAX_TIMESTAMP_LOOKAHEAD` setting to a lower value to speed up indexing. Do this if the timestamps always occur in the first part of the event.

This examples looks or timestamps in the first 20 characters of events coming from the source `foo`:

```
[source::foo]
MAX_TIMESTAMP_LOOKAHEAD = 20
...
```

Disable timestamp processor

You can turn off the timestamp processor entirely to improve indexing performance. Turn off timestamp processing for events that match a specific host, source, or source type by configuring the `DATETIME_CONFIG` setting to `NONE`. When `DATETIME_CONFIG=NONE`, Splunk software doesn't look at the text of the event for the timestamp. Instead, it uses the event time of receipt, or the time the event is received from its input. For file-based inputs such as `monitor`, the timestamp comes from the modification time of the input file.

You can also increase indexing performance by setting the `DATETIME_CONFIG` setting to `CURRENT`, which assigns the current system time to each event at the time of indexing.

This example turns off timestamp extraction for events that come from the source `foo`:

```
[source::foo]
DATETIME_CONFIG = NONE
...
```

Both `CURRENT` and `NONE` disable timestamp identification, so the default event boundary detection, `BREAK_ONLY_BEFORE_DATE = true`, might not work as you expect. When you use these settings, specify `SHOULD_LINEMERGE` or the `BREAK_ONLY_*` and `MUST_BREAK_*` settings to control event merging.

Configure indexed field extraction

About indexed field extraction

When Splunk software indexes data, it parses the data stream into a series of **events**. As part of this process, the software adds a number of **fields** to the **event data**. These fields include **default fields** that it adds automatically, as well as any custom fields that you specify.

The process of adding fields to events is known as **field extraction**. There are two types of field extraction:

- **Indexed field extraction**, which takes place when the fields are stored in the index and become part of the event data. There are two types of indexed fields:
 - ♦ Default fields, which Splunk software automatically adds to each event. For more details, see [About default fields](#).
 - ♦ Custom fields, which you specify. For more details, see [Create custom fields at index time](#).
- **Search-time field extraction**, which takes place when you search through data. Splunk software creates those fields when compiling search results and does not store them in the index. For information about this type of field extraction, see About fields in the *Knowledge Manager Manual*.

When working with fields, consider that most machine data either doesn't have structure or has structure that changes constantly. For unstructured data, use search-time field extraction for maximum flexibility. Search-time field extraction is easy to modify after you define it.

Other types of data might exhibit a more fixed structure, or the structure might already be defined within the data or events in the file. You can configure Splunk software to read the structure of these kinds of files, such as comma-separated value (CSV) files, tab-separated value (TSV) files, pipe-separated value files, and JavaScript Object Notation (JSON) data sources, and map fields at index time. To learn how these processes work, see [Extract fields from files with structured data](#).

About default fields (host, source, sourcetype, and more)

When Splunk software indexes data, it tags each event with a number of fields. These fields become part of the index **event data**. The fields that are added automatically are known as **default fields**.

Default fields serve a number of purposes:

- The default field `index` identifies the index in which the event is located.
- The default field `linecount` describes the number of lines the event contains.
- The default field `timestamp` specifies the time at which the event occurred.

Splunk software uses the values in some of the fields, particularly `sourcetype`, when indexing the data, in order to create events properly. Once the data has been indexed, you can use the default fields in your searches.

The complete list of default fields follows:

Type of	List of fields	Description
---------	----------------	-------------

field		
Internal fields	<code>_raw, _time, _indextime, _cd</code>	These fields contain information that Splunk software uses for its internal processes.
Basic default fields	<code>host, index, linecount, punct, source, sourcetype, splunk_server, timestamp</code>	These fields provide basic information about an event, such as where it originated, what kind of data it contains, what index it's located in, how many lines it contains, and when it occurred.
Default datetime fields	<code>date_hour, date_mday, date_minute, date_month, date_second, date_wday, date_year, date_zone</code>	<p>These fields provide additional searchable granularity to event timestamps.</p> <p>Note: Only events that have timestamp information in them as generated by their respective systems will have <code>date_*</code> fields. If an event has a <code>date_*</code> field, it represents the value of time/date directly from the event itself. If you have specified any timezone conversions or changed the value of the time/date at indexing or input time (for example, by setting the timestamp to be the time at index or input time), these fields will not represent that.</p>

For information about default fields from the search perspective, see [Use default fields in the Knowledge Manager Manual](#).

You can also specify additional, custom fields for inclusion in the index. See [Create custom fields at index-time](#) in this chapter.

This topic focuses on three key default fields:

- **host**
- **source**
- **sourcetype**

Defining host, source, and sourcetype

The host, source, and sourcetype fields are defined as follows:

- **host** - An event host value is typically the hostname, IP address, or fully qualified domain name of the network host from which the event originated. The host value lets you locate data originating from a specific device. For more information on hosts, see [About hosts](#).
- **source** - The source of an event is the name of the file, stream, or other input from which the event originates. For data monitored from files and directories, the value of source is the full path, such as `/archive/server1/var/log/messages.0` or `/var/log/`. The value of source for network-based data sources is the protocol and port, such as `UDP:514`.
- **sourcetype** - The source type of an event is the format of the data input from which it originates, such as `access_combined` or `cisco_syslog`. The source type determines how your data is to be formatted. For more information on source types, see [Why source types matter](#).

Source vs sourcetype

Source and source type are both default fields, but they are entirely different otherwise, and can be easily confused.

- The **source** is the name of the file, stream, or other input from which a particular event originates.
- The **sourcetype** determines how Splunk software processes the incoming data stream into individual events according to the nature of the data.

Events with the same source type can come from different sources, for example, if you monitor `source=/var/log/messages` and receive direct syslog input from `udp:514`. If you search `sourcetype=linux_syslog`, events from both of those sources are returned.

Under what conditions should you override host and sourcetype assignment?

Much of the time, Splunk software can automatically identify host and sourcetype values that are both correct *and* useful. But situations do come up that require you to intervene in this process and provide override values.

Override host assignment

You might want to change your default `host` assignment when:

- You load archive data in bulk that was originally generated from a different host and you want those events to have that host value.
- You forward data from a different host. (The forwarder assigns its host name unless you specify otherwise.)
- You are working with a centralized log server environment, which means that all of the data received from that server will have the same host, even if it originated elsewhere.

For detailed information about hosts, see the chapter [Configure host values](#).

Override sourcetype assignment

You might want to change your default `sourcetype` assignment when:

- Splunk software cannot automatically format the data properly, resulting in problems such as wrong timestamping or event linebreaking.
- You want to apply source types to specific events coming through a particular input, such as events that originate from a discrete group of hosts, or even events that are associated with a particular IP address or userid.

There are also steps you can take to expand the range of source types that Splunk software automatically recognizes, or to simply rename source types.

Assign default fields dynamically

This feature lets you dynamically assign default fields, also known as "metadata", to events as they are being consumed by Splunk software. Use this feature to specify source type, host, or source dynamically for incoming data. This feature is useful mainly with scripted data -- either a **scripted input** or an existing file processed by a script.

Do not use dynamic metadata assignment with file monitoring (tail) inputs. For more information about file inputs, see [Monitor files and directories](#) in this manual.

Note: The modular inputs feature has superseded this `***SPLUNK***` header feature. If you need dynamically-generated values for host, source and sourcetype, consider writing a modular input.

To use this feature, you append a single dynamic input header to your file and specify the metadata fields you want to assign values to. The available metadata fields are `sourcetype`, `host`, and `source`.

You can use this method to assign metadata instead of editing `inputs.conf`, `props.conf`, and `transforms.conf`.

Configure a single input file

To use this feature for an existing input file, edit the file (either manually or with a script) to add a single input header:

```
***SPLUNK*** <metadata field>=<string> <metadata field>=<string> ...
```

1. Set `<metadata field>=<string>` to a valid metadata/value pair. You can specify multiple pairs. For example, `sourcetype=log4j host=swan`.

2. Add the single header anywhere in your file. Any data following the header will be appended with the attributes and values you assign until the end of the file is reached.

3. Add your file to `$SPLUNK_HOME/var/spool/splunk` or any other directory being monitored by Splunk.

Configure with a script

In the more common scenario, you write a script to dynamically add an input header to your incoming data stream. Your script can also set the header dynamically based on the contents of the input file.

Create custom fields at index time

In general, you should try to extract your fields at **search time**. However, there are times when you might need to add to the set of custom **indexed fields** that are applied to your events at **index time**.

For example, you might have certain search-time field extractions that noticeably impact search performance. This can happen, for example, if you typically search a large event set with expressions like `foo!=bar` or `NOT foo=bar`, and the field `foo` nearly *always* takes on the value `bar`.

You also might want to add an indexed field if the value of a search-time extracted field exists outside of the field more often than not. For example, if you typically search only for `foo=1`, but 1 occurs in many events that do *not* have `foo=1`, you might want to add `foo` to the list of fields extracted by Splunk at index time.

For more information about creating custom field extractions see [About fields](#) in the Knowledge Manager manual.

If you have Splunk Cloud Platform and want to define index-time field extractions, you must create a private app that contains your desired configurations. If you are a Splunk Cloud Platform administrator with experience creating private apps, see [Manage private apps](#) in your Splunk Cloud Platform deployment. If you have not created private apps, contact your Splunk account representative for help with this customization.

Unless absolutely necessary, do not add custom fields to the set of default fields that Splunk software automatically extracts and indexes at index time. This includes fields such as `timestamp`, `punct`, `host`, `source`, and `sourcetype`. Adding to this list of fields decreases performance, as each indexed field increases the size of the searchable index. Also, you can't change the fields on data that have already been indexed. You can only apply search-time knowledge to those events.

See [About default fields](#).

Define additional indexed fields

Define additional indexed fields by editing `props.conf`, `transforms.conf`, and `fields.conf`.

Edit these files in `$SPLUNK_HOME/etc/system/local/` or in your own custom application directory in `$SPLUNK_HOME/etc/apps/`. For more information on configuration files in general, see [About configuration files in the Admin manual](#).

Where to put the configuration changes in a distributed environment

If you have a **distributed search** deployment, processing is split between search peers (indexers) and a search head. You must deploy the changes as follows:

- Deploy the `props.conf` and `transforms.conf` changes to each of the search peers.
- Deploy the `fields.conf` changes to the search head.

If you are employing heavy forwarders in front of your search peers, the props and transforms processing takes place on the forwarders, not the search peers. Therefore, you must deploy the props and transforms changes to the forwarders, not the search peers.

For details on Splunk Enterprise distributed components, read [Scale your deployment with Splunk Enterprise components](#) in the *Distributed Deployment Manual*.

For details on where you need to put configuration settings, read [Configuration parameters and the data pipeline](#) in the *Admin Manual*.

Field name syntax restrictions

You can assign field names as follows:

- Valid characters for field names are **a-z**, **A-Z**, **0-9**, or `_`.
- Field names cannot begin with **0-9** or `_`. Splunk reserves leading underscores for its internal variables.
- Avoid assigning field names that match any of the [default field names](#).
- Do not assign field names that contain international characters.

Add a regex stanza for the new field to transforms.conf

Follow this format when you define an index-time field transform in `transforms.conf` (Note: Some of these attributes, such as `LOOKAHEAD` and `DEST_KEY`, are only required for certain use cases):

```
[<unique_transform_stanza_name>]
REGEX = <regular_expression>
FORMAT = <your_custom_field_name>:::$1
WRITE_META = [true|false]
DEST_KEY = <KEY>
DEFAULT_VALUE = <string>
SOURCE_KEY = <KEY>
REPEAT_MATCH = [true|false]
LOOKAHEAD = <integer>
```

Note the following:

- The `<unique_stanza_name>` is required for all transforms, as is the `REGEX`.

- `REGEX` is a regular expression that operates on your data to extract fields.
 - ◆ Name-capturing groups in the `REGEX` are extracted directly to fields, which means that you don't have to specify a `FORMAT` for simple field extraction cases.
 - ◆ If the `REGEX` extracts both the field name *and* its corresponding value, you can use the following special capturing groups to skip specifying the mapping in the `FORMAT` attribute:

```
_KEY_<string>, _VAL_<string>
```

- For example, the following are equivalent:

Using `FORMAT`:

```
REGEX = ([a-z]+)=([a-z]+)
FORMAT = $1::$2
```

Not using `FORMAT`:

```
REGEX = (?<_KEY_1>[a-z]+)=(?<_VAL_1>[a-z]+)
```

- `FORMAT` is optional. Use it to specify the format of the field-value pair(s) that you are extracting, including any field names or values that you want to add. You don't need to specify the `FORMAT` if you have a simple `REGEX` with name-capturing groups.
- `FORMAT` behaves differently depending on whether the extraction takes place at search time or index time.
 - ◆ For index-time transforms, you use `$n` to specify the output of each `REGEX` match (for example, `$1`, `$2`, and so on).
 - ◆ If the `REGEX` does not have `n` groups, the matching fails.
 - ◆ `FORMAT` defaults to `<unique_transform_stanza_name>::$1`.
 - ◆ The special identifier `$0` represents what was in the `DEST_KEY` before the `REGEX` was performed (in the case of index-time field extractions the `DEST_KEY` is `_meta`). For more information, see "How Splunk builds indexed fields," below.
 - ◆ For index-time field extractions, you can set up `FORMAT` in several ways. It can be a `<field-name>::<field-value>` setup like:

```
FORMAT = field1::$1 field2::$2 (where the REGEX extracts field values for captured groups "field1" and "field2")
```

or:

```
FORMAT = $1::$2 (where the REGEX extracts both the field name and the field value)
```

However you can also set up index-time field extractions that create concatenated fields:

```
FORMAT = ipaddress::$1.$2.$3.$4
```

When you create concatenated fields with `FORMAT`, it's important to understand that `$` is the only special character. It is treated as a prefix for regex capturing groups **only** if it is followed by a number and **only** if that number applies to an existing capturing group.

So if your regex has only one capturing group and its value is `bar`, then:

```
FORMAT = foo$1 would yield foobar
```

FORMAT = foo\$bar would yield foo\$bar
FORMAT = foo\$1234 would yield foo\$1234
FORMAT = foo\$1\2 would yield foobar\2

- `WRITE_META = true` writes the extracted field name and value to `_meta`, which is where Splunk stores indexed fields. This attribute setting is required for all index-time field extractions, except for those where `DEST_KEY = _meta` (see the discussion of `DEST_KEY`, below).
 - ♦ For more information about `_meta` and its role in indexed field creation, see "How Splunk builds indexed fields," below.
- `DEST_KEY` is required for index-time field extractions where `WRITE_META = false` or is not set. It specifies where Splunk sends the results of the `REGEX`.
 - ♦ For index-time searches, `DEST_KEY = _meta`, which is where Splunk stores indexed fields. For other possible `KEY` values see the `transforms.conf` page in this manual.
 - ♦ For more information about `_meta` and its role in indexed field creation, see [How Splunk builds indexed fields](#), below.
 - ♦ When you use `DEST_KEY = _meta` you should also add `$0` to the start of your `FORMAT` attribute. `$0` represents the `DEST_KEY` value before Splunk performs the `REGEX` (in other words, `_meta`).
 - ♦ **Note:** The `$0` value is in no way derived *from* the `REGEX`.
- `DEFAULT_VALUE` is optional. The value for this attribute is written to `DEST_KEY` if the `REGEX` fails.
 - ♦ Defaults to empty.
- `SOURCE_KEY` is optional. You use it to identify a `KEY` whose values the `REGEX` should be applied to.
 - ♦ By default, `SOURCE_KEY = _raw`, which means it is applied to the entirety of all events.
 - ♦ Typically used in conjunction with `REPEAT_MATCH`.
 - ♦ For other possible `KEY` values see the `transforms.conf` page in this manual.
- `REPEAT_MATCH` is optional. Set it to `true` to run the `REGEX` multiple times on the `SOURCE_KEY`.
 - ♦ `REPEAT_MATCH` starts wherever the last match stopped and continues until no more matches are found. Useful for situations where an unknown number of field-value matches are expected per event.
 - ♦ Defaults to `false`.
- `LOOKAHEAD` is optional. Use it to specify how many characters to search into an event.
 - ♦ Defaults to 4096. You might want to increase your `LOOKAHEAD` value if you have events with line lengths longer than 4096 characters.
 - ♦ Specifically, if the text you need to match is past this number of characters, you will need to increase this value.
 - ♦ Be aware, however, that complex regexes can have very high costs when scanning larger text segments. The speed may fall off quadratically or worse when using multiple greedy branches or lookaheads / lookbehinds.

Note: For a primer on regular expression syntax and usage, see [Regular-Expressions.info](#). You can test regexes by using them in searches with the `rex` search command. Splunk also maintains a list of useful third-party tools for writing and testing regular expressions.

Note: The capturing groups in your regex must identify field names that follow [field name syntax restrictions](#). They can only contain ASCII characters (a-z, A-Z, 0-9 or `_`). International characters will not work.

Link the new field to props.conf

To `props.conf`, add the following lines:

```
[<spec>]
TRANSFORMS-<class> = <unique_stanza_name>
```

Note the following:

- `<spec>` can be:
 - ♦ `<sourcetype>`, the sourcetype of an event.
 - ♦ `host::<host>`, where `<host>` is the host for an event.
 - ♦ `source::<source>`, where `<source>` is the source for an event.
 - ♦ **Note:** You can use regex-type syntax when setting the `<spec>`. Also, source and source type stanzas match in a case-sensitive manner while host stanzas do not. For more information, see the `props.conf` spec file.
- `<class>` is a unique literal string that identifies the namespace of the field (key) you are extracting. **Note:** `<class>` values *do not* have to follow [field name syntax restrictions](#) (see above). You can use characters other than a-z, A-Z, and 0-9, and spaces are allowed.
- `<unique_stanza_name>` is the name of your stanza from `transforms.conf`.

Note: For index-time field extraction, `props.conf` uses `TRANSFORMS-<class>`, as opposed to `EXTRACT-<class>`, which is used for configuring search-time field extraction.

Add an entry to fields.conf for the new field

The Splunk platform uses configurations in `fields.conf` to determine which custom field extractions should be treated as **indexed fields**.

If the new indexed field comes from a source type that is fully or partially composed of unstructured data, you create a separate configuration for each custom indexed field. The stanza names of these configurations are the field names.

If you are adding fields that come from a source type that is composed entirely of structured data, such as JSON-formatted data, you can design source-type-scoped configurations that utilize wildcard expressions to match sets of indexed fields. See [Extract fields from files with structured data](#).

Add an entry to `fields.conf` for the new indexed field:

```
[<your_custom_field_name>]
INDEXED=true
```

- `<your_custom_field_name>` is the name of the custom field you set in the unique stanza that you added to `transforms.conf`.
- Set `INDEXED=true` to indicate that the field is indexed.

If a field of the same name is extracted at search time, you must set `INDEXED=false` for the field. In addition, you must also set `INDEXED_VALUE=false` if events exist that have values of that field that are not pulled out at index time, but which *are* extracted at search time.

For example, say you're performing a simple `<field>::1234` extraction at index time. This could work, but you would have problems if you also implement a search-time field extraction based on a regex like `A(\d+)B`, where the string `A1234B` yields a value for that field of `1234`. This would turn up events for `1234` at search time that Splunk would be unable to locate at index time with the `<field>::1234` extraction.

Restart Splunk for your changes to take effect

Changes to configuration files such as `props.conf` and `transforms.conf` won't take effect until you shut down and restart Splunk on all affected components.

How Splunk builds indexed fields

Splunk builds indexed fields by writing to `_meta`. Here's how it works:

- `_meta` is modified by all matching transforms in `transforms.conf` that contain either `DEST_KEY = _meta` or `WRITE_META = true`.
- Each matching transform can overwrite `_meta`, so use `WRITE_META = true` to append `_meta`.
 - ◆ If you don't use `WRITE_META`, then start your `FORMAT` with `$0`.
- After `_meta` is fully built during parsing, Splunk interprets the text in the following way:
 - ◆ The text is broken into units; each unit is separated by whitespace.
 - ◆ Quotation marks (" ") group characters into larger units, regardless of whitespace.
 - ◆ Backslashes (\) immediately preceding quotation marks disable the grouping properties of quotation marks.
 - ◆ Backslashes preceding a backslash disable that backslash.
 - ◆ Units of text that contain a double colon (::) are turned into extracted fields. The text on the left side of the double colon becomes the field name, and the right side becomes the value.

Indexed fields with regex-extracted values containing quotation marks will generally not work, and backslashes might also have problems. Fields extracted at search time do not have these limitations.

Here's an example of a set of index-time extractions involving quotation marks and backslashes to disable quotation marks and backslashes:

```
WRITE_META = true
FORMAT = field1::value field2::"value 2" field3::"a field with a \" quotation mark" field4::"a field which ends with a backslash\"
```

When Splunk creates field names

Remember: When Splunk creates field names, it applies [field name syntax restrictions](#) to them.

1. All characters that are not in a-z,A-Z, and 0-9 ranges are replaced with an underscore (_).
2. All leading underscores are removed. In Splunk, leading underscores are reserved for **internal fields**.

Index-time field extraction examples

Here are a set of examples of configuration file setups for index-time field extractions.

Define a new indexed field

This basic example creates an indexed field called `err_code`.

transforms.conf

In `transforms.conf` add:

```
[netscreen-error]
REGEX = device_id=[\w+](?<err_code>[^\:]+)
FORMAT = err_code::"$1"
WRITE_META = true
```

This stanza takes `device_id=` followed with a word within brackets and a text string terminating with a colon. The source type of the events is `testlog`.

Comments:

- The `FORMAT =` line contains the following values:
 - ♦ `err_code::` is the name of the field.
 - ♦ `$1` refers to the new field written to the index. It is the value extracted by `REGEX`.
- `WRITE_META = true` is an instruction to write the content of `FORMAT` to the index.

props.conf

Add the following lines to `props.conf`:

```
[testlog]
TRANSFORMS-netscreen = netscreen-error
```

fields.conf

Add the following lines to `fields.conf`:

```
[err_code]
INDEXED=true
```

Restart Splunk for your configuration file changes to take effect.

Define two new indexed fields with one regex

This example creates two indexed fields called `username` and `login_result`.

transforms.conf

In `transforms.conf` add:

```
[ftpd-login]
REGEX = Attempt to login by user: (.*) : login (.*)\.
FORMAT = username::"$1" login_result::"$2"
WRITE_META = true
```

This stanza finds the literal text `Attempt to login by user:`, extracts a username followed by a colon, and then the result, which is followed by a period. A line might look like:

```
2008-10-30 14:15:21 mightyhost awesomeftpd INFO Attempt to login by user: root: login FAILED.
```

props.conf

Add the following lines to `props.conf`:

```
[ftpd-log]
TRANSFORMS-login = ftpd-login
```

fields.conf

Add the following lines to `fields.conf`:

```
[username]
INDEXED=true

[login_result]
INDEXED=true
```

Restart Splunk for your configuration file changes to take effect.

Concatenate field values from event segments at index time

This example shows you how an index-time transform can be used to extract separate segments of an event and combine them to create a single field, using the `FORMAT` option.

Let's say you have the following event:

```
20100126 08:48:49 781 PACKET 078FCFD0 UDP Rcv 127.0.0.0 8226 R Q [0084 A NOERROR] A (4)www(8)google(3)com(0)
```

Now, what you want to do is get `(4)www(8)google(3)com(0)` extracted as a value of a field named `dns_requestor`. But you don't want those garbage parentheses and numerals, you just want something that looks like `www.google.com`. How do you achieve this?

transforms.conf

You would start by setting up a transform in `transforms.conf` named `dnsRequest`:

```
[dnsRequest]
REGEX = UDP[^\(\]+\(\d\)(\w+)\(\d\)(\w+)\(\d\)(\w+)
FORMAT = dns_requestor:::$1.$2.$3
```

This transform defines a custom field named `dns_requestor`. It uses its `REGEX` to pull out the three segments of the `dns_requestor` value. Then it uses `FORMAT` to order those segments with periods between them, like a proper URL.

Note: This method of concatenating event segments into a complete field value is something you can *only* perform with index-time extractions; search-time extractions have practical restrictions that prevent it. If you find that you must use `FORMAT` in this manner, you will have to create a new indexed field to do it.

props.conf

Then, the next step would be to define a field extraction in `props.conf` that references the `dnsRequest` transform and applies it to events coming from the `server1` source type:

```
[server1]
TRANSFORMS-dnsExtract = dnsRequest
```

fields.conf

Finally, you would enter the following stanza in `fields.conf`:

```
[dns_requestor]
INDEXED = true
```

Restart Splunk for your configuration file changes to take effect.

Extract fields from files with structured data

Many structured data files, such as comma-separated value (CSV) files and Internet Information Server (IIS) web server logs, have information in the file header that can be extracted as fields during indexing. You can configure Splunk Enterprise and the Splunk universal forwarder to automatically extract these values into fields that can be searched. For example, a CSV file starts with a header row that contains column headers for the values in subsequent rows:

```
host,status,message,"start date"
srv1.splunk.com,error,"No space left on device",2013-06-10T06:35:00
srv2.splunk.com,ok,-,2013-06-11T06:00:00
```

Input types that the indexed field extraction feature supports

This feature works with the following input types:

- File-based inputs only (such as monitoring files, directories, or archives.)
- Inputs that use the `oneshot` input type (or through the "Upload" feature in Splunk Web.)

It does not work with modular inputs, network inputs, or any other type of input.

More information on source types and time stamps

- For information on how to set source types when importing structured data files, see [The "Set source type" page](#).
- For information on how to adjust timestamps when previewing indexing results, see [Adjust time stamps and event breaks](#).
- For more general information about configuration files, see About configuration files in the Admin manual.

Use Splunk Web to extract fields from structured data files

When you upload or monitor a structured data file, Splunk Web loads the "Set Source type" page. This page lets you preview how your data will be indexed. See [The 'Set Source type' page](#).

1. From the Add Data page in Splunk Web, choose **Upload** or **Monitor** as the method that you want to add data.

2. Specify the structured data file that you want the software to monitor. Splunk Web loads the "Set Source type" page. It sets the source type of the data based on its interpretation of that data. For example, if you upload a CSV file, it sets the source type to `csv`.
3. Review the events in the preview pane on the right side of the page. The events are formatted based on the current source type.
4. If the events appear to be formatted correctly, click "Next" to proceed to the "Modify input settings" page. Otherwise, configure event formatting by modifying the timestamp, event breaking, and delimited settings until the previewed events look the way that you want.
5. If you don't want to save the settings as a new source type, return to Step 4. Otherwise, click the **Save As** button to save the settings as a new source type.
6. In the dialog that appears, type in a name and description for the new source type.
7. Select the category for the source type by selecting the category you want from the "Category" drop-down.
8. Select the application context that the new source type should apply to by choosing from the entries in the "App" drop-down.
9. Click "Save" to save the source type.
10. Return to Step 4 to proceed to the "Modify input settings" page.

Structured data files with large numbers of columns might not display all extracted fields in Splunk Search

If you index a structured data file with a large number of columns (for example, a CSV file with 300 columns), you might experience a problem later where the Search app does not appear to return or display all of the fields for that file. While Splunk software has indexed all of the fields correctly, this anomaly occurs because of a configuration setting for how Splunk software extracts the fields at search time.

Before Splunk software displays fields in Splunk Web, it must first extract those fields by performing a search time field extraction. By default, the limit for the number of fields that can be extracted automatically at search time is 100. You can set this number higher by editing the `limits.conf` file in `$SPLUNK_HOME/etc/system/local` and changing the `limit` setting to a number that is higher than the number of columns in the structured data file.

```
[kv]
limit = 300
```

If you work with a lot of large CSV files, you might want to configure the setting to a number that reflects the largest number of columns you expect your structured data files to have.

Use configuration files to enable automatic header-based field extraction

You can also use a combination of `inputs.conf` and `props.conf` to extract fields from structured data files. Edit these files in `$SPLUNK_HOME/etc/system/local/` or in your own custom application directory in `$SPLUNK_HOME/etc/apps/<app_name>/local`. `Inputs.conf` specifies the files you want to monitor and the source type to be applied to the events they contain, and `props.conf` defines the source types themselves. If you have Splunk Enterprise, you can edit the settings on indexer machines or machines where you are running the Splunk universal forwarder. You must restart Splunk Enterprise for any changes that you make to `inputs.conf` and `props.conf` to take effect. If you have Splunk Cloud Platform and want configure the extraction of fields from structured data, use the Splunk universal forwarder.

Props.conf attributes for structured data

To configure field extraction for files that contain headers, modify the following attributes in `props.conf`. For additional attributes in `props.conf`, review the `props.conf` specification file.

Attribute	Description	Default
INDEXED_EXTRACTIONS = <CSV W3C TSV PSV JSON HEC>	Specifies the type of file and the extraction and/or parsing method to be used on the file. Note: If you set INDEXED_EXTRACTIONS=JSON, check that you have not also set KV_MODE = json for the same source type, which would extract the JSON fields twice, at index time and again at search time.	n/a (not set)
PREAMBLE_REGEX	Some files contain preamble lines. This attribute contains a regular expression that Splunk software uses to ignore any matching lines.	n/a
FIELD_HEADER_REGEX	A regular expression that specifies a pattern for prefixed header line. Splunk software parses the first matching line into header fields. Note that the actual header starts after the matching pattern, which is not included in the parsed header fields. You can specify special characters in this attribute.	n/a
FIELD_DELIMITER	Specifies which character delimits or separates fields in the monitored file or source. You can specify special characters in this attribute.	n/a
FIELD_QUOTE	Specifies the character to use for quotes in the specified file or source. You can specify special characters in this attribute.	n/a
HEADER_FIELD_ACCEPTABLE_SPECIAL_CHARACTERS	Specifies which special characters can appear in header fields. When not set, the Splunk software replaces all characters that are neither alphanumeric or a space with underscores. If this setting is configured, the Splunk software does not perform a special character replacement in header field names when the special character matches one that you specify. For example, if you configure this setting to . , this setting does not replace that character with underscores during CSV ingestion.	n/a
HEADER_FIELD_DELIMITER	Specifies which character delimits or separates field names in the header line. You can specify special characters in this attribute. If HEADER_FIELD_DELIMITER is not specified, FIELD_DELIMITER applies to the header line.	n/a
HEADER_FIELD_QUOTE	Specifies which character is used for quotes around field names in the header line. You can specify special characters in this attribute. If HEADER_FIELD_QUOTE is not specified, FIELD_QUOTE applies to the header line.	n/a
HEADER_FIELD_LINE_NUMBER	Specifies the line number of the line within the file that contains the header fields. If set to 0, Splunk attempts to locate the header fields within the file automatically.	0
TIMESTAMP_FIELDS = field1,field2,...,fieldn	Some CSV and structured files have their timestamp encompass multiple fields in the event separated by delimiters. This attribute tells Splunk software to specify all such fields which constitute the timestamp in a comma-separated fashion.	Splunk Enterprise tries to automatically extract the timestamp of the event.
FIELD_NAMES	Some CSV and structured files might have missing headers. This attribute specifies the header field names.	n/a
MISSING_VALUE_REGEX	If Splunk software finds data that matches the specified regular expression in the structured data file, it considers the value for the field in the row to be empty.	n/a

Special characters or values are available for some attributes

You can use special characters or values such as spaces, vertical and horizontal tabs, and form feeds in some attributes. The following table lists these characters:

Special value	Props.conf representation
form feed	\f
space	space or ' '
horizontal tab	\t or tab
vertical tab	\v
whitespace	whitespace
none	none or \0
file separator	fs or \034
group separator	gs or \035
record separator	rs or \036
unit separator	us or \037

You can use these special characters for the following attributes only:

- FIELD_DELIMITER
- FIELD_HEADER_REGEX
- FIELD_QUOTE

Edit configuration files to create and reference source types

To create and reference the new source types to extract files with headers:

1. Using a text editor, open the file `props.conf` in the appropriate location as described in [Use configuration files to enable automatic header-based field extraction](#) earlier in this topic. If the `props.conf` file does not exist, you must create it.
2. Define a new sourcetype by creating a stanza which tells Splunk Enterprise how to extract the file header and structured file data, using the attributes described above. You can define as many stanzas - and thus, as many sourcetypes - as you like in the file. For example:

```
[HeaderFieldsWithFewEmptyFieldNamesWithSpaceDelim]
FIELD_DELIMITER=,
HEADER_FIELD_DELIMITER=\s
FIELD_QUOTE="
```

3. Save the `props.conf` file and close it.
4. Create a file `inputs.conf` in the same directory, if it does not already exist.
5. Open the file for editing.
6. Add a stanza which represents the file or files that you want Splunk Enterprise to extract file header and structured data from. You can add as many stanzas as you wish for files or directories from which you want to extract header and structured data. For example:


```
[monitor:///opt/test/data/StructuredData/HeaderFieldsWithFewEmptyFieldNamesWithSpaceDelim.csv]
sourcetype=HeaderFieldsWithFewEmptyFieldNamesWithSpaceDelim
```

7. Save the inputs.conf file and close it.

8. Restart Splunk Enterprise or the universal forwarder for the changes to take effect.

Scope indexed structured data fields by source type to improve search performance

Field extraction from structured data formats with fixed semantic schemas such as JSON tend to yield sets of like-named fields, due to the hierarchical field-naming systems that those formats employ. When you update `fields.conf` to add these field sets to your set of custom **indexed fields**, you can use a source-type-scoped wildcard expression to cover the entire set of custom indexed fields in one configuration.

This method isn't just a convenient way to configure groups of extracted structured data fields as custom indexed fields. It also gives you an advantage when you search those fields. Searches on custom indexed fields that have been configured in `fields.conf` with source-type-scoped wildcard expressions complete faster than searches against the same fields when they are configured individually in `fields.conf` with `[<field name>]` stanzas.

For more information about how `fields.conf` is used to identify individual index-time fields from unstructured data, see [Create custom fields at index time](#).

Format for a source-type-scoped custom indexed field configuration in fields.conf

This is the format for a source-type-scoped custom indexed field configuration. It uses a wildcard expression to target a set of fields that have been extracted from structured data.

```
[sourcetype::<sourcetype>::<wildcard_expression>]
INDEXED=true
```

The `<sourcetype>` must contain only structured data, such as JSON-formatted data. Do not apply source-type-scoped wildcard expressions to source types that contain unstructured data.

The `<wildcard_expression>` should target a set of extracted structured data fields. All of the field names that match the wildcard expression are scoped with the specified source type.

You can use the wildcard expression to assign field names as follows:

- Valid characters for field names are **a-z**, **A-Z**, **0-9**, **.**, **:**, and **_**.
- Field names cannot begin with **0-9** or **_**. Splunk reserves leading underscores for its internal variables.
- Do not assign field names that contain international characters.
- Do not assign field names that match any of the **default field names**.
- Do not create indexed fields with names that collide with names of fields that are extracted at search time.

Set `INDEXED=true` to indicate that the set of fields matched by the configuration are indexed.

Source-type-scoped indexed field configurations require `indexed_fields_expansion=t` in `limits.conf`.

Source-type-scoped wildcard expression example

This configuration indexes all extracted fields from the `splunk_resource_usage` source type that begin with the string `data..`

```
[sourcetype::splunk_resource_usage::data.*]
INDEXED=True
```

With this configuration, the following search runs faster than it would if you had simply set up a `fields.conf` configuration for `[data.search_props.delta_scan_count]`.

```
index=_introspection data.search_props.delta_scan_count=0
```

Forward fields extracted from structured data files

Forward fields extracted from a structured data file to a heavy forwarder or a universal forwarder.

To forward fields extracted from structured data files:

1. Configure the Splunk instance that monitors the files to forward data to a heavy forwarder or a universal forwarder.\.
2. Configure the receiving instance.
3. On the monitoring instance, configure `props.conf` and `inputs.conf` to properly handle event breaking and timestamps for your data. You can do this in one of two ways.
 - To use Splunk Web, follow the instructions in [Use Splunk Web to extract fields from structured data files](#) earlier in this topic.
 - To use configuration files, follow the instructions in [Edit configuration files to create and reference sourcetypes](#) earlier in this topic.
4. Optionally, if you need to transform this data in any way prior to indexing it, edit `transforms.conf`.
5. Restart the receiving instance.
6. Restart the monitoring instance.
7. On the receiving instance, use the Search app to confirm that the fields have been extracted from the structured data files and properly indexed.

Caveats to extracting fields from structured data files

Splunk software does not parse structured data that has been forwarded to an indexer

When you forward structured data to an indexer, it is not parsed when it arrives at the indexer, even if you have configured `props.conf` on that indexer with `INDEXED_EXTRactions`. Forwarded data skips the following pipelines on the indexer, which precludes any parsing of that data on the indexer:

- parsing
- merging
- typing

The forwarded data must arrive at the indexer already parsed.

Field extraction settings for forwarded structured data must be configured on the forwarder

If you want to forward fields that you extract from structured data files to another Splunk instance, you must configure the `props.conf` settings that define the field extractions on the forwarder that sends the data. This includes configuration of `INDEXED_EXTRactions` and any other parsing, filtering, anonymizing, and routing rules. Performing these actions on the instance that indexes the data will have no effect, as the forwarded data must arrive at the indexer already parsed.

When you use Splunk Web to modify event break and time stamp settings, it records all of the proposed changes as a

stanza for `props.conf`. You can find those settings in the "Advanced" tab on the "Set Source type" page.

Use the "Copy to clipboard" link in the "Advanced" tab to copy the proposed changes to `props.conf` to the system clipboard. You can then paste this stanza into `props.conf` in a text editor on Splunk instances that monitor and forward similar files.

Only header fields containing data are indexed

When Splunk software extracts header fields from structured data files, it only extracts those fields where data is present in at least one row. If the header field has no data in any row, it is skipped (that is, not indexed). Take, for example, the following csv file:

```
header1,header2,header3,header4,header5
one,1,won,,111
two,2,too,,222
three,3,thri,,333
four,4,fore,,444
five,5,faiv,,555
```

When Splunk software reads this file, it notes that the rows in the `header4` column are all empty, and does not index that header field or any of the rows in it. This means that neither `header4` nor any of the data in its row can be searched for in the index.

If, however, the `header4` field contains rows with empty strings (for example, ""), the field and all the rows underneath it are indexed.

Take care when allowing special characters for header fields

The `HEADER_FIELD_ACCEPTABLE_SPECIAL_CHARACTERS` setting is designed to manage situations where column headers have characters like `.` or `:`. If you do not use this setting, the Splunk software replaces those characters with underscores during the ingestion process.

No support for mid-file renaming of header fields

Some software, such as Internet Information Server, supports the renaming of header fields in the middle of the file. Splunk software does not recognize changes such as this. If you attempt to index a file that has header fields renamed within the file, the renamed header field is not indexed.

Example configuration and data files

Following are an example `inputs.conf` and `props.conf` to give you an idea of how to use the file header extraction attributes.

To extract the data locally, edit `inputs.conf` and `props.conf` to define inputs and sourcetypes for the structured data files, and use the attributes described above to specify how to deal with the files. To forward this data to another Splunk instance, edit `inputs.conf` and `props.conf` on the forwarding instance, and `props.conf` on the receiving instance.

Inputs.conf

```
[monitor:///opt/test/data/StructuredData/CSVWithFewHeaderFieldsWithoutAnyValues.csv]
sourcetype=CSVWithFewHeaderFieldsWithoutAnyValues
```


maximum,vqmorignationdelayaverage,vqmorignationdelayinstant,vqmorignationdelaymaximum,vqmoutoforder,vqmo
 rundiscardpkts,vqmpdpdvs,vqmpdvaveragems,vqmpdvmaximumms,vqmpktsrcvd,vqmrvaluecq,vqmrvaluegl07,vqmrvalueqlq,vq
 rvalueuoninal,vqmreliabilityindex,vqmresynccount,vqmrtdelayaverage,vqmrtdelayinstant,vqmrtdelaymaximum,vqmrtpd
 esturi,vqmrtpdestinationip,vqmrtpdestinationport,vqmrtpssrc,vqmrtpsourceip,vqmrtpsourceport,vqmrtpsourceuri,vq
 msourcointname,vqmsrcifc,vqmstreamquality,vqmterminateddelayaverage,vqmterminateddelayinstant,vqmterminateddelaym
 aximum,vqmthroughputindex,vqmtto,vqmunderrundiscardpkts,vqmvocoderclass,vqmvocodertype,created,modified
 99152,CFG0730084,-3,-2,356,64000,1,280,14,14.29,36,3499,201000,BW163736844290611-173170743@10.253.143.13,2011-06-29
 12:37:37.292,0,4.68,1.43,0.19,0,0,0,0,52,60,15,17,60,10,0,Loopback,0.48,48,46,0,30,1334,10,99.55,10008,9962,0,
 0,,,,,6096147095,2,100590,5029,0.48,87,200,10,50,1,625,487.5,8767,50,99.58,93,50,,518,,2,0.5,
 -60,975,488,179,192,999.3,0,0,4.07,,4.12,,4.2,,4.03,,0.02,63,76,76,,,,43,0,6.8,0,520,10054,87,87,89,93,9,79,12
 ,12,12,6096147089,10.255.43.12,10010,706222942,10.253.136.231,25110,6096147095,eth
 0/1,2,0,54,80,80,18500,6096147089,48,1,0,2011-06-29 12:41:47.303,2011-06-29 12:41:47.303
 99154,CFG0730084,-3,-1,251,64000,4,195,9,20.52,28,3494,359000,BW163502270290611594566299@10.253.143.13,2011-06-29
 12:35:02.324,0,2.88,1.11,3.44,0,0,0,0,40,40,26,24,50,10,0,Loopback,0.31,54,46,0,31,2455,10,99.8,17769,17732,0,
 0,,,,,6096147095,5,71556,3577,0.62,87,200,10,50,1,1120,496.5,15437,50,99.73,123,74,,529,,65,0.67,
 -62,993,496.5,126,139,3404.7,0,0,4.04,,4.07,,4.2,,3.94,,0.36,58,64,69,,,,49,0,286,0,529,17839,86,86,87,93,9,13
 7,8,8,8,6096147089,10.255.43.12,10000,536353626,10.253.136.231,25096,6096147095,eth
 0/1,2,0,48,60,70,30400,6096147089,54,1,0,2011-06-29 12:41:47.342,2011-06-29 12:41:47.342
VeryLargeCSVFile.csv

IncidentNum,Category,Descript,DayOfWeek,Date,Time,PdDistrict,Resolution,Location,X,Y
 030203898,FRAUD,"FORGERY, CREDIT CARD",Tuesday,02/18/2003,16:30,NORTHERN,NONE,2800 Block of VAN NESS
 AV,-122.424612993055,37.8014488257836
 000038261,WARRANTS,WARRANT ARREST,Thursday,04/17/2003,22:45,NORTHERN,"ARREST, BOOKED",POLK ST / SUTTER
 ST,-122.420120319211,37.7877570602182
 030203901,LARCENY/THEFT,GRAND THEFT PICKPOCKET,Tuesday,02/18/2003,16:05,NORTHERN,NONE,VAN NESS AV /
 MCALLISTER ST,-122.42025048261,37.7800745746105
 030203923,DRUG/NARCOTIC,SALE OF BASE/ROCK COCAINE,Tuesday,02/18/2003,17:00,BAYVIEW,"ARREST, BOOKED",1600
 Block of KIRKWOOD AV,-122.390718076188,37.7385560584619
 030203923,OTHER OFFENSES,CONSPIRACY,Tuesday,02/18/2003,17:00,BAYVIEW,"ARREST, BOOKED",1600 Block of KIRKWOOD
 AV,-122.390718076188,37.7385560584619
 030203923,OTHER OFFENSES,PROBATION VIOLATION,Tuesday,02/18/2003,17:00,BAYVIEW,"ARREST, BOOKED",1600 Block
 of KIRKWOOD AV,-122.390718076188,37.7385560584619
UselessLongHeaderToBeIgnored.log

***** Start Display Current Environment *****
 WebSphere Platform 6.1 [ND 6.1.0.21 cf210844.13] running with process name
 sammys_cell_A\fsgwsws189Node_A\sammys_A_c01_s189_m06 and process id 17904
 Detailed IFix information: ID: 6.1.0-WS-WASSDK-AixPPC32-FP0000021 BuildVrsn: null Desc: Software
 Developer Kit 6.1.0.21
 ID: 6.1.0-WS-WAS-AixPPC32-FP0000021 BuildVrsn: null Desc: WebSphere Application Server 6.1.0.21
 ID: 6.1.0-WS-WASSDK-AixPPC32-FP0000019 BuildVrsn: null Desc: Software Developer Kit 6.1.0.19
 ID: 6.1.0-WS-WAS-AixPPC32-FP0000019 BuildVrsn: null Desc: WebSphere Application Server 6.1.0.19
 ID: sdk.FP61021 BuildVrsn: null Desc: WebSphere Application Server 6.1.0.21
 ID: sdk.FP61019 BuildVrsn: null Desc: WebSphere Application Server 6.1.0.19
 ID: was.embed.common.FP61021 BuildVrsn: null Desc: WebSphere Application Server 6.1.0.21
 ID: was.embed.FP61021 BuildVrsn: null Desc: WebSphere Application Server 6.1.0.21
HeaderFieldsWithFewEmptyFieldNamesWithSpaceDelim.csv

"Field 1" "Field 3" "Field 4" "Field 6"
 Value11,Value12,Value13,Value14,Value15,Value16
 Value21,Value22,Value23,Value24,Value25
 Value31,Value32,Value33,Value34,Value35, Value36

FieldHeaderRegex.log

```
Garbage
Garbage
Garbage
Ignore_This_Stuff: Actual_Header1 Actual_Header2
```

Answers

Have questions? Visit [Splunk Answers](#) and see what questions and answers the Splunk community has around extracting fields.

Process events with ingest-time eval

Splunk Enterprise users can create ingest-time eval expressions to process data before indexing occurs.

An ingest-time eval is a type of **transform** that evaluates an expression at index-time. Ingest-time eval provides much of the same functionality provided by search-time eval. The primary difference is that an ingest-time eval processes event data prior to indexing and the new fields and values that result from the evaluation are sent to indexers.

For more information on search-time eval expressions, see Use the eval command and functions in the *Search Manual*.

You can use ingest-time eval expressions to create new fields and perform a wide range of operations on incoming data, including mathematical, statistical, and cryptographic functions. See Evaluation functions in the *Search Reference*.

Why use ingest-time eval?

Ingest-time eval provides an alternative to ingest-time transformations that are difficult or impossible with regular expressions alone, such as normalizing metrics data. See Example of targeted log-to-metrics conversions in the *Metrics manual*.

Through ingest-time eval you can set up ingest-time lookups, which enable you to enrich your data with lookup fields as it is ingested, and before it is indexed. If you have certain lookups that are performed on almost all of your events, you may want to set them up as ingest-time lookups. See [Reduce lookup overhead with ingest-time lookups](#).

Ingest-time eval also gives you more direct control over index-time fields. For example, you can use ingest-time eval to control exactly how an index-time field is stored in the **rawdata journal** of a Splunk Enterprise index. For more information, see How the indexer stores indexes in *Managing Indexers and Clusters of Indexers*.

Ingest-time eval syntax and usage

Ingest-time eval takes a similar format to the search-time `| eval` command. For more information, see `eval` in the *Search Reference*.

An ingest-time eval stanza in `transforms.conf` contains an `INGEST_EVAL` expression. For example:

```
[eval1]
INGEST_EVAL= field3=length (_raw) *2
```

You can also chain multiple comma-separated `INGEST_EVAL` expressions, for example:

```
[eval2]
INGEST_EVAL= field4=_time, field5=field4+1
```

For detailed usage information and examples of `INGEST_EVAL`, see `transforms.conf`.

Search-time calculated fields that use the `EVAL-fieldname` setting in `props.conf` are not available.

Data processing that occurs before indexing with ingest-time eval can impact performance.

Configure an ingest-time eval transform

You configure eval-based transforms the same way you configure other index-time transforms, using a `transforms.conf` file that contains the transform stanza, in conjunction with a `props.conf` file that references it. You must also configure a `fields.conf` file on the search head to enable searching of newly indexed eval fields.

To process event data with ingest-time eval, configure the following files:

Configure `transforms.conf`

To configure `transforms.conf` for ingest-time eval, follow these steps:

1. Create a `transforms.conf` file in the `$SPLUNK_HOME/etc/system/local` directory.
2. Add an ingest-time eval stanza that specifies the `INGEST_EVAL` expression. For example, the following `INGEST_EVAL` expression creates a new field called `eval_user` and populates the field with the lowercase version of the values in the `username` field:

```
[myeval]
INGEST_EVAL = eval_user=lower(username)
```

Configure `props.conf`

To configure `props.conf` for ingest-time eval, follow these steps:

1. Create a `props.conf` in the `$SPLUNK_HOME/etc/system/local` directory.
2. Add a stanza that specifies the data you want to process, such as `<my_sourcetype>`, and references the ingest-time eval stanza in `transforms.conf`. For example:

```
[my_sourcetype]
TRANSFORMS = myeval
```

Ingest-eval transforms require a sourcetype stanza in `props.conf`

You can mix eval-based transforms and regex-based transforms in `props.conf` in any order. The order in which you list the transforms determines when the transforms run relative to other stanzas in `transforms.conf`. For example, `TRANSFORMS = eval1,regex1,eval2,regex2` runs four different `transforms.conf` stanzas in that specific order.

Configure `fields.conf`

To configure `fields.conf` to enable search of ingest-time eval fields, do the following:

1. On the search head, create a `fields.conf` file in the `$SPLUNK_HOME/etc/system/local` directory.
2. Add a stanza that references the newly indexed field created by the `INGEST_EVAL` expression, as follows:

```
[eval_user]
INDEXED = True
```

For more information on how to configure index-time transforms, see [Define additional indexed fields](#).

Examples

For basic and extended examples of eval expressions, see `eval` in the *Search Reference*.

Reduce lookup overhead with ingest-time lookups

If you have certain lookups that you routinely apply to all of your incoming events in Splunk Enterprise, consider processing them at ingest time with ingest-time lookups. You can do this by configuring an ingest-time eval that uses the `lookup()` eval function to add values from lookup tables to events before they are added to your indexes.

Ingest-time lookups are currently only supported in Splunk Enterprise, not Splunk Cloud Platform.

Ingest-time lookup prerequisites

This section covers things you should know before you attempt to configure an ingest-time lookup.

- Familiarize yourself with the `lookup()` function for `eval`, as ingest-time lookups rely upon it to apply output fields and values in the form of JSON objects. See Comparison and conditional functions in the *Search Reference*.
- Get configuration file access.
 - ◆ Review the steps in How to edit a configuration file in the Splunk Enterprise *Admin Manual*.
 - ◆ Read Where you can place (or find) your modified configuration files in the Splunk Enterprise *Admin Manual*.
 - ◆ If your Splunk deployment uses distributed search, use the configuration bundle cluster manager to push CSV lookup files and lookup configurations to cluster members and peers. See Managing Indexers and Clusters of Indexers in *Managing Indexers and Clusters of Indexers*.
- Learn about ingest-time eval expressions. Ingest-time lookups are a type of ingest-time eval expression. You use ingest-time eval expressions to create new fields and perform a wide range of operations on incoming data, including mathematical, statistical, and cryptographic functions. For an overview, see [Process events with ingest-time eval](#).
- Ingest-time lookups are CSV file lookups and as such use CSV files as their lookup tables.
 - ◆ Ingest-time lookups expect the CSV lookup file to be stored in `$SPLUNK_HOME/etc/system/lookups`. If you have a single instance of Splunk you can manually load the file to this location. If you have a distributed search environment, you can use the deployer to update the files on your peers.
 - ◆ You can optionally specify a CSV lookup definition instead of a CSV lookup file. CSV lookup definitions include references to CSV lookup files. CSV lookup definitions can also include filters, field and value matching rules, and other settings. If you specify a CSV lookup definition, you must configure the definition as a `transforms.conf` stanza at `$SPLUNK_HOME/etc/system/local`.

- ◆ For more information about CSV file lookups, see *Configure CSV lookups in the Knowledge Manager Manual*.
- Ingest-time lookups have a syntax that is similar to that of the `lookup` command, and to the syntax of configurations for automatic search-time lookups. See *Make your lookup automatic in the Knowledge Manager Manual*.

Ingest-time lookup syntax

Ingest-time lookups run the `lookup()` function through an `INGEST_EVAL` expression. The syntax looks like this:

```
[lookup1]
INGEST_EVAL= <string>=lookup("<lookup_table>", json_object("<input_field>",<match_field>,...),
json_array("<output_field>",...))
```

If the first quoted string supplied for the `<lookup_table>` lacks a ".csv" file descriptor, the Splunk software assumes it is the name of a CSV lookup definition.

Specify a CSV lookup definition if you want the various settings associated with the definition to apply to the ingest-time lookup. These can include filters, field and value matching rules, and more.

Ingest-time lookup examples

A `lookup()` function can use multiple `<input_field>/<match_field>` pairs to identify events, and multiple `<output_field>` values can be applied to those events. Here is an example of valid `lookup()` syntax with multiple inputs, matches, and outputs.

```
[lookup1]
INGEST_EVAL= <string>=lookup("<lookup_table>", json_object("<input_field1>", <match_field1>,
"<input_field2>", <match_field2>), json_array("<output_field1>", "<output_field2>", "<output_field3>"))
```

You can set up `INGEST_EVAL` expressions that nest a `lookup()` function inside another `eval` function. This example uses a `json_extract` function to pull a field value from the JSON object produced by the `lookup()` function:

```
[lookup-extract]
INGEST_EVAL= status_detail=json_extract(lookup("http_status.csv", json_object("status", status),
json_array("status_description")), "status_description")
```

This results in the ingest-time addition of field-value pairs like `status_detail=Created` and `status_detail=Not Found` to your events, depending on the value of the `status` field in those events.

Limits.conf settings

Two `limits.conf` settings give you deployment-wide control over the usage of ingest-time lookups.

ingest_max_memtable_bytes

The `ingest_max_memtable_bytes` setting puts an upper boundary on the size of CSV lookup tables that are used for ingest-time lookup processing. This ensures that ingest-time lookup files never take up too much space in memory. By default, lookup tables larger than 10mb in size cannot be used for the `lookup()` `eval` function when it is used with `INGEST_EVAL`. When the size of a lookup table being used by an ingest-time lookup exceeds the `ingest_max_memtable_bytes` setting, any lookups that rely on it fail. Error messages indicating their failure appear in `splunkd.log` when Splunk is restarted.

The `ingest_max_memtable_bytes` setting is intentionally separate from `max_memtable_bytes`, a similar setting for search-time lookups. It is set to a lower value so that the indexing pipeline is not affected as much in terms of memory and CPU usage.

ingest_lookup_refresh_period_secs

If you have ingest-time lookups that are based on CSV lookup tables that change on a frequent basis, you can adjust the `ingest_lookup_refresh_period_secs` setting to ensure that these changes are captured. By default this setting ensures that the in-memory lookup tables that are used with the `lookup()` function at ingest time are refreshed every 60 seconds.

Configure host values

About hosts

The **host** field value of an event is the name of the physical device from which the event originates. Because the host field value is a **default field**, which means that Splunk Enterprise assigns a host to every event it indexes, you can use it to search for all events that have been generated by a particular host.

The host value is typically the hostname, IP address, or fully qualified domain name of the networked machine on which the event originated.

Both Splunk Cloud Platform and Splunk Enterprise assign host names at index time, but whereas you can configure host assignment directly on a Splunk Enterprise instance, you must do this configuration on a universal or heavy forwarder for Splunk Cloud Platform.

How Splunk Enterprise determines the host value

Splunk Enterprise assigns a host value to each event by examining settings in the following order and using the first host setting it encounters:

1. Any event-specific host assignment that you specify in the transforms.conf configuration file. For Splunk Cloud Platform, you must use a heavy forwarder to assign host names through events.
2. The default host value for the input that created the event, if any.
3. The default host value for the indexer or forwarder that initially ingests the data.

The default host value

If you don't specify host rules for a source, Splunk Enterprise assigns the host field a default value that applies to all data coming into the instance from any input. The default host value is the hostname or IP address of the indexer or forwarder that initially ingests the data. When Splunk Enterprise or, in the case of Splunk Cloud Platform, the heavy forwarder, runs on the server where the event occurred, the behavior is correct and requires no manual intervention.

For more information, see [Set a default host for a Splunk platform instance](#).

The default host for a file or directory input

If you run Splunk Cloud Platform, Splunk Enterprise on a central log archive, or you work with files that are forwarded from other machines in your environment, you might need to override the default host assignment for events that come from particular inputs.

There are two methods for assigning a host value to data that's received through a particular input: You can define a static host value for all data that comes through a specific input, or you can have the Splunk platform dynamically assign a host value to a portion of the path or filename of the source. The latter method can be helpful when you have a directory structure that segregates each host's log archive in a different subdirectory.

For more information, see [Set a default host for a file or directory input](#).

Event-specific assignments

Some situations require that you assign host values by examining the event data. For example, if you have a central log host sending events to a Splunk Enterprise deployment, you might have several machines that feed data to that main log server. To ensure that each event has the host value of its originating server, you need to use the event data to determine the host value.

For more information, see [Set host values based on event data](#).

Handle incorrectly assigned host values

If your event data gets tagged with the wrong host value, there are a number of ways to either fix or work around the problem. See [Change host values after indexing](#) for fixes to the most common scenarios.

Tag host values

You can tag host values to aid in the execution of robust searches. Tags let you cluster groups of hosts into useful, searchable categories.

For details, see About tags and aliases in the Splunk Enterprise *Knowledge Manager Manual*.

Set a default host for a Splunk platform instance

An event host value is the IP address, host name, or fully qualified domain name of the physical device on the network from which the event originates. Because Splunk software assigns a `host` value at index time for every event it indexes, host value searches let you easily find data that originates from a specific device.

You aren't able to change the default host name on Splunk Cloud Platform. Instead, you can assign host names based on inputs, sources, and source types. Finding data from a specific device is available only on Splunk Enterprise.

Default host assignment

If you haven't specified other host rules for a source, the default host value for an event is the hostname or IP address of the machine that runs the Splunk platform instance that ingests the event data. When the event originates on the Splunk platform instance itself, that host assignment is correct and there is no need to change anything. However, if you forward your data from a different host or if you're bulk-loading archive data, you might want to change the default host value for that data.

To set the default value of the host field, you can use Splunk Web or edit the `inputs.conf` configuration file.

Set the default host value using Splunk Web

Follow these steps to set the default value of the host field for all events coming into that Splunk instance. You can override the value for individual sources or events.

1. In Splunk Web, click **Settings > Server settings**.
2. On the Settings page, click **General settings**.
3. On the General settings page, scroll down to the **Index settings** section and change the **Default host name**.
4. Save your changes.

Set the default host value using inputs.conf

The default host assignment is set in the `inputs.conf` configuration file during installation. You can modify the host value by editing that file in the `$SPLUNK_HOME/etc/system/local/` directory or in your own custom application directory in `$SPLUNK_HOME/etc/apps/`. For more information, see `inputs.conf`.

The host assignment setting appears in the `[default]` stanza in the file.

This is the format of the default host assignment in the `inputs.conf` file:

```
[default]
host = <string>
```

Set the `<string>` value to your chosen default host value. `<string>` defaults to the IP address or domain name of the host where the data originated. Don't put quotes around the `<string>` value. For example, type `host=foo`, not `host="foo"`.

After you edit the `inputs.conf` file, restart your Splunk platform instance to put your changes into effect.

By default, the host setting is configured to the variable `$decideOnStartup`, which means that it's set to the hostname of the machine `splunkd` is running on. The daemon reinterprets the value each time it starts up.

Override the default host value for data received from a specific input

If you're running Splunk Enterprise on a central log archive or you're working with files forwarded from other hosts in your environment, you might need to override the default host assignment for events coming from particular inputs.

There are two methods for assigning a host value to data received through a particular input: you can define a static host value for all data coming through a specific input, or you can dynamically assign a host value to a portion of the path or file name of the source. The latter method can be helpful when you have a directory structure that segregates each host's log archive in a different subdirectory.

For more information, see [Set a default host for a file or directory input](#).

Override the default host value using event data

Some situations require you to assign host values by examining the event data. For example, if you have a central log host sending events to your Splunk platform deployment, you might have several host servers feeding data to that main log server. To ensure that each event has the host value of its originating server, you need to use the event's data to determine the host value.

For more information, see [Set host values based on event data](#).

Set a default host for a file or directory input

You can set a host value for all data from a particular file or directory input on the universal forwarder and Splunk Enterprise. You can set the host statically or dynamically.

On Splunk Cloud Platform, you must use a universal forwarder to assign host values as part of collecting data to send to Splunk Cloud Platform. You cannot configure host names in Splunk Web.

If you set the host value statically, the Splunk platform assigns the same host to every event received from a designated file or directory input.

If you set the host value dynamically, the Splunk platform extracts the same host name from the source input using a regular expression or segment of the full directory path of the source.

You can also assign host values to events that come through a particular file or directory input based on their source or source type values as well as other kinds of information. See [Set host values based on event data](#).

Currently, you cannot enable the setting of default host values for network (TCP and UDP) or **scripted inputs**.

Statically set the default host value

This method applies a single default host value to each event that a specific file or directory input generates.

A static host value assignment affects only new events that a certain input generates. You cannot assign a default host value to data that is already indexed. Instead, you must tag the host value to the existing events. See Define and manage tags in Settings in the *Knowledge Manager Manual*.

Edit the inputs.conf configuration file to set a default host statically

To specify a host value for a monitored file or directory input, you can edit the inputs.conf configuration file. When you edit the inputs.conf file, set the `host` setting in the stanza that defines the input. If you use Splunk Cloud Platform, you configure this setting on the machines where you run the universal forwarder.

```
[monitor://<path>]
host = <your_host>
```

Edit the inputs.conf file in `$SPLUNK_HOME/etc/system/local/` or in your own custom Splunk application directory in `$SPLUNK_HOME/etc/apps/`.

For more information on configuration files in general, see About configuration files in the *Admin Manual*. For more information about inputs and input types, see [What data can I index?](#)

Example of static host value assignment

This example covers any events coming in from `/var/log/httpd`. Any events coming from this input receives a `host` value of `webhead-1`.

```
[monitor:///var/log/httpd]
host = webhead-1
```

Use Splunk Web to set a default host statically

On Splunk Enterprise, you can define a host for a file or directory input whenever you add or edit an input of that type.

To set the default host when creating a new input, see [Set a default host for a new input](#) later in this topic.

To set a default host statically on an existing input, follow these steps:

1. On Splunk Web, click **Settings > Data Inputs**.
2. Click **Files & Directories**.
3. On the **Files & Directories** page, click the name of an existing input to update it.
4. In the **Host** section, select **constant value** from the **Set host** drop-down list.
5. Enter the static host value for the input in the **Host field value** field.
6. Click **Save**.

Set a default host for a new input

When you create an input, you must follow a different process to set a default host.

1. On Splunk Web, click **Settings > Data Inputs**.
2. Click **Files & Directories**.
3. On the **Files & Directories** page, click **New** to add an input.
4. Specify the file or directory that you want to monitor, and specify any allow lists or deny lists.
5. Click **Next**.
6. (Optional) Set the source type for your new input.
If you specified a directory, the Set Source Type page does not appear.
7. Click **Next**.
8. On the **Input Settings** page in the **Host** section, click **Constant Value**.
9. In the **Host field value** field, enter the host name for the input.
10. Click **Review** to continue to the Review page.
11. Click **Submit** to create the input.

Dynamically set the default host value

This method dynamically extracts the host value for a file or directory input, either from a segment of the source input path or from a regular expression. For example, if you want to index an archived directory and the name of each file in the directory contains relevant host information, you can extract this information and assign it to the host field.

You can test regular expressions by using them in searches with the `rex` search command.

Use the `inputs.conf` file to set a default host dynamically

You can set up dynamic host extraction rules by configuring `inputs.conf`. For more information on configuration files in general, see About configuration files in the *Admin Manual*.

Set the event host with the `host_regex` attribute

1. Edit `inputs.conf` in `$SPLUNK_HOME/etc/system/local/` or in your own custom application directory in `$SPLUNK_HOME/etc/apps/`.
2. Use the `host_regex` attribute to override the host field with a value extracted through a regular expression.

```
[monitor://<path>]
host_regex = <your_regular_expression>
```

3. Save the `inputs.conf` file.
4. Restart the Splunk platform instance.

The regular expression extracts the `host` value from the filename of each input. The input uses the first capturing group of the regular expression as the host. If the regular expression fails to match, the input sets the default `host` attribute as the host.

Set the event host with the host_segment attribute

The `host_segment` value overrides the `host` field with a value that is extracted from a segment in the path of your data source.

1. Edit `inputs.conf` in `$SPLUNK_HOME/etc/system/local/` or in your own custom application directory in `$SPLUNK_HOME/etc/apps/`.
2. Add a `host_segment` attribute to a stanza to override the `host` field with a value that is extracted from a segment in the path of your data source. For example, if the path to the source is `/var/log/<host server name>` and you want the host server name or the third segment to be the host value, set `host_segment` as follows:

```
[monitor:///var/log/]
host_segment = 3
```

3. Save the `inputs.conf` file.
4. Restart the Splunk platform instance.

Examples of dynamic host assignment

In this example, the regular expression assigns all events from `/var/log/foo.log` a host value of `foo`:

```
[monitor:///var/log]
host_regex = /var/log/(\w+)
```

This example assigns the host value to the third segment in the path `apache/logs`:

```
[monitor:///apache/logs/]
host_segment = 3
```

Use Splunk Web to set a default host dynamically

1. Click **Settings > Data Inputs**.
2. Click **Files & Directories**.
3. On the **Files & Directories** page, click the name of an existing input to update it.
4. In the **Host** section, select one of the following two options from the **Set host** drop-down list.

Extraction method	Steps
Extract the host name with a regular expression	<ol style="list-style-type: none">1. Select regex on path.2. Enter the regex for the host you want to extract in the Regular expression field.
Extract the host name from a segment in your data source path	<ol style="list-style-type: none">1. Select segment in path.2. Enter the segment number in the Segment number field. For example, if the path to the source is <code>/var/log/<host server name></code> and you want the host server name to be the host value, enter 3 to extract the third segment.

5. Click **Save**.

Dynamically set a default host for a new input

When you create an input, you must follow a different process to set a default host dynamically.

1. Click **Settings > Data Inputs**.
2. Click **Files & Directories**.
3. On the **Files & Directories** page, click **New** to add an input.

4. Specify the file or directory that you want to monitor, and specify any allow lists or deny lists.
5. Click **Next**.
6. (Optional) Set the source type for your new input.
If you specified a directory, the Set Source Type page does not appear.
7. Click **Next**.
8. On the **Input Settings** page in the **Host** section, click either **Regular expression on path** or **Segment in path**.
9. If you choose **Regular expression on path**, enter a regular expression to be used to extract the host name from the source path in the **Regular expression** field. Otherwise, enter the number for the source path segment to be used to determine the host name in the **Segment Number** field.
10. Click **Review** to continue to the Review page.
11. Click **Submit** to create the input.

Caveats to setting the host_segment attribute to extract a host name

There are some caveats to using the `host_segment` attribute in an `inputs.conf` stanza:

- You cannot simultaneously specify the `host_regex` and `host_segment` attributes in the same stanza.
- When you simultaneously specify a `host_segment` and `source` attribute in the same stanza, the behavior of the `host_segment` attribute changes:
 - ◆ If the value you specify for the source contains a forward slash (/), the host value is extracted based on the segment number you specify in `host_segment`.
 - ◆ If `source` does not contain a forward slash (/), or you specify a `host_segment` value that is larger than the number of segments available in `source`, then the Splunk platform can't extract the host value and instead uses the name of the host that extracted the data.

The following examples show what happens when `source` doesn't contain a forward slash, or you specify a `host_segment` value that is larger than the number of segments available in `source`:

Host name	Source path	Inputs.conf configuration	Resulting host value
server01	/mnt/logs/server01	[monitor:///mnt/logs/] host_segment = 3	server01
server01	/mnt/logs/server01	[monitor:///mnt/logs/server01] source = /mnt/logs/server01 host_segment = 3	server01
server02	/mnt/logs/server02	[monitor:///mnt/logs/server02] source = serverlogs host_segment = 3	server02

Set host values based on event data

You can configure the Splunk platform to assign host names to your events based on the data in those events. You can use event data to override the default assignment that the Splunk platform makes by supplying a regular expression for the event data and configuring two configuration files to determine when the platform is to override the host name for an event.

On Splunk Cloud Platform, you must configure a heavy forwarder to perform host name assignment, and then forward that data to your Splunk Cloud Platform instance. You have to take this extra step because you can't edit configuration files on a Splunk Cloud Platform instance directly.

On Splunk Enterprise, you can edit configuration files, either on an indexer or a heavy forwarder. In any case, you can't use a universal forwarder because universal forwarders can't transform data except in certain limited cases.

For a primer on regular expression syntax and usage, see the website <http://www.regular-expressions.info/reference.html>. You can test regular expressions by using them in searches with the `rex` search command.

Using configuration files to override the host name default field in events

The Splunk platform tags event data with default fields while the data is being ingested. Creating host name overrides for events that the Splunk platform indexes requires you to edit two configuration files on the Splunk platform instance that collects the data, based on some of those default fields.

About updating the `transforms.conf` file

The first file, `transforms.conf`, configures the host name override by using a regular expression to determine when the instance is to overwrite, or transform, the host name default field. You supply the regular expression by determining what exactly in your event data is to trigger the transformation, and then providing that regular expression to the `transforms.conf` file. You provide this regular expression as a stanza within the file, and the Splunk platform triggers the override when incoming event data matches the regular expression that you specify.

About updating the `props.conf` file

The second file, `props.conf`, determines the default fields to which the host name override can apply. These fields appear as a stanza within the file which specifies the default fields where the Splunk platform can potentially modify the host name field for incoming events.

You can apply host name overrides to the following default fields:

- The source, using the `source::<source>` keyword
- The source type, using the `sourcetype=<sourcetype>` keyword
- The host name, using the `host::<host>` keyword

Host name overrides occur when you specify one of these default fields in the `props.conf` file. The following events must occur before the Splunk platform can override the host name:

- The host, source, or sourcetype in the incoming event data must match what you specify in the `props.conf` file to activate the host name override transform configuration in the `transforms.conf` file.
- The event data must match the regular expression you set for the host name override transform to trigger.

Create a host name override

1. Review your event data to determine a string that represents when you want the Splunk platform to perform the host name override. This string becomes the regular expression you supply later in the procedure. See the example later in this topic.
2. Review the [Configure a transforms.conf stanza with a host name override transform](#) and [Configure a props.conf stanza to reference a host name override transform](#) sections in this section to understand how stanza syntax for host name overrides works.
3. On a heavy forwarder where you want to do the host name overrides, open a text editor.
4. With that editor, open the `$SPLUNK_HOME/etc/system/local/transforms.conf` file for editing.
5. Add a stanza to this file that represents when the Splunk platform is to do the host name override.
6. Save the `transforms.conf` file and close it.

7. Open the `$SPLUNK_HOME/etc/system/local/props.conf` file for editing.
8. Add a stanza to this file that represents the default fields for which the host name override is to apply.
9. Save the `props.conf` file and close it.
10. Restart the heavy forwarder.

On Splunk Enterprise, you can perform this procedure on either the instance that ingests the data or on a heavy forwarder that sends data to the instance.

For more information about configuration files in general, see *About configuration files* in the *Splunk Enterprise Admin Manual*.

Configure a `transforms.conf` stanza with a host name override transform

The `transforms.conf` file controls where and how the Splunk platform transforms the incoming event data.

The host name override transformation stanza in `transforms.conf` uses the following syntax:

```
[<unique_stanza_name>]
REGEX = <your_regex>
FORMAT = host::$1
DEST_KEY = MetaData:Host
```

There are a few things to note in this stanza:

- Use the `<unique_stanza_name>` part of the syntax to refer to the transform from the `props.conf` configuration file. A best practice is for it to reflect that it involves a host value.
- `<your_regex>` is the regular expression that identifies where in the event you want to extract the host value and assign that value as the default field for that event.
- `FORMAT = host::$1` writes the `REGEX` value into the `host::` field.

Configure a `props.conf` stanza to reference the host name override transform

The `props.conf` file references the stanza in the `transforms.conf` file that performs the transformation:

```
[<spec>]
TRANSFORMS-<class> = <unique_stanza_name>
```

There are a few things to note in this stanza:

- `<spec>` can be any of these values:
 - ◆ `<sourcetype>`, the source type of an event.
 - ◆ `host::<host>`, where `<host>` is the host value for an event.
 - ◆ `source::<source>`, where `<source>` is the source value for an event.
- `<class>` is any unique identifier that you want to give to your transform.
- `<unique_stanza_name>` is the name of the stanza you created in `transforms.conf`.

Example of host name default field overriding

Given the following set of events from the `housesness.log` file. You want the Splunk platform to set the host default field for each event to the host name found within the event. The host is in the third position of each line in the log file. In this example, it's `fflanda`.

```
41602046:53 accepted fflanda
41602050:29 accepted rhallen
41602052:17 accepted fflanda
```

First, create a new stanza in the transforms.conf file and provide a regular expression that extracts the host value:

```
[housesness]
DEST_KEY = MetaData:Host
REGEX = \s(\w*)$
FORMAT = host::$1
```

Next, reference the transforms.conf stanza in a stanza in the props.conf configuration file. See the following example:

```
[source::.../housesness.log]
TRANSFORMS-rhallen=housesness
SHOULD_LINEMERGE = false
```

This example stanza has the additional setting SHOULD_LINEMERGE = false to break events at each new line.

The events then appear in search results like this:



Event ID	Timestamp	Log Line	host	sourcetype	source
8	6/22/09 4:44:44.000 PM	41602052:17 accepted fflanda	fflanda	housesness	./housesness.log
9	6/22/09 4:44:44.000 PM	41602050:29 accepted rhallen	rhallen	housesness	./housesness.log
10	6/22/09 4:44:44.000 PM	41602046:53 accepted fflanda	fflanda	housesness	./housesness.log

Change host values after indexing

At some point after indexing, you might notice that the host value for some of your events isn't correct. For example, you might be collecting Web proxy logs into a directory directly on your Splunk platform instance and you add that directory as an input without remembering to override the value of the host field, which results in the host value being the same as your Splunk platform instance.

If something like that happens, here are your options, from easiest to hardest. You can do all of these with a Splunk Cloud Platform instance:

- Delete and reindex the data. See Remove indexes and indexed data in the Splunk Enterprise *Managing Indexers and Clusters of Indexers* manual.
- Use a search to delete the specific events that have the incorrect host value and reindex those events. See Remove an index entirely in the Splunk Enterprise *Managing Indexers and Clusters of Indexers* manual.
- Tag the incorrect host values and use the tag to search. See Tag field-value pairs in Search in the Splunk Enterprise *Managing Indexers and Clusters of Indexers* manual.
- Set up a comma-separated values (CSV) lookup to look up the host, map it in the lookup file to a new field name, and use the new name in searches. See Introduction to lookup configuration in the Splunk Enterprise *Knowledge Manager Manual*.
- Create an alias for the host field to a new field such as `temp_host`, set up a CSV lookup to look up the correct host name using the name `temp_host`, and then have the lookup overwrite the original `host` with the new lookup value using the `OUTPUT` option when defining the lookup. See Create field aliases in Splunk Web and Introduction to lookup configuration in the Splunk Enterprise *Knowledge Manager Manual*.

Of these options, deleting and reindexing gives you the best performance and is the easiest to do. If you can't delete and reindex the data, then the last option provides the fastest alternative.

For more information about overriding the value of a host field, see [Override the value of the host field](#).

Configure source types

Why source types matter

The **source type** is one of the **default fields** that the Splunk platform assigns to all incoming data. It tells the platform what kind of data you have, so that it can format the data intelligently during indexing. Source types also let you categorize your data for easier searching.

Source types determine how incoming data is formatted

Because the source type controls how the Splunk platform formats incoming data, it is important that you assign the correct source type to your data. That way, the indexed version of the data (the **event data**) looks the way you want, with appropriate **timestamps** and **event** breaks. This facilitates easier searching of the data later.

Splunk software comes with a large number of predefined source types. When consuming data, the Splunk platform usually selects the correct source type automatically. If you have specialized data, you might need to manually select a different predefined source type. If your data is unusual, you might need to create a new source type with customized event processing settings. And if your data source contains heterogeneous data, you might need to assign the source type on a per-event, rather than a per-source, basis.

Like any other field, you can also use the source type field to search event data after the data has been indexed. You use it a lot in your searches since the source type is a key way to categorize your data.

Common source types

Any common data input format can be a source type. Most source types are log formats. For example, some common source types that the Splunk platform automatically recognizes include the following:

Source type	Description
access_combined	For NCSA combined log format HTTP Web server logs.
apache_error	For standard Apache Web server error logs.
cisco_syslog	For the standard syslog produced by Cisco network devices (including PIX firewalls, routers, and ACS), usually using remote syslog to a central log host.
websphere_core	A core file export from WebSphere.

For a complete list of predefined source types, see [List of pretrained source types](#) in this manual.

Configuring source types

There are two basic types of configuration you can do with source types:

- Assign source types explicitly to your incoming data.
- Create new source types, either from scratch or by modifying an existing source type.

Assign source types

In most cases, the Splunk platform determines the best source type for your data and automatically assigns it to incoming events. In some cases, however, you might need to explicitly assign a source type to your data. You usually do this when

you define the data input. For details on how to improve source type assignment, see the following topics:

- [Override automatic source type assignment](#)
- [Override source types on a per-event basis](#)
- [Configure rule-based source type recognition](#)
- [Create source types](#)
- [Rename source types](#)

For more information about how the Splunk platform assigns source types, see [How the Splunk platform assigns source types](#).

Create new source types

If none of the existing source types fits the needs of your data, create a new one.

Splunk Web lets you adjust source type settings to fit your data. In essence, it's a visual source type editor. See [Use the Set Source Type page](#).

If you use Splunk Cloud Platform, use Splunk Web or Apps to define source types. If you use Splunk Enterprise, use Apps. See [Create source types](#).

A source type name can only contain the letters a through z, the numerals 0 through 9, the colon (:) character, and the underscore (_) character.

Preview data to test and modify source types

Splunk Web lets you review the effects of applying a source type to an input. It lets you preview the resulting events without actually committing them to an index. You can also edit timestamp and event breaking settings interactively and then save the modifications as a new source type. For information on how data preview functions as a source type editor, see [Use the Set Source Type page](#).

Search on source types

`sourcetype` is the name of the source type search field. You can use the `sourcetype` field to find similar types of data from any source type. For example, you can search `sourcetype=weblogic_stdout` to find all of your WebLogic server events, even when WebLogic is logging from more than one domain, or host in Splunk terms.

How the Splunk platform assigns source types

The Splunk platform uses a variety of methods to assign source types to event data at index time. Both Splunk Cloud Platform and Splunk Enterprise perform these methods the same way. The difference is that, on Splunk Cloud Platform, you can only make changes to source type configurations by using Splunk Web or Apps to set `inputs.conf` and `props.conf`.

As the Splunk platform processes event data, it steps through these methods in a defined order of precedence. It starts with source type configurations that have been statically configured in the `inputs.conf` and `props.conf` configuration files, moves on to rule-based source type association, and then works through methods like automatic source type recognition and automatic source type learning. This range of methods enables you to configure how the Splunk platform applies source type values to specific kinds of events, while assigning source type values to other events automatically.

The following list shows how the Splunk platform goes about determining the source type for a data input. The Splunk platform starts with the first method and then descends through the others as necessary, until it can determine the source type.

- [Explicit source type specification based on the data input](#)
- [Explicit source type specification based on the data source](#)
- [Rule-based source type recognition](#)
- [Automatic source type matching](#)
- [Delayed rule-based source type association](#)
- [Automatic source type learning](#)

Explicit source type specification based on the data input

If the Splunk platform finds an explicit source type for the data input, it stops here.

If you use Splunk Cloud Platform, you configure this in [Splunk Web](#). If you use Splunk Enterprise, you configure this in either Splunk Web or in the inputs.conf configuration file. Here is the syntax for configuring the inputs.conf file to assign source types to a file input:

```
[monitor://<path>]
sourcetype=<sourcetype>
```

You can also assign a source type when defining an input in Splunk Web. For information on doing this for file inputs, see [Monitor files and directories in Splunk Enterprise with Splunk Web](#) in this manual. The process is similar for network or other types of inputs.

For more information, see [Specify source type for an input](#).

Explicit source type specification based on the data source

If the Splunk platform finds an explicit source type for the particular data source, it stops here.

If you use Splunk Enterprise, or you want to use a heavy forwarder to forward this data to Splunk Cloud Platform, you can configure this in the props.conf configuration file, using the following syntax:

```
[source::<source>]
sourcetype=<sourcetype>
```

For more information, see [Specify source type for a source](#).

Rule-based source type recognition

The Splunk platform looks next for any rules that you have created for source types.

If you use Splunk Enterprise, you can create source type classification rules in the props.conf file. If you use Splunk Cloud, you can use Apps or the UI. Use the following syntax:

```
[rule::<rule_name>]
sourcetype=<sourcetype>
MORE_THAN_[0-100] = <regex>
LESS_THAN_[0-100] = <regex>
```

For information about setting up source type recognition rules, see [Configure rule-based source type recognition](#).

Automatic source type matching

The Splunk platform next attempts to use automatic source type recognition to match similar-looking files and assign a source type.

The Splunk platform calculates signatures for patterns in the first few thousand lines of any file or network input stream. These signatures identify things like repeating word patterns, punctuation patterns, line length, and so on. When the Splunk platform calculates a signature, it compares it to its set of signatures for known, "pretrained" source types. If it identifies a match, it assigns that source type to the data.

See [List of pretrained source types](#) for a list of the source types that the Splunk platform can recognize by default.

Delayed rule-based source type association

If the Splunk platform hasn't identified a source type by now, it looks for any delayed rules.

This works like rule-based associations. If you use Splunk Enterprise, you can create a `delayedrule::` stanza in the `props.conf` file. This is a useful catch-all for source types, in case Splunk Enterprise missed any with intelligent matching.

A good use of delayed rule associations is for generic versions of very specific source types that were defined earlier with `rule::` in the rule-based step. For example, you could use `rule::` to catch event data with specific syslog source types, such as "sendmail syslog" or "cisco syslog" and then have `delayedrule::` apply the generic `syslog` source type to the remaining syslog event data.

Here is the syntax:

```
[delayedrule::$RULE_NAME]
sourcetype=$SOURCETYPE
MORE_THAN_[0-100] = $REGEX
LESS_THAN_[0-100] = $REGEX
```

For more information about setting up or removing delayed rules for source type recognition, see [Configure rule-based source type recognition](#).

Automatic source type learning

If the Splunk platform is unable to assign a source type for the event using the preceding methods, it creates a new source type for the event signature. The Splunk platform stores learned pattern information in the `sourcetypes.conf` configuration file.

Override automatic source type assignment

Splunk Enterprise attempts to assign a source type to your data automatically. You can specify what source type to assign. You can also configure the Splunk platform so that it assigns a source type based on either the data input or the data source.

For details on the precedence rules that the Splunk platform uses to assign source types to data, read [How the Splunk platform assigns source types](#) in [Why source types matter](#).

Overrides work only on file and directory monitoring inputs or files you upload. You can't override the source type on network inputs. Additionally, overrides affect only new data that arrives after you set up the override. To correct the source

types of events that are already indexed, create a tag for the source type instead. See Tag field-value pairs in Search in the *Knowledge Manager Manual*.

You can specify a source type for data based on its input and source.

Specify source type for an input

You can assign the source type for data coming from a specific input, such as `/var/log/`. If you use Splunk Cloud Platform, use Splunk Web to define source types. If you use Splunk Enterprise, define source types in Splunk Web or by editing the `inputs.conf` configuration file.

Be aware that assigning source type by input is not very granular. When you specify source type by input, the Splunk platform assigns the same source type to all data from an input, even if some of the data comes from different sources or hosts. To bypass automatic source type assignment in a more targeted manner, you can assign source types based on the source of the data, as described in the Specify source type for a source section.

Use Splunk Web

When you define a data input, you can set a source type value to be applied to all incoming data from that input. You can pick a source type from a list or enter your own source type value.

To select a source type for an input, change the source type settings for the data input type you want to add. For example, for file inputs, complete the following steps:

1. Click **Settings** in the upper right-hand corner of Splunk Web.
2. In the **Data** section of the Settings drop-down list, click **Data Inputs**.
3. Click **Files & Directories**.
4. Click **New** to add an input.
5. In the **Add Data** page, browse or enter the name of the file you want to monitor, then click **Next**.
6. In the **Set Source Type** page, click the **Sourcetype** drop-down list and choose from the list of pretrained source types. See [List of pretrained source types](#).
Splunk Web updates the page to show how the data looks when it receives the new source type.
7. If you want to make changes to the source type, use the **Event Breaks**, **Timestamp**, and **Advanced** tabs to modify settings and refresh the data preview. See [Assign the correct source types to your data](#) in this manual.
8. If you want to save the source type as a different name, click **Save As** ⓘ to open the **Save Sourcetype** dialog box to save the new source type. Otherwise, proceed to Step 10.
9. If you choose to save the source type, enter the name, description, category, and app that the source type will apply to. See [Save modifications as a new source type](#).
10. Click **Next** to set the source type for the data and proceed to the Input Settings page. See [Modify input settings](#).

Splunk Enterprise now assigns your selected source type to all events it indexes for that input.

Use the inputs.conf configuration file

When you configure an input in the `inputs.conf` configuration file on a Splunk Enterprise instance, you can specify a source type for the input. On a Splunk Cloud Platform instance, you can configure a universal forwarder on the machine that has the data you want to collect and forward it to the Splunk Cloud Platform instance.

Edit the `inputs.conf` file in `$SPLUNK_HOME/etc/system/local/` or in your own custom application directory in `$SPLUNK_HOME/etc/apps/`. For information on configuration files in general, see About configuration files in the *Admin*

Manual.

To specify a source type, include a `sourcetype` attribute within the stanza for the input. For example:

```
[tcp://:9995]
connection_host=dns
sourcetype=log4j
source=tcp:9995
```

This example sets the source type to `log4j` for any events coming from your TCP input on port 9995.

Do not put quotes around the attribute value. The correct format, for example, is `sourcetype=log4j`, not `sourcetype="log4j"`.

Specify source type for a source

Use the `props.conf` file to override automated source type matching and explicitly assign a single source type to all data coming from a specific source.

Edit `props.conf` in `$SPLUNK_HOME/etc/system/local/` or in your own custom application directory in `$SPLUNK_HOME/etc/apps/`. For information on configuration files in general, see About configuration files in the *Admin Manual*.

If you want to override a source type, you must configure the setting in `props.conf` on the forwarder where the input is configured.

To override source type assignment, add a stanza for your source to `props.conf`. In the stanza, identify the source path, using regular expression (regex) syntax for flexibility if necessary. Then specify the source type by including a `sourcetype` attribute. For example:

```
[source::.../var/log/anaconda.log(.\d+)?]
sourcetype=anaconda
```

This example sets the source type to `anaconda` for events from any sources containing the string `/var/log/anaconda.log` followed by any number of numeric characters.

Your stanza source path regular expressions, such as `[source::.../web/....log]`, must be as specific as possible. Avoid using a regex that ends in `....`. For example, do not do this:

```
[source::/home/fflanda/...]
sourcetype=mytype
```

This formatting is dangerous. It tells the Splunk platform to process any GZIP files in `/home/fflanda` as `mytype` files rather than GZIP files.

Instead, write using the following format:

```
[source::/home/fflanda/....log(.\d+)?]
sourcetype=mytype
```

Configure rule-based source type recognition

You can use rule-based source type recognition to expand the range of source types that Splunk software recognizes.

You must configure rule-based source type recognition using configuration files. If you use Splunk Cloud Platform, you must configure source type recognition using a heavy forwarder before you send the data to Splunk Cloud Platform. You might also need to file a support ticket to confirm that your Splunk Cloud Platform deployment recognizes the source type rules.

If you use Splunk Enterprise, create a `rule::` stanza in the `props.conf` configuration file that associates a specific source type with a set of qualifying criteria. When it consumes data, the Splunk platform assigns the specified source type to file inputs that meet the rule qualifications.

How rule-based source type recognition works

You can create two types of rules in the `props.conf` file: rules and delayed rules. The difference between the two is the point at which the Splunk platform checks them during the source typing process. As it processes each set of incoming data, the Splunk platform uses several methods to determine source types:

- After the Splunk platform checks for explicit source type definitions, it looks at any `rule::` stanzas that you defined in the `props.conf` file and tries to match source types to the data based on the classification rules specified in those stanzas.
- If the Splunk platform is unable to find a matching source type using the available `rule::` stanzas, it tries to use automatic source type matching, where it identifies patterns similar to source types it has learned in the past.
- If that method fails, the Splunk platform then checks any `delayedrule::` stanzas in the `props.conf` file and tries to match the data to source types using the rules in those stanzas.

For details on the precedence rules that the Splunk platform uses to assign source types to data, see [How the Splunk platform assigns source types](#).

You can configure the Splunk platform so that `rule::` stanzas contain classification rules for specialized source types, while `delayedrule::` stanzas contain classification rules for generic source types. That way, the Splunk platform applies the generic source types to broad ranges of events that aren't qualified for more specialized source types.

For example, you could use `rule::` stanzas to catch data with specific syslog source types, such as `sendmail_syslog` or `cisco_syslog`, and then configure a `delayedrule::` stanza to apply the generic `syslog` source type to any remaining syslog data.

Create source typing rules in `props.conf`

To set source typing rules, edit the `props.conf` configuration file in `$SPLUNK_HOME/etc/system/local/` or in your own custom application directory in `$SPLUNK_HOME/etc/apps/`. For information on configuration files in general, see *About configuration files* in the *Admin Manual*.

Create a rule by following these steps:

1. In `props.conf`, add a `rule::` or `delayedrule::` stanza. Provide a name for the rule in the stanza header.

```
[rule::<rule_name>] OR [delayedrule::<rule_name>]
```

2. Declare the source type in the body of the stanza.

```
[rule::<rule_name>] OR [delayedrule::<rule_name>]
sourcetype=<source_type>
```

3. After the source type declaration, set a numerical value in the `MORE_THAN` and `LESS_THAN` attributes, corresponding to the percentage of input lines that must contain the string specified by the regular expression. For example, `MORE_THAN_80` means at least 80% of the lines must contain the associated expression. `LESS_THAN_20` means that less than 20% of the lines can contain the associated expression.

You can set any number of `MORE_THAN` and `LESS_THAN` conditions in a rule. The rule's source type is assigned to a data file only if the data qualifies all the statements in the rule. For example, you can define a rule that assigns a specific source type to a file input only if more than 60% of the lines match one regular expression and less than 20% match another regular expression.

Follow this syntax to define the source type assignment rules:

```
[rule::<rule_name>] OR [delayedrule::<rule_name>]
sourcetype=<source_type>
MORE_THAN_[0-99] = <regex>
LESS_THAN_[1-100] = <regex>
```

Despite its nomenclature, the `MORE_THAN_` attribute actually means more than or equal to. Similarly the `LESS_THAN_` attribute means less than or equal to.

You can test regular expressions by using them in searches with the `rex search` command.

Examples

The following examples show ways to configure source type rules.

Postfix syslog files

```
# postfix_syslog sourcetype rule
[rule::postfix_syslog]
sourcetype = postfix_syslog
# If 80% of lines match this regex, then it must be this type
MORE_THAN_80=^w{3} +\d+ \d\d:\d\d:\d\d .* postfix(/\w+)?[\d+]:
```

Delayed rule for breakable text

```
# breaks text on ascii art and blank lines if more than 10% of lines have
# ascii art or blank lines, and less than 10% have timestamps
[delayedrule::breakable_text]
sourcetype = breakable_text
MORE_THAN_10 = (^(:---|==|\*\*\*|___|=+))|^\s*$
LESS_THAN_10 = [: ][012]?[0-9]:[0-5][0-9]
```

List of pretrained source types

Splunk software ships with built-in or pretrained **source types** that it uses to parse incoming data into events.

The Splunk platform can automatically recognize and assign many of these pretrained source types to incoming data. You can also manually assign pretrained source types that the Splunk platform doesn't recognize automatically. To assign source types manually, see [Override automatic source type assignment](#).

From a heavy or universal forwarder, you can also configure source types from the inputs.conf configuration file. If you use Splunk Enterprise, you can assign source types from either Splunk Web or from the inputs.conf file.

Use a pretrained source type if it matches your data, as the Splunk platform already knows how to properly index pretrained source types. If your data doesn't fit any pretrained source types, you can create your own source types. See [Create source types](#). The Splunk platform can also index virtually any format of data even without custom properties.

Automatically recognized source types

The following table shows automatically recognized source types:

Source type	Origin	Examples
access_combined	NCSA combined format http web server logs. Can be generated by Apache or other web servers.	10.1.1.43 - webdev [08/Aug/2005:13:18:16 -0700] "GET / HTTP/1.0" 200 0442 "-" "check_http/1.10 (nagios-plugins 1.4)"
access_combined_wcookie	NCSA combined format http web server logs. Can be generated by Apache or other web servers, with cookie field added at the end.	"66.249.66.102.1124471045570513" 59.92.110.121 - - [19/Aug/2005:10:04:00 -0700] "GET /themes/splunk_com/images/logo_splunk.png HTTP/1.1" 200 994 "http://www.splunk.org/index.php/docs" "Mozilla/5.0 (X11; U; Linux i686 en-US; rv:1.7.8) Gecko/20050524 Fedora/1.0.4-4 Firefox/1.0.4" "61.3.110.148.1124404439914689"
access_common	NCSA common format http web server logs. Can be generated by Apache or other web servers.	10.1.1.140 - - [16/May/2005:15:01:52 -0700] "GET /themes/ComBeta/images/bullet.png HTTP/1.1" 404 304
apache_error	Standard Apache web server error log	[Sun Aug 7 12:17:35 2005] [error] [client 10.1.1.015] File does not exist: /home/reba/public_html/images/bullet_image.gif
asterisk_cdr	Standard Asterisk IP PBX call detail record	"", "5106435249", "1234", "default", "" James Jesse "<5106435249>", "SIP/5249-1ce3", "", "VoiceMail", "u1234", "2005-05-26 15:19:25", "2005-05-26 15:19:25", "2005-05-26 15:19:42", 17, 17, "ANSWERED", "DOCUMENTATION"
asterisk_event	Standard Asterisk event log (management events)	Aug 24 14:08:05 asterisk[14287]: Manager 'randy' logged on from 127.0.0.1
asterisk_messages	Standard Asterisk messages log (errors and warnings)	Aug 24 14:48:27 WARNING[14287]: Channel 'Zap/1-1' sent into invalid extension 's' in context 'default', but no invalid handler
asterisk_queue	Standard Asterisk queue log	1124909007 NONE NONE NONE CONFIGRELOAD
cisco_syslog	Standard Cisco syslog produced by all Cisco network devices including PIX firewalls, routers, ACS, and so on. Usually through remote syslog to a central log host.	Sep 14 10:51:11 stage-test.splunk.com Aug 24 2005 00:08:49: %PIX-2-106001: Inbound TCP connection denied from IP_addr/port to IP_addr/port flags TCP_flags on interface int_name Inbound TCP connection denied from 144.1.10.222/9876 to 10.0.253.252/6161 flags SYN on interface outside
db2_diag	Standard IBM DB2 database administrative and error log	2005-07-01-14.08.15.304000-420 I27231H328 LEVEL: Event PID : 2120 TID : 4760 PROC : db2fmp.exe INSTANCE: DB2 NODE : 000 FUNCTION: DB2 UDB, Automatic Table Maintenance, db2HmonEvalStats, probe:900 STOP : Automatic Runstats: evaluation has finished on database TRADEDB

Source type	Origin	Examples
exim_main	Exim MTA mainlog	2005-08-19 09:02:43 1E69KN-0001u6-8E => support-notifications@splunk.com R=send_to_relay T=remote_smtp H=mail.int.splunk.com [10.2.1.10]
exim_reject	Exim reject log	2005-08-08 12:24:57 SMTP protocol violation: synchronization error (input sent without waiting for greeting): rejected connection from H=gate.int.splunk.com [10.2.1.254]
linux_messages_syslog	Standard Linux syslog, located at /var/log/messages on most platforms	Aug 19 10:04:28 db1 sshd(pam_unix)[15979]: session opened for user root by (uid=0)
linux_secure	Red Hat, Debian, and equivalent distributions Linux authentication log	Aug 18 16:19:27 db1 sshd[29330]: Accepted publickey for root from ::ffff:10.2.1.5 port 40892 ssh2
log4j	Log4j standard output produced by any J2EE server using log4j	2005-03-07 16:44:03,110 53223013 [PoolThread-0] INFO [STDOUT] got some property...
mysqld_error	Standard MySQL error log	050818 16:19:29 InnoDB: Started; log sequence number 0 43644 /usr/libexec/mysqld: ready for connections. Version: '4.1.10a-log' socket: '/var/lib/mysql/mysql.sock' port: 3306 Source distribution
mysqld	Standard MySQL query log that also matches the MySQL binary log following conversion to text	53 Query SELECT xar_dd_itemid, xar_dd_propid, xar_dd_value FROM xar_dynamic_data WHERE xar_dd_propid IN (27) AND xar_dd_itemid = 2
postfix_syslog	Standard Postfix MTA log reported through the *nix syslog facility	Mar 1 00:01:43 avas postfix/smtpd[1822]: 0141A61A83: client=host76-117.pool80180.interbusiness.it[80.180.117.76]
sendmail_syslog	Standard Sendmail MTA log reported through the *nix syslog facility	Aug 6 04:03:32 nmrjl00 sendmail[5200]: q64F01Vr001110: to=root, ctladdr=root (0/0), delay=00:00:01, xdelay=00:00:00, mailer=relay, min=00026, relay=[101.0.0.1] [101.0.0.1], dsn=2.0.0, stat=Sent (v00F3HmX004301 Message accepted for delivery)
sugarcrm_log4php	Standard Sugarcrm activity log reported using the log4php utility	Fri Aug 5 12:39:55 2005,244 [28666] FATAL layout_utils - Unable to load the application list language file for the selected language(en_us) or the default language(en_us)
weblogic_stdout	Weblogic server log in the standard native BEA format	####<Sep 26, 2005 7:27:24 PM MDT> <Warning> <WebLogicServer> <bea03> <asiAdminServer> <ListenThread.Default> <<WLS Kernel>> <> <BEA-000372> <HostName: 0.0.0.0, maps to multiple IP addresses:169.254.25.129,169.254.193.219>
websphere_activity	Websphere activity log, also often referred to as the service log	----- ComponentId: Application Server ProcessId: 2580 ThreadId: 0000001c ThreadName: Non-deferrable Alarm : 3 SourceId: com.ibm.ws.channel.framework.impl.WSChannelFrameworkImpl ClassName: MethodName: Manufacturer: IBM Product: WebSphere Version: Platform 6.0 [BASE 6.0.1.0 o0510.18] ServerName: nd6Cell101\was1Node01\TradeServer1 TimeStamp: 2005-07-01 13:04:55.187000000 UnitOfWork: Severity: 3 Category: AUDIT PrimaryMessage: CHFW0020I: The Transport Channel Service has stopped th Chain labeled SOAPAcceptorChain2 ExtendedMessage: -----
websphere_core	Core file export from WebSphere	NULL----- OSECTION TITLE subcomponent dump routine NULL===== 1TISIGIN signal 0 received 1TIDATETIME Date: 2005/08/02 at 10:19:24 1TIFILENAME Javacore filename: /kmbcc/javacore95014.1122945564.txt NULL ----- OSECTION XHPI

Source type	Origin	Examples
		subcomponent dump routine NULL ===== 1XHTIME Tue Aug 2 10:19:24 20051XHSIGRECV SIGNONE received at 0x0 in <unknown> Processing terminated. 1XHFULLVERSION J2RE 1.3.1 IBM AIX build ca131-20031105 NULL
websphere_trlog_syserr	Standard Websphere system error log in the IBM native trlog format	[7/1/05 13:41:00:516 PDT] 000003ae SystemErr R at com.ibm.ws.http.channel.inbound.impl.HttpICLReadCallback.complete (HttpICLReadCallback.java(Compiled Code)) (truncated)
websphere_trlog_sysout	Standard Websphere system out log in the IBM native trlog format. Similar to the log4j server log for Resin and Jboss. Sample format as the system error log but contains lower severity and informational events.	[7/1/05 13:44:28:172 PDT] 0000082d SystemOut O Fri Jul 01 13:44:28 PDT 2005 TradeStreamerMDB: 100 Trade stock prices updated: Current Statistics Total update Quote Price message count = 4400 Time to receive stock update alerts messages (in seconds): min: -0.013 max: 527.347 avg: 1.0365270454545454 The current price update is: Update Stock price for s:393 old price = 15.47 new price = 21.50
windows_snare_syslog	Standard windows event log reported through a third-party Intersect Alliance Snare agent to remote syslog on a *nix server	0050818050818 Sep 14 10:49:46 stage-test.splunk.com Windows_Host MSWinEventLog 0 Security 3030 Day Aug 24 00:16:29 2005 560 Security admin4 User Success Audit Test_Host Object Open: Object Server: Security Object Type: File Object Name: C:\Directory\secrets1.doc New Handle ID: 1220 Operation ID: {0,117792} Process ID: 924 Primary User Name: admin4 Primary Domain: FLAME Primary Logon ID: (0x0,0x8F9F) Client User Name: Client Domain: - Client Logon ID: - Accesses SYNCHRONIZE ReadData (or ListDirectory) Privileges -Sep

Special source types

The following table shows the special source types:

Source type	Origin	Examples
known_binary	The file name matches a pattern generally known as that of a binary file, not a log file	MP3 files, images, .rdf files, .dat files, and other obvious non-text files

Pretrained source types

These following table shows pretrained source types, including both those that are automatically recognized and those that are not:

Category	Source types
Application servers	log4j, log4php, weblogic_stdout, websphere_activity, websphere_core, websphere_trlog, catalina, ruby_on_rails
Databases	db2_diag, mysqld, mysqld_error, mysqld_bin, mysql_slow
E-mail	exim_main, exim_reject, postfix_syslog, sendmail_syslog, procmail
Operating systems	linux_messages_syslog, linux_secure, linux_audit, linux_bootlog, anaconda, anaconda_syslog, osx_asl, osx_crashreporter, osx_crash_log, osx_install, osx_secure, osx_daily, osx_weekly, osx_monthly, osx_window_server, windows_snare_syslog, dmesg, ftp, ssl_error, syslog, sar, rpmpkgs
Metrics	collectd_http, metrics_csv, statsd
Network	novell_groupwise, tcp
Printers	cups_access, cups_error, spooler

Category	Source types
Routers and firewalls	cisco_cdr, cisco:asa, cisco_syslog, clavister
VoIP	asterisk_cdr, asterisk_event, asterisk_messages, asterisk_queue
Web servers	access_combined, access_combined_wcookie, access_common, apache_error, iis*
Splunk software	splunk_com_php_error, splunkd, splunkd_crash_log, splunkd_misc, splunkd_stderr, splunk-blocksignature, splunk_directory_monitor, splunk_directory_monitor_misc, splunk_search_history, splunkd_remote_searches, splunkd_access, splunkd_ui_access, splunk_web_access, splunk_web_service, splunkd_conf*, django_access, splunk_help, mongod
Non-log files	csv*, psv*, tsv*, _json*, json_no_timestamp, fs_notification, exchange*, generic_single_line
Miscellaneous	snort, splunk_disk_objects*, splunk_resource_usage*, kvstore*

The source types marked with an asterisk (*) use the INDEXED_EXTRactions attribute, which sets other attributes in props.conf to specific defaults and requires special handling to forward to another Splunk platform instance. See Forward fields extracted from structured data files.

Learn a source type configuration

To find out what configuration information the Splunk platform uses to index a given source type, you can use the `bttool` utility to show the properties on your forwarder. If you use Splunk Enterprise, you can do this on your Splunk Enterprise instance.

For more information on using `bttool`, refer to Use `bttool` to troubleshoot configurations in the *Troubleshooting Manual*.

The following example shows how to list out the configuration for the `tcp` source type:

```
$ ./splunk bttool props list tcp
[tcp]
BREAK_ONLY_BEFORE = (=\\+)+
BREAK_ONLY_BEFORE_DATE = True
CHARSET = UTF-8
DATETIME_CONFIG = /etc/datetime.xml
KV_MODE = none
LEARN_SOURCETYPE = true
MAX_DAYS_AGO = 2000
MAX_DAYS_HENCE = 2
MAX_DIFF_SECS_AGO = 3600
MAX_DIFF_SECS_HENCE = 604800
MAX_EVENTS = 256
MAX_TIMESTAMP_LOOKAHEAD = 128
MUST_BREAK_AFTER =
MUST_NOT_BREAK_AFTER =
MUST_NOT_BREAK_BEFORE =
REPORT-tcp = tcpdump-endpoints, colon-kv
SEGMENTATION = inner
SEGMENTATION-all = full
SEGMENTATION-inner = inner
SEGMENTATION-outer = foo
SEGMENTATION-raw = none
SEGMENTATION-standard = standard
SHOULD_LINEMERGE = True
TRANSFORMS =
TRANSFORMS-baindex = banner-index
TRANSFORMS-dlindex = download-index
```

```
TRUNCATE = 10000
maxDist = 100
pulldown_type = true
```

Override source types on a per-event basis

You can override source types on a per-event basis on the Splunk platform by using a heavy forwarder to assign the events to a new source type and sending those events to Splunk Cloud Platform. On Splunk Enterprise, you can override source types directly on the instance itself.

This source type assignment happens at parse-time, after the platform has made its initial source type assignment. For more information about this process, see [How the Splunk platform assigns source types](#) in [Why source types matter](#).

Since this type of override occurs at parse-time, the override works only on an indexer or heavy forwarder. It doesn't work on a universal forwarder or directly on Splunk Cloud Platform. See Configuration parameters and the data pipeline in the *Admin Manual* for more information on what configurations are available at different points in the input, parsing, and indexing processes.

To configure per-event overrides, use the transforms.conf and props.conf configuration files in tandem to specify the events that must use a new source type, along with the source type that the events must use.

For information about configuring basic source type overrides for event data that comes from specific inputs or that has a particular source, see [Override automatic source type assignment](#).

Configuration

To configure per-event overrides, create one stanza in the transforms.conf file and another in the props.conf file. Edit these files in the \$SPLUNK_HOME/etc/system/local/ directory or in your own custom application directory at \$SPLUNK_HOME/etc/apps/. For more information about configuration files in general, see About configuration files in the *Admin Manual*.

Edit the transforms.conf file

1. Open \$SPLUNK_HOME/etc/system/local/transforms.conf file for editing.
2. Create a stanza in transforms.conf that follows this syntax:

```
[<unique_stanza_name>]
REGEX = <your_regex>
FORMAT = sourcetype::<your_custom_sourcetype_value>
DEST_KEY = MetaData:Sourcetype
```

3. Save the file and close it.

In the file, the settings have the following meaning:

- `<unique_stanza_name>` means that it involves a source type. You'll use this name later in the props.conf stanza.
- `<your_regex>` is a regular expression that identifies the events that you want to apply a custom source type to, such as events that carry a particular host name or other field value.
- `<your_custom_sourcetype_value>` is the source type that you want to apply to the events that `<your_regex>` selected.

You can test regular expressions by using them in searches with the `rex` search command. See `rex` in the *Search Reference*.

Edit the props.conf file

1. Open `$SPLUNK_HOME/etc/system/local/props.conf`.
2. Create a stanza in the props.conf file that references the stanza that you specified in the transforms.conf file:

```
[<spec>]
TRANSFORMS-<class> = <unique_stanza_name>
```

Refer to the following table for the meanings of each placeholder variable within this stanza:

Placeholder variable	Description
spec	Can be set to the following options: <ul style="list-style-type: none">◆ <code><sourcetype></code>, or the source type of an event◆ <code>host::<host></code>, where <code><host></code> is the host value for an event◆ <code>source::<source></code>, where <code><source></code> is the source value for an event
<class>	Any unique identifier that you want to give to your transform
<unique_stanza_name>	The name of the stanza you created in transforms.conf

3. Save the file and close it.
4. Restart the Splunk platform instance.

Example: Assign a source type to events from a single input but different hosts

Suppose that you have a shared UDP input, UDP514. Your Splunk platform instance indexes a wide range of data from a number of hosts through this input. You find that you need to apply a particular source type called `my_log` to data originating from three specific hosts, `host1`, `host2`, and `host3`, reaching your instance through the UDP514 input.

To start, you can use the regular expression that Splunk software typically uses to extract the host field for syslog events. You can find it in `$SPLUNK_HOME/etc/system/default/transforms.conf`:

```
[syslog-host]
REGEX = :d\d\s+(?:d\d\s+(?:user|daemon|local.?)\.\w\s+)*\[?(?(\w[\w\.-]{2,})\])?\s
FORMAT = host::$1
DEST_KEY = MetaData:Host
```

You can modify this regular expression to match events from only the host names you want. In this example, the host names are `host1`, `host2`, and `host3`:

```
REGEX = :d\d\s+(?:d\d\s+(?:user|daemon|local.?)\.\w\s+)*\[?(host1|host2|host3) [\w\.-]*\]\s
```

Now you can use the modified regular expression in a transform that applies the `my_log` source type to events that come from those three hosts:

```
[set_sourcetype_my_log_for_some_hosts]
REGEX = :d\d\s+(?:d\d\s+(?:user|daemon|local.?)\.\w\s+)*\[?(host1|host2|host3) [\w\.-]*\]\s
FORMAT = sourcetype::my_log
DEST_KEY = MetaData:Sourcetype
```

Then you can specify that transform in a props.conf stanza that identifies the specific input for the events:

```
[source::udp:514]
TRANSFORMS-changesourcetype = set_sourcetype_my_log_for_some_hosts
```

Create source types

You can create new source types on the Splunk platform in several ways:

- Use the Set Source Type page in Splunk Web as part of adding the data.
- Create a source type in the Source types management page, as described in [Add Source Type](#).
- Edit the props.conf configuration file. This option isn't available on Splunk Cloud Platform unless you define the source types on a universal forwarder and send them to Splunk Cloud Platform.

Although you can configure individual forwarders to create source types by editing the configuration files that reside on the forwarders, a best practice for creating source types is to use Splunk Web to guarantee that source types are consistent across your Splunk platform deployment.

Set the source type as part of creating a data input in Splunk Web

The Set Source Type page in Splunk Web lets you view the effects of applying a source type to your data. It also lets you make adjustments to the source type settings as necessary. You can save your changes as a new source type, which you can then assign to data inputs.

The page lets you make the most common types of adjustments to **timestamps** and **event** breaks. For other modifications, it lets you edit the underlying props.conf file directly. As you change settings, you can immediately see how the changes affect the event data.

The page appears only when you specify or upload a single file. It doesn't appear when you specify any other type of data source.

To learn more about the Set Source Type page and how to assign source types to your data, see [Assign the correct source types to your data](#).

You can also use the Source types management page to create a new source type. See [Add Source Type](#).

Edit the props.conf configuration file to create a source type

If you use Splunk Enterprise, you can create a new source type by editing the props.conf configuration file and adding a new source type stanza. For detailed information on the props.conf file, read the props.conf specification in the Splunk Enterprise *Admin Manual*. For information on configuration files in general, see About configuration files in the the Splunk Enterprise *Admin Manual*.

The following entry is an example of an entry in the props.conf file. This entry defines the `access_combined` source type and then assigns that source type to files that match the specified source. You can configure multiple files or directories in a source by using a regular expression.

```
[access_combined]
pulldown_type = true
maxDist = 28
MAX_TIMESTAMP_LOOKAHEAD = 128
REPORT-access = access-extractions
SHOULD_LINEMERGE = False
TIME_PREFIX = \[
category = Web
description = National Center for Supercomputing Applications (NCSA) combined fo
rmat HTTP web server logs (can be generated by apache or other web servers)
```

```
[source::/opt/weblogs/apache.log]
sourcetype = access_combined
```

To edit the props.conf file, follow these steps:

1. On the machine where you want to create a source type, create the \$SPLUNK_HOME/etc/system/local/props.conf file if it doesn't already exist.

You might need to create the local directory. If you use an app, go to the app in the \$SPLUNK_HOME/etc/apps directory.
2. Using a text editor, open the the props.conf file in \$SPLUNK_HOME/etc/system/local directory.
3. Add a stanza for the new source type and specify any settings that Splunk software is to use when handling the source type.

```
[my_sourcetype]
setting1 = value
setting2 = value
```

See the props.conf specification in the Splunk Enterprise Admin Manual for a list of settings.

4. (Optional) If you know the name of the file to which the source type is to be applied, specify them in the [source::<source>] stanza:

```
[my_sourcetype]
setting1 = value
setting2 = value
<br>
[source::.../my/logfile.log]
sourcetype = my_sourcetype
```

5. Save the props.conf file.
6. Restart Splunk Enterprise. The new source types take effect after the restart completes.

Specify event breaks and time stamps

When you create a source type, there are some important settings to specify:

- Event breaks: To learn how to use the props.conf file to specify event breaks, see [Configure event line breaking](#).
- Timestamps: To learn how to use the props.conf file to specify timestamps, see [Configure timestamp recognition](#), as well as other topics in the Configure timestamps chapter of this manual.

There are also a number of additional settings that you can configure for event breaks and time stamps. See the props.conf specification in the Splunk Enterprise *Admin Manual* for more information.

Manage source types

Create, edit, and delete source types on the Source Types page. To get to the Source Types page in Splunk Web, go to **Settings > Source types**. While this page and the Set Source Type page have similar names, the pages offer different functions.

The Source Types page displays all source types that have been configured on a Splunk Enterprise instance. It shows the default source types provided by your deployment and any source types that you added.

Sort source types

By default, the Source Types page sorts source types alphabetically. You can change how the page sorts by clicking the header bar for the Name, Category, and App columns.

Each header bar, except for Actions, acts as a toggle. Click once to sort in ascending order and click again to sort in descending order.

Filter source types

You can filter the number of source types you see on the Source Type page.

Filter by commonality

To see only the most common source types, click the **Show only popular** checkbox along the top of the page. Popular source types are the most common source types. They have a `pulldown_type` source type field value of 1. When the **Show only popular** checkbox is not selected, the page shows all source types that are defined on the instance.

Filter by category

To see only source types that belong to a certain category, click the **Category** drop-down list and select the category you want. Only source types that belong to that category will display. To see all source types again, select **All** from the Category drop-down list.

Filter by application context

To see only source types that belong in a certain application context, click the **App** drop-down list and select the application context that the source type applies to. Only source types that apply to that application context will display. To see all source types again, select **All** from the App drop-down list.

Filter by string

To see only source types whose names contain a certain string, type that string in the **Filter** text box next to the App drop-down list, then press **Enter**. Only source types whose names or descriptions match what you have typed in the Filter box display. To see all source types again, click the **x** button on the right side of the Filter text box.

By default, the Source Types management page shows up to 20 source types on a page. If you want to see more or less, click **20 per page** on the right side of the page and select the number of source types you want to see. Choices are 10, 20, 50, or 100 listings on a page.

Modify source types

To modify a source type, click its name in the list, or click **Edit** in the **Actions** column. The Edit Source type page appears.

The **Edit Source Type** dialog box lets you change the configuration of a source type. You can change the following fields:

Field	Description
Description	The description of the source type.
Destination app	The application context that the source type applies to. You cannot change the app destination for source types that come with Splunk software.

CategoryThe category that the source type is a member of. Click the button to select from the list of categories and choose the one you want. When you save, the source type appears in the category you selected.

The Log to Metrics source type category is used specifically for converting log events to metric data points. See About the Log to Metrics source type category later in this topic.

Indexed ExtractionsA format for extracting fields at index time from files with structured data. Select the type of indexed extraction that best represents the contents of the file:

- none: The file does not contain structured data.
- json: The file is in JavaScript Object Notation (JSON) format.
- csv: The file has comma-separated values.
- tsv: The file has tab-separated values.
- psv: The file has pipe (|) separated values.
- w3c: The file conforms to the World Wide Web Consortium (W3C) logging format.
- field_extraction: The file contains unstructured events with fields in `<field>=<value>` format.

TimestampHow timestamps are determined for events from the source file. See the Timestamp section later in this topic.**Advanced**Shows all of the configurations for the source type in key/value format. See the Advanced section later in this topic.

Timestamp

The Timestamp section of the dialog box controls how timestamps are determined for events from the source file. You can choose the following options:

- Auto: Use default logic to extract timestamp from event.
- Current Time: Use current system time.
- Advanced: Use advanced internal logic to extract timestamp from event.

The following advanced configurations are available when you select **Advanced** in the Timestamp Extraction section:

- Time zone: Specify the time zone to be assigned to the timestamps.
- Timestamp format: Specify the format of the timestamp in the source event. The available formats come from the properties of the `strptime()` programming function.

For example, suppose the source file contains logs with timestamps in this format:

```
6 Jun 2015 18:35:05
```

Then, you might specify the following in the **Timestamp format** field:

```
%d %b %Y %H:%M:%S
```

Consider the following example:

```
Tue Jun 4 2:55:18 PM 2015
```

If the source file contains logs with timestamps in that format, specify the following in the **Timestamp format** field:

```
%a %b %d %I:%M:%S %p %Y
```

For a list of the strings that you can use to define the timestamp format, see `strptime(3)` at <https://linux.die.net/man/3/strptime> on the die.net Linux site.

Timestamp prefix: A regular expression that represents the characters that come before a timestamp. When Splunk Enterprise sees this set of characters in an event, it expects a timestamp to occur after that.

Lookahead: Specifies the maximum number of characters to scan into an event for the timestamp. If you specify a regular expression in the Timestamp prefix field, it looks no more than the number of characters specified past the string that the regular expression represents for the timestamp.

Advanced

The Advanced section shows you all of the configurations for the source type in key/value format. This represents what is in the `props.conf` configuration file that defines the source type. You can edit each setting directly, or add and delete settings.

Use the Advanced section with care. Adding or changing values here can cause data to be incorrectly indexed.

To delete settings, click the **x** on the right side of each setting. To add an entry, click **New setting** at the bottom of the dialog box to expose a key/value pair of fields. Enter the key name in the **Name** field and its value in the **Value** field.

About the Log to Metrics source type category

The **Log to Metrics** source type category is used for the ingest-time conversion of logs to metric data points. If you select it, a **Metrics** tab appears. For more information about this source type and the fields in the Metrics tab, see *Convert event logs to metric data points in Metrics*.

Add a source type

To create a new source type:

1. Click **New Source Type**. The Create Source Type dialog box opens.
2. Specify a name for the new source type. Source type names do not support the following characters: `<`, `>`, `?`, `#`, or `&`.
3. Follow the same steps used to modify an existing source type to configure your new source type. See [Modify source types](#) earlier in this topic.
4. Click **Save**.

Delete a source type

To delete a source type, click the **Delete** link in the **Actions** column for the source type that you want to delete. You can only delete a source type in Cloud if you are a Victoria Experience customer.

If you have a Splunk Cloud Platform instance and are unable to delete one of your user or app created source types, please contact Splunk Support for assistance. You cannot delete the source types built-in to Splunk Enterprise.

When you delete a source type, the following dialog appears:

A dialog box titled "Delete Source Type" with a close button (X) in the top right corner. The main text asks, "Are you sure you want to delete the source type named *earthquakes.csv*?". At the bottom, there are two buttons: "Cancel" and "Delete".

Delete Source Type

Are you sure you want to delete the source type named *earthquakes.csv*?

Cancel Delete

Deleting a source type has significant consequences, especially if the source type is already in use:

- Data can be indexed incorrectly after you delete the source type. Making the data searchable in the way you want later can take a lot of effort. Many apps and add-ons use source types to look for data, and data indexed under a missing source type is data those apps and add-ons do not see.
- Any configurations that the source type uses, such as field extractions, index time filtering, and timestamp formats, are irretrievably lost.
- You cannot undo a source type deletion. The only options available in this case are to restore the `props.conf` file that defines the source type from a backup, or recreate the source type manually.

If you are sure you want to delete the source type, click **Delete**.

The dialog box closes, and Splunk Web returns you to the Source Types management page.

Rename source types at search time

You might encounter situations where you want to rename a source type. For example, say you accidentally assigned an input to the wrong source type. Or you realize that two differently named source types actually need to be handled the same way at search time.

If you use Splunk Enterprise, you can add the `rename` setting in the `props.conf` configuration file to assign events to a new source type at search time. If you need to search on it, Splunk Enterprise moves the original source type to a separate field, called `_sourcetype`.

On Splunk Cloud Platform, you must open a support ticket to rename source types, as the `props.conf` file is not available for editing on a Splunk Cloud Platform instance and using a heavy forwarder is not possible as renaming source types is only applicable on data that you have already indexed.

The indexed events still contain the original source type name. The renaming of source types occurs only at search time. Also, renaming the source type does only that. It doesn't fix problems with the indexed format of your event data that were caused by assigning the wrong source type in the first place.

To rename the source type, add the `rename` setting to your source type stanza in the `props.conf` file:

```
rename = <string>
```

Source type names do not support the following characters: `,` `>`, `?`, `#`, and `&`

For example, say you're using the source type `cheese_shop` for your application server. Then you accidentally index a bunch of data as source type `whoops`. You can rename `whoops` to `cheese_shop` with the following stanza in the `props.conf` file:

```
[whoops]
rename=cheese_shop
```

Now, a search on `cheese_shop` returns all the `whoops` events as well as any events that had the `cheese_shop` source type:

```
sourcetype=cheese_shop
```

If you ever need to single out the `whoops` events, you can use `_sourcetype` in your search:

```
_sourcetype=whoops
```

Data from a renamed source type uses only the search-time configuration for the target source type, in this example it is `cheese_shop`. The Splunk platform ignores any field extractions for the original source type.

Manage event segmentation

About event segmentation

Event segmentation breaks events up into searchable **segments** at **index** time, and again at **search** time. Segments can be classified as major or minor. Minor segments are breaks within major segments. For example, the IP address `192.0.2.223` is a major segment. But this major segment can be broken down into minor segments, such as `192` or `0`, as well as groups of minor segments like `192.0.2`.

You can define how detailed the event segmentation is. This is important because index-time segmentation affects indexing and search speed, storage size, and the ability to use typeahead functionality, where Splunk Web provides items that match text you type into the Search bar. Search-time segmentation, on the other hand, affects search speed and the ability to create searches by selecting items from the results displayed in Splunk Web.

For more information about the distinction between index time and search time, see Index time versus search time in the *Managing Indexers and Clusters of Indexers* manual.

You can assign segmentation to specific categories of events in `props.conf`, as described in [Set the segmentation for event data](#).

If you use Splunk Cloud Platform, configure index-time segmentation on heavy forwarder machines. You must file a Support ticket to configure search-time segmentation.

If you use Splunk Enterprise, configure index-time segmentation on the indexer or heavy forwarder machines, and configure search-time segmentation on the search head.

Types of event segmentation

There are three main types, or levels, of segmentation, which you can configure either at index or search time. You can also disable segmentation. The `segmenters.conf` configuration file, located in `$SPLUNK_HOME/etc/system/default`, defines all available segmentation types. By default, index-time segmentation is set to the `indexing` type, which is a combination of inner and outer segmentation. Search-time segmentation is set to full segmentation.

Inner segmentation

Inner segmentation breaks events down into the smallest minor segments possible. For example, when an IP address such as `192.0.2.223` goes through inner segmentation, it is broken down into `192`, `0`, `2`, and `223`. Setting inner segmentation at index time leads to faster indexing and searching and reduced disk usage. However, it restricts the typeahead functionality, so that a user can only type ahead at the minor segment level.

Outer segmentation

Outer segmentation is the opposite of inner segmentation. Under outer segmentation, the Splunk platform only indexes major segments. For example, the IP address `192.0.2.223` gets indexed as `192.0.2.223`, which means that you cannot search on individual pieces of the phrase. You can still use wildcards, however, to search for pieces of a phrase. For example, you can search for `192.0*` and you will get any events that have IP addresses that start with `192.0`. Also, outer segmentation disables the ability to click on different segments of search results, such as the `192.0` segment of the same IP address. Outer segmentation tends to be marginally more efficient than full segmentation, while inner segmentation tends to be much more efficient.

Full segmentation

Full segmentation is a combination of inner and outer segmentation. Under full segmentation, the IP address is indexed both as a major segment and as a variety of minor segments, including minor segment combinations like `192.0` and `192.0.2`. This is the least efficient indexing option, but it provides the most versatility in terms of searching.

No segmentation

The most space-efficient segmentation setting is to disable segmentation completely. This has significant implications for search, however. By disabling segmentation, you restrict searches to indexed fields, such as time, source, host, and source type. Searches for keywords will return no results. You must pipe your searches through the search command to further restrict results. See search in the *Search Reference*. Use this setting only if you do not need any advanced search capability.

Configure segmentation types

`segmenters.conf` defines segmentation types. You can define custom segmentation types, if necessary.

For information on the types of segmentation available by default, see the `segmenters.conf` file in `$SPLUNK_HOME/etc/system/default`.

Do not modify the default file. If you want to make changes to the existing segmentation stanzas or create new ones altogether, you can specify the settings you want to change in a file in the `$SPLUNK_HOME/etc/system/local/` directory or to a custom app directory in `$SPLUNK_HOME/etc/apps/`.

Set segmentation types for specific hosts, sources, or source types

You can configure index-time and search-time segmentation to apply to specific hosts, sources, or source types. If you run searches that involve a particular source type on a regular basis, you can use this capability to improve the performance of those searches. Similarly, if you typically index a large number of `syslog` events, you can use this feature to help decrease the overall disk space that those events take up.

For details about how to apply segmentation types to specific event categories, see [Set the segmentation for event data](#).

Set the segmentation for event data

By default, Splunk software segments events during indexing to allow for the most flexible searching. There are numerous types of segmentation available, and you can create others if necessary. The type of segmentation that you employ affects indexing speed, search speed, and the amount of disk space the indexes occupy. To learn more about segmentation and the trade-offs between the various types of segmentation, refer to ["About segmentation"](#).

Splunk software can also segment events at search time. You can set search-time segmentation in Splunk Web, as described in ["Set search-time segmentation in Splunk Web"](#).

If you know how you want to search for or process events from a specific host, source, or source type, you can configure index-time segmentation for that specific type of event. You can also configure search-time segmentation options for specific types of events.

Specify segmentation in props.conf

Specify segmentation for events of particular hosts, sources, or source types by assigning segmentation types to the appropriate stanzas in `props.conf`. In the stanzas, you assign segmentation types, or "rules", that have been defined in `segmenters.conf`. These can either be predefined types (such as `inner`, `outer`, or `full`), or custom types that you've defined. For more information on defining custom types, read ["Configure segmentation types"](#).

The attribute you configure in `props.conf` to use these types depends on whether you're configuring index-time or search-time segmentation:

- For index-time segmentation, use the `SEGMENTATION` attribute.
- For search-time segmentation, use the `SEGMENTATION-<segment_selection>` attribute.

You can define either one of the attributes or both together in the stanza.

Add your stanza to `$SPLUNK_HOME/etc/system/local/props.conf`.

Index-time segmentation

The `SEGMENTATION` attribute determines the segmentation type used at index time. Here's the syntax:

```
[<spec>]
SEGMENTATION = <seg_rule>
```

[<spec>] can be:

- `<sourcetype>`: A source type in your event data.
- `host::<host>`: A host value in your event data.
- `source::<source>`: A source of your event data.

```
SEGMENTATION = <seg_rule>
```

- This specifies the type of segmentation to use at index time for [<spec>] events.
- `<seg_rule>`
 - ◆ A segmentation type, or "rule", defined in `segmenters.conf`
 - ◆ Common settings are `inner`, `outer`, `none`, and `full`, but the default file contains other predefined segmentation rules as well.
 - ◆ Create your own custom rule by editing `$SPLUNK_HOME/etc/system/local/segmenters.conf`, as described in ["Configure segmentation types"](#).

Search-time segmentation

The `SEGMENTATION-<segment_selection>` attribute helps determine the segmentation type used at search time. Here's the syntax:

```
[<spec>]
SEGMENTATION-<segment_selection> = <seg_rule>
```

[<spec>] can be:

- `<sourcetype>`: A source type in your event data.

- `host::<host>`: A host value in your event data.
- `source::<source>`: A source of your event data.

`SEGMENTATION-<segment_selection> = <seg_rule>`

- This specifies the type of segmentation to use at search time in Splunk Web for `[<spec>]` events.
- `<segment_selection>` can be one of the following: `full`, `inner`, `outer`, or `raw`.
 - ◆ These four values are the set of options displayed in the **Event segmentation** dropdown box in the **Results display options** panel, invoked from **Options** directly above search results in Splunk Web.
 - ◆ Note that these values are just the set of available Splunk Web dropdown options. You use this attribute to specify the actual segmentation type that the option invokes, which might not be of the same name as the dropdown option itself. For example, you could even define the "inner" dropdown option to invoke the "outer" segmentation type, not that you'd likely want to.
 - ◆ By mapping the dropdown option to a `<seg_rule>`, a user can later specify the option when looking at search results to set search-time segmentation, as described in ["Set search-time segmentation in Splunk Web"](#).
- `<seg_rule>`
 - ◆ A segmentation type, or "rule", defined in `segmenters.conf`
 - ◆ Common settings are `inner`, `outer`, `none`, and `full`, but the default file contains other predefined segmentation rules as well.
 - ◆ Create your own custom rule by editing `$SPLUNK_HOME/etc/system/local/segmenters.conf`, as described in ["Configure segmentation types"](#).

Example

This example sets both index-time and search-time segmentation rules for `syslog` events.

Add the following to the `[syslog]` source type stanza in `props.conf`:

```
[syslog]
SEGMENTATION = inner
SEGMENTATION-full= inner
```

This stanza changes the index-time segmentation for all events with a `syslog` source type to inner segmentation. It also causes the `full` radio button in Splunk Web to invoke inner segmentation for those same events.

Note: You must restart Splunk Enterprise to apply changes to search-time segmentation. You must re-index your data to apply index-time segmentation changes to existing data.

Set search-time event segmentation in Splunk Web

Splunk Web allows you to set segmentation for search results. While this has nothing to do with index-time segmentation, search-time segmentation in Splunk Web affects browser interaction and can speed up search results.

To set search-result segmentation:

1. Perform a search. Look at the results.

2. Click **Options...** above the returned set of events.

3. In the **Event Segmentation** dropdown box, choose from the available options: full, inner, outer, or raw. The default is "full".

You can configure the meaning of these dropdown options, as described in ["Set the segmentation for event data"](#).

Improve the data input process

Use a test index to test your inputs

Before you add new inputs to a production index on your Splunk Enterprise instance, it's a good idea to test the inputs first by adding them to a test index. After you verify that you're receiving the right data and that the resulting events are in a usable form, you can reconfigure the inputs to a production index. You can continue to test new inputs this way over time. If you find that the inputs you started with aren't the ones you want, you can keep working with the test index until you get results you like.

You can also preview how Splunk Enterprise indexes your data into a test index. During preview, adjust the event processing settings interactively. See [Assign the correct source type to your data](#) for details.

Use a test index

1. Create the test index using Splunk Web or, if you have Splunk Enterprise, using the CLI or by editing the `indexes.conf` configuration file.
2. When you configure the data inputs, send events to the test index. You can do this in Splunk Web. For each input, perform these steps:
 1. When you configure the input from the **Add data** page, check the **More settings** option. It reveals several new fields, including one called **Index**.
 2. From the **Index** drop-down list, select your test index. All events for that data input now go to that index.
 3. Repeat this process for each data input that you want to send to your test index.
3. When you search, specify the test index in your search command. By default, Splunk Enterprise searches the **main** index. Use the `index=` command, like this:

```
index=test_index
```

When you search a test index for events coming in from your newly created input, change the time range for the fields side bar to `"Real-time > All time (real-time)"`. The resulting real-time search shows all events being written to that index regardless of the value of their extracted time stamp. This result is particularly useful if you're indexing historical data into your index that a search for `"Last hour"` or `"Real-time > 30 minute window"` wouldn't show.

To learn how to create and use custom indexes with Splunk Web, see *Create custom indexes* in the Splunk Enterprise *Managing Indexers and Clusters of Indexers* manual. You can also specify an index when you configure an input in the `inputs.conf` configuration file.

Delete indexed data and start over

If you want to clean out your test index and start over on Splunk Enterprise, use the CLI `clean` command, as described *Remove indexes and indexed data* in the Splunk Enterprise *Managing Indexers and Clusters of Indexers* manual.

Configure your inputs to use the default index

When you're satisfied with the test results and you're ready to start indexing for real, edit your data inputs so that they send data to the default **main** index instead of the test index. For each data input that you've already set up, follow the reversed steps that you took to set up the test index:

1. Go back to the place where you initially configured the input. For example, if you configured the input from the **Add data** page in Splunk Web, return to the configuration screen for that input:
 1. Select **System > System configurations > Data inputs**.
 2. Select the input data type to see a list of all configured inputs of that type.
 3. Select the specific data input that you want to edit. Selecting the input takes you to a screen where you can edit it.
 4. Select the **Display advanced settings** option. Go to the field named **Index**.
 5. In the **Index** drop-down list, select the **main** index. All events for that data input now go to that index.

If you instead used the `inputs.conf` file to configure an input, you can change the index directly in that file, as described in *Create custom indexes in the Splunk Enterprise [Managing Indexers and Clusters of Indexers](#) manual*.

Now when you search, you no longer need to specify an index in the search command. By default, Splunk software searches the **main** index.

Use persistent queues to help prevent data loss

Persistent queuing lets you store data in an input queue to disk. In a Splunk Cloud Platform deployment, persistent queues can help prevent data loss if a **forwarder** that you configured to send data to your Splunk Cloud Platform instance backs up. In a Splunk Enterprise deployment, persistent queues work for either forwarders or **indexers**. You can't configure persistent queues directly on a Splunk Cloud Platform instance.

By default, forwarders and indexers have an in-memory input queue of 500 KB. If the input stream runs at a faster rate than the forwarder or indexer can process, to a point where the input queue on the forwarder maxes out, undesired consequences occur. In the case where you send network data over the UDP protocol, that data drops off of the queue and gets lost. For other types of data inputs, the application that generates the data can get backed up.

By implementing persistent queues, you can help prevent this data drop or loss from happening. With persistent queuing, after the in-memory queue is full, the forwarder or indexer writes the input stream to files on disk. It then processes data from the in-memory and disk queues until it reaches the point when it can again start processing directly from the data stream.

While persistent queues help prevent data loss if processing gets backed up, you can still lose data if the forwarder or indexer crashes. For example, the forwarder holds some input data in the in-memory queue as well as in the persistent queue files. The in-memory data can get lost if the forwarder crashes. Similarly, data that is in the parsing or indexing pipeline but that has not yet been written to disk can get lost in a crash.

When can you use persistent queues?

Persistent queuing is available for certain types of inputs, but not all. Generally speaking, persistent queuing is available for inputs of an ephemeral nature, such as network inputs, but isn't available for inputs that have their own form of persistence, such as monitoring files.

Persistent queues are available for these input types:

- Network inputs that use the TCP protocol
- Network inputs that use the UDP protocol
- First-In, First-Out (FIFO) inputs
- Scripted inputs

- Windows Event Log inputs
- HTTP Event Collector tokens

Persistent queues aren't available for these input types:

- Monitor inputs
- Batch inputs
- File system change monitor

Configure a persistent queue

Use the `inputs.conf` configuration file to configure a persistent queue. You can configure the persistent queue on the universal forwarder that you configured to send data to Splunk Cloud Platform. You can also configure persistent queues on Splunk Enterprise indexers. Use the same procedure directly on the indexer or forwarder that sends data to the indexer.

Inputs don't share queues. You configure a persistent queue in the stanza for the specific input.

1. On the machine that forwards data to Splunk Cloud Platform, use a text editor to open the `$SPLUNK_HOME/etc/system/local/inputs.conf` file for editing.
2. Locate or add the input stanza where you want to enable persistent queuing.
3. Specify the following setting within that input stanza:

```
persistentQueueSize = <integer>[KB|MB|GB|TB]
```

4. Save the file and close it
5. Restart the forwarder.

For more information about the `inputs.conf` file, see `inputs.conf` in the Splunk Enterprise *Admin Manual*.

Example of configuring a persistent queue

Here's an example of specifying a 10MB persistent queue for a TCP network input:

```
[tcp://9994]
persistentQueueSize=10MB
```

Here is another example for specifying a 15MB persistent queue for a Windows Event Log input:

```
[WinEventLog]
persistentQueueSize=15MB
```

The Windows Event Log monitor accepts a persistent queue configuration for the default Windows Event Log stanza only. You cannot configure a persistent queue for a specific Event Log channel. You can configure a persistent queue for a specific Windows host monitoring input.

Persistent queue location

The persistent queue has a hardcoded location, which varies according to the type of input.

For network inputs, the persistent queue is located at `$SPLUNK_HOME/var/run/splunk/[tcpin|udpin]/pq__<port>`.

Put two underscores in the file name: `pq__`, not `pq_`.

See the following examples:

- The persistent queue for TCP port 2012 is `$(SPLUNK_HOME)/var/run/splunk/tcpin/pq__2012`.
- The persistent queue for UDP port 2012 is `$(SPLUNK_HOME)/var/run/splunk/udpin/pq__2012`.

For FIFO inputs, the persistent queue resides in `$(SPLUNK_HOME)/var/run/splunk/fifoin/<encoded path>`.

For scripted inputs, the persistent queue resides in `$(SPLUNK_HOME)/var/run/splunk/exec/<encoded path>`. The FIFO scripted input stanza in the `inputs.conf` file derives the `<encoded path>`.

Use ingest actions to improve the data input process

Ingest actions is a feature for routing, filtering, and masking data while it is streamed to your indexers. Each data transformation is expressed as a rule. You can apply multiple rules to a data stream, and save the combined rules as a ruleset.

The Ingest Actions page in Splunk Web allows you to dynamically preview and build rules, using sample data.

You can configure ingest actions for these deployment topologies:

- Indexer clusters. Configure and preview the ruleset from the cluster manager. You then explicitly deploy the ruleset from the manager to the indexers using the configuration bundle method.
- Standalone indexers. Configure, preview, and save the ruleset directly on the indexer. The ruleset is effective immediately.
- Heavy forwarders via deployment server. Configure the ruleset on a deployment server. The deployment server automatically deploys the ruleset to heavy forwarders configured as deployment clients.
- Standalone heavy forwarders. Configure and save the ruleset directly on the forwarder. The ruleset is effective immediately.
- Splunk Cloud Platform. Configure and preview the ruleset from your search head. In the case of the Victoria Experience, the ruleset will be deployed automatically to the indexers. In the case of the Classic Experience, you need to explicitly deploy the ruleset.

Requirements

Indexer cluster

- All nodes on the indexer cluster must be running Splunk Enterprise for Linux.
- Requires access to Splunk Web on the cluster manager as the `admin` role, or as a member of a role with the `list_ingest_rulesets` and `edit_ingest_rulesets` capabilities.

Standalone indexer

- The indexer must be running Splunk Enterprise for Linux.
- Requires access to Splunk Web as the `admin` role, or as a member of a role with the `list_ingest_rulesets` and `edit_ingest_rulesets` capabilities.

Heavy forwarders managed through a deployment server

- The heavy forwarders and deployment server must each be running Splunk Enterprise for Linux.
- Requires access to Splunk Web on the deployment server as the `admin` role, or as a member of a role with the `list_ingest_rulesets` and `edit_ingest_rulesets` capabilities.
- A maximum of ten heavy forwarders is supported.
- The deployment server must be dedicated to the ingest actions function and cannot be used for other types of configurations. The deployment server cannot service any other deployment clients.
- The heavy forwarders must be preconfigured as deployment clients of the deployment server where the data ingest configuration occurs. For information on configuring deployment clients, see [Configure deployment clients](#).
- The Ingest Actions page on the deployment server automatically creates the `IngestAction_AutoGenerated` server class and assigns that class to the forwarders.
- If you want the heavy forwarders to send data to an S3 destination, you must configure the S3 destination on each of the heavy forwarders individually, either through the Ingest Actions page on each forwarder or through an `outputs.conf` file on each forwarder. You cannot configure the destination on the deployment server. To configure the destination on the Ingest Actions page, the heavy forwarders require access to Splunk Web as the `admin` role, or as a member of a role with the `list_ingest_rulesets` and `edit_ingest_rulesets` capabilities.

Standalone heavy forwarder

- The heavy forwarder must be running Splunk Enterprise for Linux.
- Requires access to Splunk Web as the `admin` role, or as a member of a role with the `list_ingest_rulesets` and `edit_ingest_rulesets` capabilities.

Splunk Cloud Platform

- Requires access to Splunk Web on the search head as the `sc_admin` role, or as a member of a role with the `list_ingest_rulesets` and `edit_ingest_rulesets` capabilities.

Introduction to rules and rulesets

A rule is a specific type of data transformation. A rule can route, filter, or mask data. Descriptions are provided below. By using multiple rules, you can perform complex modifications to an incoming data source before its data is indexed, or skip indexing of some data entirely.

A ruleset is a set of rules applied to a data source. Only one ruleset per source type is supported. Rules in a ruleset are processed in order.

You create rules through the Ingest Actions page:

- On indexer clusters, you access the Ingest Actions page on the cluster manager.
- For groups of heavy forwarders, you access the Ingest Actions page on a deployment server dedicated to the ingest actions function.
- On Splunk Cloud Platform, you access the Ingest Actions page on a search head.

Once you create a ruleset you must save it. Depending on where you created the ruleset, the ruleset is either immediately effective or requires an additional deployment step.

Once the ruleset has been deployed, each rule in the ruleset is applied to its matching data stream before the data is indexed.

After a ruleset is applied, the data cannot be reverted to its original form. Changing or disabling an existing ruleset affects only new data. If you want to retain the original data while also modifying some of it, use the clone events feature, described in this topic.

Access and edit the Ingest Actions page

The process of accessing the Ingest Actions page varies slightly depending on the deployment topology.

On indexer clusters

For Splunk Enterprise indexer clusters, perform these steps to create a ruleset:

1. On the cluster manager, select Settings > Data > Ingest Actions.
2. If routing to S3, add an S3 destination through the Destinations tab.
3. Through the Rulesets tab:
 1. Provide a ruleset name and description.
 2. In the Event Stream, provide a source type for the data preview.
 3. Add a rule. Descriptions are provided below.
 4. Use the data preview to review the impact of the rule on your data source.
 5. Add additional rules as needed.
 6. Save your rules in the ruleset.
4. Deploy the ruleset from the indexer cluster manager node. See [Deploy a ruleset on an indexer cluster](#).
5. Use Splunk Search to validate the changes to your data.

If you edit or delete an existing destination, the peer nodes will undergo a rolling restart when the changes are deployed.

On standalone indexers

For Splunk Enterprise indexers, perform these steps to create a ruleset:

1. On the indexer, select Settings > Data > Ingest Actions.
2. If routing to S3, add an S3 destination through the Destinations tab.
3. Through the Rulesets tab:
 1. Provide a ruleset name and description.
 2. In the Event Stream, provide a source type for the data preview.
 3. Add a rule. Descriptions are provided below.
 4. Use the data preview to review the impact of the rule on your data source.
 5. Add additional rules as needed.
 6. Save your rules in the ruleset. The updates are effective immediately on the indexer.
4. Use Splunk Search to validate the changes to your data.

If you edit or delete an existing destination, you must restart the instance for the changes to take effect.

On heavy forwarders managed through a deployment server

For Splunk Enterprise heavy forwarders managed through a deployment server, perform these steps to create a ruleset:

1. On the deployment server, select Settings > Data > Ingest Actions.

2. If routing to S3, add an S3 destination directly on each heavy forwarder, as described in the note below.
3. Through the Rulesets tab:
 1. Provide a ruleset name and description.
 2. In the Event Stream, provide a source type for the data preview.
 3. Add a rule. Descriptions are provided below.
 4. Use the data preview to review the impact of the rule on your data source. Currently, you can only preview changes by uploading a log file to the deployment server. You cannot use preview with live data streaming to the heavy forwarders..
 5. Add additional rules as needed.
 6. Save your rules in the ruleset. The deployment server saves the ruleset in the `splunk_ingest_actions` app for the `IngestAction_AutoGenerated` server class. It then automatically deploys the app to all members of the `IngestAction_AutoGenerated` server class, first adding all forwarders to that class, if necessary. The ruleset takes effect immediately.
4. Use Splunk Search to validate the changes to your data.

If you want the heavy forwarders to send data to an S3 destination, you must configure the destination individually on each heavy forwarder prior to creating the ruleset on the deployment server. Select Settings > Data > Ingest Actions on each heavy forwarder and configure the destination. You can alternatively create the destination in `outputs.conf` on each forwarder. If you edit or delete an existing destination, you must restart the forwarder for the changes to take effect.

On standalone heavy forwarders

For Splunk Enterprise heavy forwarders, perform these steps to create a ruleset:

1. On the heavy forwarder, select Settings > Data > Ingest Actions.
2. If routing to S3, add an S3 destination through the Destinations tab.
3. Through the Rulesets tab:
 1. Provide a ruleset name and description.
 2. In the Event Stream, provide a source type for the data preview.
 3. Add a rule. Descriptions are provided below.
 4. Use the data preview to review the impact of the rule on your data source.
 5. Add additional rules as needed.
 6. Save your rules in the ruleset. The updates are effective immediately on the heavy forwarder.
4. Use Splunk Search to validate the changes to your data.

If you edit or delete an existing destination, you must restart the forwarder for the changes to take effect.

On Splunk Cloud Platform

For Splunk Cloud Platform, perform these steps to create a ruleset:

1. On the search head, select Settings > Data > Ingest Actions.
2. If routing to S3, add an S3 destination through the Destinations tab.
3. Through the Rulesets tab:
 1. Provide a ruleset name and description.
 2. In the Event Stream, provide a source type for the data preview.
 3. Add a rule. Descriptions are provided below.
 4. Use the data preview to review the impact of the rule on your data source.
 5. Add additional rules as needed.

6. Save your rules in the ruleset. In the case of the Victoria Experience, the ruleset deploys immediately. In the case of the Classic Experience, you must explicitly deploy the ruleset with the Deploy button at the top right of the Ingest Actions page.
4. Use Splunk Search to validate the changes to your data.

Use the Ingest Actions page

Data preview

Data preview is available when you're building a ruleset using the Ingest Actions page. The data preview uses live indexed data, an uploaded sample file, or copied/pasted event logs to help you define rules. It also estimates the changes a rule will have on the data source. The data preview is only a preview of the rule changes, and does not modify any indexed data.

Selecting **Sample** retrieves events from the indexers. Selecting **Apply** applies all the created rules to the events in the preview, and provides an estimate of data volume based upon any modifications made using the rule. The **All Events** tab provides a visual indication of the rule matches. The **Affected Events** tab provides a total count, and displays the full event for every rule match.

Live data preview is not available on deployment servers. You can, however, use data preview on an uploaded sample file.

Mask with regular expression

Use a masking rule to replace strings of text in your logs. A mask rule is typically applied to fields with unique identifiers, or user names, that are captured through logging.

The mask rule requires you to provide:

Setting	Description
Match Regular Expression	The field accepts a regular expression, or a simple string to match in the events.
Replace Expression	The field accepts a string value you want to use to replace any matches.

Filter with regular expression

Use a filtering rule to remove entire events from your logs. A filter rule is typically applied to log events that are not valued, such as `DEBUG` messages, log headers, and redundant log messages.

This filter rule requires you to provide:

Setting	Description
Source Field	Use the drop down to select a data source by: <code>_raw</code> , host, index, source, or source type.
Drop Events Matching Regular Expression	The field accepts a regular expression, or a simple string to match in the events.

When using a filter rule, the **Affected Events** tab is a preview of events that will be deleted once the ruleset is deployed. If you add another rule after a filter, the new rule applies to any remaining, unfiltered events only.

Filter with eval expression

Using an eval expression is an alternative to using a regular expression for filtering. In most cases, the eval syntax is easier to read and comprehend, while offering the same functionality as a regular expression.

The eval expression rule does not support ingest-time lookups.

Use a filtering rule to remove entire events from your logs. A filter rule is typically applied to log events that are not valued, such as `DEBUG` messages, log headers, and redundant log messages.

This filter rule requires you to provide:

Setting	Description
Drop Events Matching Eval Expression	When the eval expression match is true, those events will be dropped.

When using a filter rule, the **Affected Events** tab is a preview of events that will be deleted once the ruleset is deployed. If you add another rule after a filter, the new rule applies to any remaining, unfiltered events only.

Route to Destination rule

Use a routing rule to select events, and split or duplicate them between one or more destinations.

This routing rule requires you to provide:

Setting	Description
Condition	Choose a method to match events for routing. Choose the regex or eval condition to select specific events, or none when you want all events sent to a destination. If a condition is set, only events matching the condition will be sent to the destination(s).
Immediately send to	By default, the destination is "Splunk Index". Any matching events are placed back into the Splunk Enterprise indexing queue for processing and indexing. The destination rule also supports AWS S3 and other S3-compliant destinations. If more than one destination is chosen, a copy of any matching events is sent to all destinations chosen.
Clone events and apply more rules	This toggle causes data ingest to create a clone of the event stream, applying the rules currently defined in the ruleset, and route the stream to the specified destination, while applying any additional newly defined rules against the event stream and routing that subset to a second specified destination, defined in a second Route to Destination rule. As with all rules, the ruleset must be saved and deployed before the destination rules start functioning.

Routing to AWS S3

Use the AWS S3 destination to write any matching events to a bucket on a remote storage volume.

Configure the AWS S3 remote storage settings before using AWS S3 in a "Route to Destination" rule.

You can configure and validate an S3 destination through the Data Ingest page under the Destinations tab. Select **New Destination** and fill out the fields, following the examples provided there.

In the case of heavy forwarders managed through a deployment server, the S3 destination must be configured on each heavy forwarder individually, not on the deployment server.

While Destinations on the Data Ingest page can handle most common S3 configuration needs, for some advanced configurations, like encryption-at-rest with KMS, you might need to directly edit `outputs.conf`, using the `rfs` stanza. For example, assuming you are using an EC2 instance with a properly configured IAM role:

```
[rfs:s3]
path = s3://mybucket/myprefix/
remote.s3.endpoint = https://s3.us-west-2.amazonaws.com
remote.s3.signature_version = v4
remote.s3.supports_versioning = false
remote.s3.encryption = sse-kms
remote.s3.kms.key_id = <key ID from AWS>
remote.s3.kms.auth_region = <auth region>
```

For a complete list of `rfs` settings, see Remote File System (RFS) Output. The remote filesystem settings and options for AWS S3 are similar to the SmartStore S3 configuration.

To troubleshoot the AWS S3 remote file system, search the `_internal` index for events from the `RfsOutputProcessor` and `S3Client` components. For example:

```
index="_internal" sourcetype="splunkd" (ERROR OR WARN) RfsOutputProcessor OR S3Client
```

Note the following:

- Only a single, globally configured remote storage location is supported.
- Index-time fields will not be transferred to S3. This can lead to loss of field names from INDEXED_EXTRACTIONS sources like W3C and CSV.
- In the case of an indexer cluster, the remote storage configuration must be identical across the indexer cluster peers.
- The remote file system creates buckets similar to index buckets on the remote storage location. The bucket names include the peer GUID and date.
- Remember to set the correct life cycle policies for your S3 bucket and their paths. This data will live forever by default unless removed.
- For information on S3 authentication requirements, see SmartStore on S3 security strategies in *Managing Indexers and Clusters of Indexers*. Ingest actions requirements are similar.

Data Preview for Final Destination

The last rule in every ruleset sends any remaining events along the ingestion pipeline to the indexer for indexing. The rule offers an estimate of the data volume that will be indexed.

If you use the "Route to Destination" rule in your ruleset, this rule might be skipped. For example, if a Route to Destination rule includes "Immediately send to: Splunk Index," the data stream is split at the routing rule, and the matching events are sent to be indexed. In that scenario, the Final Destination rule will display a 0Kb indexed data estimate, despite events being sent for indexing from the routing rule.

Deploy a ruleset on an indexer cluster

You must explicitly deploy a ruleset on an indexer cluster. For most other deployment types, the ruleset deploys automatically when it is saved.

When you save a ruleset, the system places the ruleset in an ingest-actions-specific app on the cluster manager. You will then be prompted to deploy the ruleset to the peer nodes, which you do through the normal configuration bundle method.

Note the following:

- All rulesets are defined in the same app on the cluster manager node. The app path is:
`$SPLUNK_HOME/etc/manager-apps/splunk_ingest_actions`
- When you deploy the app with your ruleset, any other configuration bundle changes queued on the cluster manager node will also be deployed. This can include other rulesets that are saved, but might be incomplete.

Deploying a ruleset might cause a rolling restart, if there are other configuration changes queued on the cluster manager node that require a restart.

Interaction with TRANSFORMS

The `RULESET` setting is similar in behavior to the `TRANSFORMS` setting in `props.conf`. There are some additional considerations when using `RULESET`:

- If a `TRANSFORMS` stanza and a `RULESET` stanza apply to the same source type, the `TRANSFORMS` is applied first.
- A source type must be associated with just one `RULESET` configuration.

Create or modify rulesets only through the Ingest Actions page or the REST endpoint `/services/data/ingest/rulesets`. Do not create or modify rulesets through the underlying `.conf` files.

Differences between RULESET and TRANSFORMS in the context of heavy forwarders

The `RULESET` setting has a key difference in behavior from the `TRANSFORMS` setting in the context of a heavy forwarder deployment:

- `TRANSFORM` settings are applied only at the initial, heavy forwarder layer of processing, and not again later with downstream heavy forwarders or indexers.
- `RULESET` settings can be applied at every layer of processing. For example, a heavy forwarder can apply a ruleset and then stream the data to an indexer with its own ruleset for that data. In that case, both the heavy forwarder's and the indexer's rulesets will be applied to the data in turn. Similarly, if a heavy forwarder streams data to a second heavy forwarder, which then streams the data onward to the indexer, all three processing layers can apply their own rulesets to the data.

Troubleshoot the input process

Following are some initial steps you can take to troubleshoot the data input process on Splunk Enterprise.

Determine why you do not find the events you expect

When you add an input to your Splunk Enterprise deployment, that input gets added relative to the app you are in. Some apps write input data to a specific index. If you cannot find data that you are certain is in your Splunk Enterprise deployment, confirm that you are looking at the right index. See Retrieve events from indexes in the *Search Manual*. You might want to add indexes to the list of default indexes for the role you are using.

- For more information about roles, see Add and edit roles in the *Securing the Splunk Platform* manual.
- For more information about troubleshooting data input issues, read the rest of this topic or see I can't find my data! in the *Troubleshooting Manual*.

If you use Splunk Enterprise and add inputs by editing the `inputs.conf` configuration file, Splunk Enterprise might not recognize the inputs immediately. Splunk Enterprise looks for inputs every 24 hours, starting from the time it was last restarted, so if you add a new stanza to monitor a directory or file, it could take up to 24 hours for Splunk Enterprise to start indexing the contents of that directory or file. To ensure that your input is immediately recognized and indexed, add the input through Splunk Web or the CLI, or restart Splunk services after you make edits to the `inputs.conf` file.

Troubleshoot your tailed files

You can use the **FileStatus** Representational State Transfer (REST) endpoint to get the status of your tailed files. For example:

```
curl https://serverhost:8089/services/admin/inputstatus/TailingProcessor:FileStatus
```

You can also monitor the **fishbucket**, a subdirectory that Splunk software uses to keep track of how much of a file's contents have been indexed. In Splunk Enterprise deployments, the fishbucket resides at `$SPLUNK_DB/fishbucket/splunk_private_db`. In Splunk Cloud Platform deployments you do not have physical access to this subdirectory.

To monitor the fishbucket, use the REST endpoint. Review the REST API Reference manual for additional information.

Troubleshoot ingestion congestion on Splunk Enterprise

Sometimes, Splunk Enterprise data ingestion can slow for what appears to be an unknown reason. One possibility for this slowness could be the number of inactive input channels available on your Splunk Enterprise indexers.

Description of an input channel

An indexer must track the state of each unique stream of data that it processes. For example, when it breaks up lines of data that it has ingested from a set of tailed files, the indexer receives data from these files in an order that cannot be predicted. Parts of various files can be interleaved with one another. An indexer prevents this interleaving from causing the line breaking of one file from interfering with the line breaking of another by tracking the state of each file with a data structure called an input channel.

An input channel stores a variety of information, including the following information:

- The state of the linebreaker
- The state of the aggregator
- The punct state
- The settings in `props.conf` for the input

There is a unique input channel for each source, source type, or host stream that the indexer encounters.

Description of an inactive input channel

An indexer does not, for performance and memory usage reasons, keep input channels around forever. After a channel has not been used for a while, for example, after data for a particular source, source type, and host tuple has not appeared for a while, a channel becomes eligible for reuse by a different stream. Splunk Enterprise has several settings that control the recycling behavior for inactive channels. You configure these settings in the `limits.conf` configuration file.

For example, suppose the indexer has just encountered a new stream. As a result, it needs an input channel into which it can save the state of this stream as it ingests it. At this point, it must decide whether to create a new input channel, thus

using more memory, or to reuse an inactive channel, and incur a performance penalty if that inactive channel becomes active again.

When determining whether or not to use an inactive input channel, the indexer follows the following decision process:

1. If the number of inactive channels is less than or equal to the value set for the `lowater_inactive` setting in the `limits.conf` configuration file, it creates an input channel.
2. If the number of inactive channels is greater than the value set for the `max_inactive` setting, or the age of the oldest inactive channel is greater than the value set for the `inactive_eligibility_age_seconds` in the `limits.conf` file, do one of the following things:
 - ◆ Recycle the oldest inactive input channel.
 - ◆ Create an input channel.

Put another way:

- The indexer always creates a new input channel if it is currently below the `lowater_inactive` value.
- The indexer always recycles an inactive input channel if it is currently above the `max_inactive` value.
- If the indexer is above the `lowater_inactive` value and below the `max_inactive` value at the same time, it recycles the oldest inactive channel if that channel is older than `inactive_eligibility_age_seconds` seconds; otherwise, it creates a new input channel.

The `max_inactive` setting now has a setting value `auto`. This configures the indexer to adjust the `max_inactive` setting based on the amount of memory that is present in the machine that runs the instance.

Configure manual or automatic inactive input channel limits

You can adjust the amount of maximum inactive input channels that an indexer keeps available. Increasing this number manually increases the amount of memory that the indexer uses. Lower numbers mean less memory usage by the indexer, but an increase in the amount of new input channels that the indexer creates, which can significantly reduce performance based on the amount of sources, source types, and hosts that the indexer encounters while it processes incoming data. Each inactive input channel takes around 5kB of memory.

1. On the indexer where you want to adjust inactive input channel limits, open a shell or command prompt or text editor.
2. Open the `$SPLUNK_HOME/etc/system/local/limits.conf` file for editing.
3. In this file, locate the `[input_channels]` stanza. If the stanza does not exist in the file, create it.
4. Under the `[input_channels]` stanza, add the following line:

```
max_inactive = <positive integer>
```

If you want the indexer to manage the number of inactive channels automatically, change the line to

```
max_inactive = auto
```

5. Save the file and close it.
6. Restart The Splunk Enterprise instance to apply the change.

For more information on the `max_inactive`, `lowater_inactive`, and `inactive_eligibility_age_seconds` settings for `limits.conf`, see the `limits.conf` specification file.

Can't find forwarded data?

Confirm that the forwarder functions properly and is visible to the indexer. You can use the Distributed Management Console (DMC) to troubleshoot Splunk topologies and get to the root of any forwarder issues. See *Monitoring Splunk Enterprise* for details.

Resolve data quality issues

You can troubleshoot the following event processing and data quality issues when you get data in to the Splunk platform:

- Incorrect line breaking
- Incorrect event breaking
- Incorrect timestamp extraction

Line breaking issues

The following symptoms indicate that there might be issues with line breaking:

- You have fewer events than you expect and the events are very large, especially if your events are single-line events.
- The Monitoring Console **Data Quality** dashboard displays issues with line breaking.
- You might see the following error message in the Splunk Web Data Input workflow or in the splunkd.log file:
"Truncating line because limit of 10000 bytes has been exceeded".

Diagnosis

To confirm that the Splunk platform has line breaking issues, do one or more of the following troubleshooting steps:

- Visit the Monitoring Console **Data Quality** dashboard. Check the dashboard table for line breaking issues. See About the Monitoring Console in *Monitoring Splunk Enterprise*.
- Look for messages in the splunkd.log file like the following example:

```
12-12-2016 13:45:48.709 -0800 WARN LineBreakingProcessor - Truncating line because limit of 10000 bytes has been exceeded with a line length >= 301367
```

- Search for events. Multiple combined events, or a single event broken into many, indicates a line breaking issue.

Solution

To resolve line breaking issues, complete these steps in Splunk Web:

1. Click **Settings > Add Data**.
2. Click **Upload** to test by uploading a file or **Monitor** to redo the monitor input.
3. Select a file with a sample of your data.
4. Click **Next**.
5. On the **Set Source Type** page, work with the options on the left panel until your sample data is correctly broken into events. To configure `LINE_BREAKER` or `TRUNCATE`, click **Advanced**.
6. Complete the data input workflow or record the correct settings and use them to correct your existing input configurations.

While you work with the options on the Set Source Type page, the `LINE_BREAKER` setting might not be properly set. The `LINE_BREAKER` setting must have a capturing group and the group must match the events.

For example, you might have a value of `LINE_BREAKER` that is not matched. Look for messages with "Truncating line because limit of 10000 bytes has been exceeded" in the `splunkd.log` file or look for the following message in Splunk Web:



If you find such a message, do the following:

1. Confirm that `LINE_BREAKER` is properly configured to segment your data into lines as you expect.
2. Confirm that the string you specify in the `LINE_BREAKER` setting exists in your data.
3. If `LINE_BREAKER` is configured correctly but you have very long lines, or if you are using `LINE_BREAKER` as the only method to define events, bypassing line merging later in the indexing pipeline, confirm that the `TRUNCATE` setting is large enough to contain the entire data fragment delimited by `LINE_BREAKER`. The default value for `TRUNCATE` is 10,000. If your events are larger than the `TRUNCATE` value, you might want to increase the value of `TRUNCATE`. For performance and memory usage reasons, do not set `TRUNCATE` to unlimited.

If you do not specify a capturing group, `LINE_BREAKER` is ignored.

For more information, see [Configure event line breaking](#).

Event breaking or aggregation issues

Event breaking issues can pertain to the `BREAK_ONLY_BEFORE_DATE` and `MAX_EVENTS` settings and any `props.conf` configuration file setting with the keyword `BREAK`.

You might have aggregation issues if you see the following indicators:

- Aggregation issues present in the Monitoring Console **Data Quality** dashboard.
- An error in the Splunk Web Data Input workflow.
- Count events. If events are missing and are very large, especially if your events are single-line events, you might have event breaking issues

Diagnosis

To confirm that the Splunk platform has event breaking issues, do one or more of the following troubleshooting steps:

- View the Monitoring Console **Data Quality** dashboard.
- Search for events that are multiple events combined into one.
- Check `splunkd.log` for messages such as the following:

```
12-07-2016 09:32:32.876 -0500 WARNââ AggregatorMiningProcessor - Breaking event because limit of
256 has been exceeded
12-07-2016 09:32:32.876 -0500 WARNââ AggregatorMiningProcessor - Changing breaking behavior for
event stream because MAX_EVENTS (256) was exceeded without a single event break. Will set
BREAK_ONLY_BEFORE_DATE to False, and unset any MUST_NOT_BREAK_BEFORE or MUST_NOT_BREAK_AFTER rules.
Typically this will amount to treating this data as single-line only.
```

Solution

For line and event breaking, determine whether this issue occurs for one of the following reasons:

- Your events are properly recognized but are too large for the limits in place. The `MAX_EVENTS` defines the maximum number of lines in an event.
- Your events are not properly recognized.

If your events are larger than the limit set in `MAX_EVENTS`, you can increase limits. Be aware that large events are not optimal for indexing performance, search performance, and resource usage. Large events can be costly to search. The upper values of both limits result in 10,000 characters per line, as defined by `TRUNCATE`, times 256 lines, as set by `MAX_EVENTS`. The combination of those two limits is a very large event.

If the cause is that your events are not properly recognized, which is more likely, the Splunk platform is not breaking events properly. Check the following:

- Your event breaking strategy. The default strategy breaks before the date, so if the Splunk platform does not extract a timestamp, it does not break the event. To diagnose and resolve this issue, investigate timestamp extraction. See [How timestamp assignment works](#).
- Your event breaking regular expression.

For more information, see the following topics:

- [Tune timestamp recognition for better indexing performance](#)
- [Configure event line breaking](#)

Time stamping issues

Time stamping issues can pertain to the following settings in the `props.conf` configuration file:

- `DATE_TIME_CONFIG`
- `TIME_PREFIX`
- `TIME_FORMAT`
- `MAX_TIMESTAMP_LOOKAHEAD`
- `TZ`

You might have timestamp parsing issues if you see the following indicators:

- Timestamp parsing issues are present in the Monitoring Console **Data Quality** dashboard.
- An error occurs in the Splunk Web Data Input workflow.
- Count events. If you are missing events and have very large events, especially if your events are single-line events, parsing might be a problem.
- The time zone is not properly assigned.
- The value of `_time` assigned by the Splunk platform does not match the time in the raw data.

Diagnosis

To confirm that you have a timestamping issue, do one or more of the following:

- Visit the Monitoring Console **Data Quality** dashboard. Check for timestamp parsing issues in the table. Time stamp assignment resorts to various fallbacks. For more details, see [How timestamp assignment works](#). For most of the fallbacks, even if one of them successfully assigns a timestamp, you still get an issue in the Monitoring Console dashboard.
- Search for events that are multiple events combined into one.
- Look in the splunkd.log file for messages like the following:

```
12-09-2016 00:45:29.956 -0800 WARN DateParserVerbose - Failed to parse timestamp. Defaulting to timestamp
of previous event (Fri Dec 9 00:45:27 2016). Context:
source::/disk2/sh-demo/splunk/var/log/splunk/entity.log|host::svdev-sh-demo|entity-too_small|682235
12-08-2016 12:33:56.025 -0500 WARNââ AggregatorMiningProcessor - Too many events (100K) with the same
timestamp: incrementing timestamps 1 second(s) into the future to insure retrievability
```

All events are indexed with the same timestamp, which makes searching that time range ineffective.

Solution

To resolve a timestamping issue, complete the following steps:

- Make sure that each event has a complete timestamp, including a year, full date, full time, and a time zone.
- See [Configure timestamp recognition](#) for more possible resolution steps.