Splunk User Manual

Version: 4.2.3

Generated: 10/18/2011 10:24 am

Table of Contents

Table of Contents

Welcome

What's in this manual

What's in this manual

In this manual, you'll find information and procedures for the **Splunk enterprise user**—if you use Splunk to investigate problems and report on results, this is the manual for you.

Where to start?

If you're new to Splunk, check out the overview and then proceed to the **Splunk tutorial!** It guides you through adding data, searching your data, and building simple reports and dashboards. Let us know what you think!

Continue reading to:

- learn how to add data to your indexes
- start searching with terms, Boolean expressions, and fields
- learn how to use the search results and timeline to interactively narrow your search
- learn how to save event types, extract new fields, and tag field values
- learn how to save searches and set alert conditions for scheduled searches
- start building reports and charts to save and share with others

If you want to just jump right in and start searching, see the Search command cheat sheet for a quick reference complete with descriptions and examples.

Make a PDF

If you'd like a PDF of any version of this manual, click the **pdf version** link above the table of contents bar on the left side of this page. A PDF version of the manual is generated on the fly for you, and you can save it or print it out to read later.

Splunk Overview

Splunk overview

Splunk overview

Splunk is powerful and versatile IT search software that takes the pain out of tracking and utilizing the information in your data center. If you have Splunk, you won't need complicated databases, connectors, custom parsers or controls--all that's required is a web browser and your imagination. Splunk handles the rest.

Use Splunk to:

- Continually index all of your IT data in real time.
- Automatically discover useful information embedded in your data, so you don't have to identify it yourself.
- Search your physical and virtual IT infrastructure for literally anything of interest and get results in seconds.
- Save searches and tag useful information, to make your system smarter.
- Set up alerts to automate the monitoring of your system for specific recurring events.
- Generate analytical reports with interactive charts, graphs, and tables and share them with others.
- Share saved searches and reports with fellow Splunk users, and distribute their results to team members and project stakeholders via email.
- Proactively review your IT systems to head off server downtimes and security incidents before they arise.
- Design specialized, information-rich views and dashboards that fit the wide-ranging needs of your enterprise.

Index new data

Splunk offers a variety of flexible data input methods to index everything in your IT infrastructure in real time, including live log files, configurations, traps and alerts, messages, scripts, performance data, and statistics from all of your applications, servers, and network devices. Monitor file systems for script and configuration changes. Enable change monitoring on your file system or Windows registry. Capture archive files and SNMP trap data. Find and tail live application server stack traces and database audit tables. Connect to network ports to receive syslog and other network-based instrumentation.

No matter how you get the data, or what format it's in, Splunk indexes it the same way--without any specific parsers or adapters to write or maintain. It stores both the raw data and the rich index in an efficient, compressed, filesystem-based datastore--with optional data signing and auditing if you need to prove data integrity.

For more details on data indexing with Splunk, see the "Index new data" chapter in this manual.

Search and investigate

Now you've got all that data in your system...what do you want to do with it? Start by using Splunk's powerful search functionality to look for anything, not just a handful of predetermined fields. Combine time and term searches. Find errors across every tier of your IT infrastructure and track down configuration changes in the seconds before a system failure occurs. Splunk identifies fields from your records as you search, providing flexibility unparalleled by solutions that require setup of rigid field mapping rulesets ahead of time. Even if your system contains terrabytes of data, Splunk enables you to search across it with precision.

To get the full picture of Splunk's IT search capability, see the "Search and investigate" chapter in this manual.

Capture knowledge

Freeform searching on raw data is just the start. Enrich that data and improve the focus of your searches by adding your own knowledge about fields, events, and transactions. Tag high-priority assets, and annotate events according to their business function or audit requirement. Give a set of related server errors a single tag, and then devise searches that use that tag to isolate and report on events involving that set of errors. Save and share frequently-run searches. Splunk surpasses traditional approaches to log management by mapping knowledge to data at search time, rather than normalizing the data up front. It enables you to share searches, reports, and dashboards across the range of Splunk apps being used in your organization.

To get more details on capturing and utilizing knowledge with event types and fields, see the "Capture knowledge" chapter in this manual.

Automate monitoring

Any search can be run on a schedule, and scheduled searches can be set up to trigger notifications or when specific conditions occur. This automated alerting functionality works across the wide range of components and technologies throughout your IT infrastructure--from applications to firewalls to access controls. Have Splunk send notifications via email or SNMP to other management consoles. Arrange for alerting actions to trigger scripts that perform activities such as restarting an application, server, or network device, or opening a trouble ticket. Set up alerts for known bad events and use sophisticated correlation via search to find known risk patterns such as brute force attacks, data leakage, and even application-level fraud.

For more information about monitoring recurring events, see the "Automate monitoring" chapter in this manual.

Analyze and report

Splunk's ability to quickly analyze massive amounts of data enables you to summarize any set of search results in the form of interactive charts, graphs, and tables. Generate reports on-the-fly that use statistical commands to trend metrics over time, compare top values, and report on the most and least frequent types of conditions. Visualize report results as interactive line, bar, column, pie, scatterplot and heat-map charts.

Splunk offers a variety of ways to share reports with team members and project stakeholders. You can schedule reports to run at regular intervals and have Splunk send each report to interested parties via email, print reports, save them to community collections of commonly-run reports, and add reports to specialized dashboards for quick reference.

For more information about defining reports, generating charts, and sharing them with others, see the "Analyze and report" chapter in this manual.

Ways to access Splunk

Ways to access Splunk

This topic discusses the different ways in which you can connect to and use Splunk.

Splunk Web

Splunk Web is Splunk's dynamic and interactive graphical user interface. Accessed via a Web browser, Splunk Web is the primary interface used to search and investigate, report on results, and manage one or more Splunk deployment. For our list of supported operating systems and browsers, see the "System requirements" in the Installation manual.

Launch Splunk Web in a browser After you install and start Splunk, launch a Web browser and navigate to:

```
http://mysplunkhost:8000
```

Use whatever host and HTTP port you chose during installation. The HTTP port defaults to 8000 if not otherwise specified.

The first time you log in to Splunk with an Enterprise license, use username `admin` and password `changeme`. Splunk with a free license does not have access controls.

Note: Starting in Splunk version 4.1.4, you cannot access Splunk Free from a remote browser until you have edited `$SPLUNK_HOME/etc/local/server.conf` and set `allowRemoteLogin` to `Always`. If you are running Splunk Enterprise, remote login is disabled by default (set to `requireSetPassword`) for the admin user until you have changed the default password.

Splunk apps in Splunk Web

When you first launch Splunk Web, you're looking at an app. For most users, this will be the core Search app. Others may see platform-specific apps, with dashboards and views for use with the OS. When you're using Splunk, you're using an app at all times; the dashboards and views available to you depend on the app you're currently using. For more information about Splunk and apps, see the "Splunk apps" topic in this chapter.

Splunk management pages

Whatever app you're in when you launch Splunk, you'll see at least two links in the upper right corner of the screen: Manager and Jobs.

The **Manager** link takes you to configuration and management pages for your Splunk system and apps. For more information about **Splunk Manager**, see [coming soon!].

The **Jobs** link opens the **Job Manager** window, which allows you to manage all search jobs, both completed and currently running. For more information about job management, see "Supervise your search jobs" in the Capture Knowledge chapter of this manual.

Splunk CLI

Splunk users can perform much of the core Splunk tasks directly from a commandline interface (CLI). These tasks include: managing inputs and indexes, searching, saving and scheduling searches as alerts, and exporting search results. If you don't have access to the CLI, you should communicate with your Splunk admin. For more information, see "About the CLI" in the Admin manual.

Splunk apps

Splunk apps

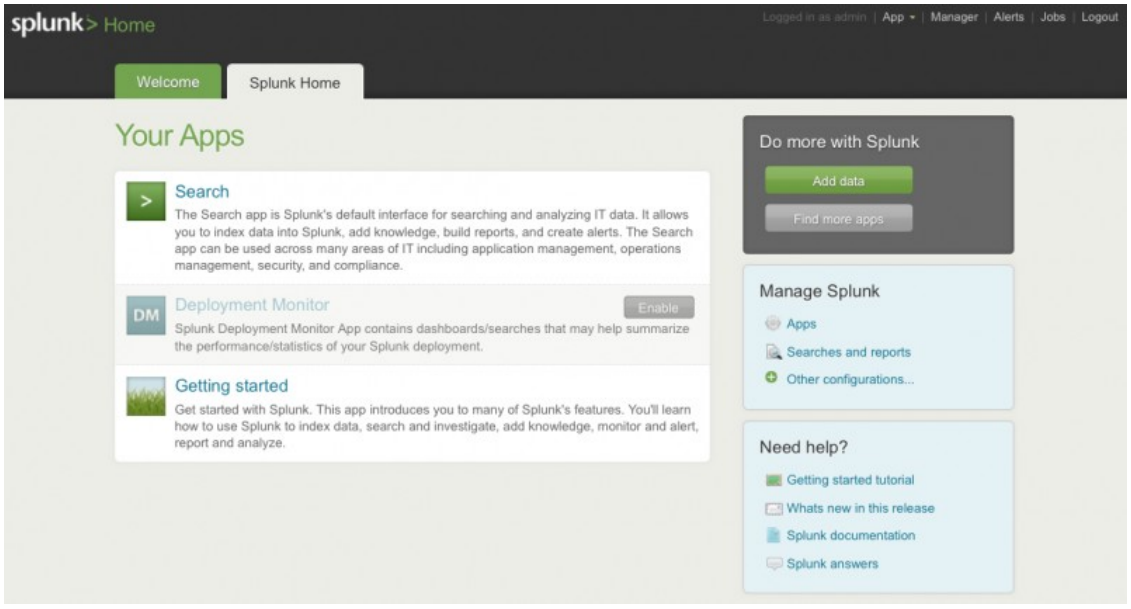
When you use Splunk, you're experiencing it in the context of one or more apps. Each Splunk app is a collection of dashboards and views. Some of them are designed specifically to help you manage data in specific OS platforms or address particular business objectives.

At all times, when you're using Splunk, you're using a Splunk app. We refer to this as being "in" an app.

For more detailed information about Splunk and apps, see "What's an app?" in the Admin Manual.

Splunk Home and the Getting Started app

Unless your Splunk administrator has configured your Splunk deployment differently, the first time you install and log into Splunk, you'll see the Welcome to Splunk screen. Click on the green Splunk Home tab. Splunk Home shows you the list of apps that have been preinstalled for you and that you have permissions to see. By default, one of these apps is the Getting Started app. This app has been developed to introduce new users to Splunk's features. If you're new to Splunk, we recommend you check it out and give us your feedback!



Search
The Search app is Splunk's default interface for searching and analyzing IT data. It allows you to index data into Splunk, add knowledge, build reports, and create alerts. The Search app can be used across many areas of IT including application management, operations management, security, and compliance.

Deployment Monitor
Enable
Splunk Deployment Monitor App contains dashboards/searches that may help summarize the performance/statistics of your Splunk deployment.

Getting started
Get started with Splunk. This app introduces you to many of Splunk's features. You'll learn how to use Splunk to index data, search and investigate, add knowledge, monitor and alert, report and analyze.

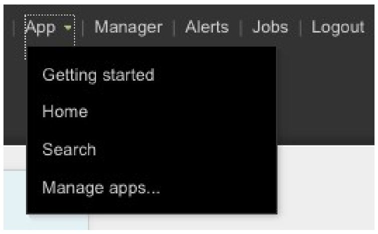
What else you get by default

Splunk also comes with the Search app and another app to support your OS by default.

- The Search app provides an interface that provides the core functionality of Splunk and is designed for general-purpose use. If you've used Splunk before, the Search app replaces the main Splunk Web functionality from earlier versions. In the Search app you see a search bar and a dashboard full of graphs. When you are in the Search app, you change the dashboard or view by selecting new ones from the **Views** drop-down menu in the upper left of the window.

- The OS-specific app (Splunk for Windows or Splunk for *NIX) provides dashboards and pre-built searches to help you get the most out of Splunk on your particular platform.

If you want to change the app you're in, select a new one from the app drop-down menu at the top right:



You can also return to Splunk Home and select another app from there.

Get more apps

You can add other apps to the list of apps in Splunk Home or in the Apps menu. For example, if the bulk of your data operations work involves tasks related to things like security, change management or PCI (Payment Card Industry) compliance, you'll be happy to know that Splunk has apps that specialize in helping you with them.

To find more apps to download, click the **Find More Apps** button on the right in Splunk Home.

Build apps to fit your needs

Splunk provides all the tools you need to create apps that answer the unique data management needs faced by your organization. You can design apps that have specialized dashboards and views and make them as simple or as sophisticated as you wish.

For more information about the nuts and bolts of Splunk app design and development, see the Developer manual.

Splunk Tutorial!

Welcome to the Splunk Tutorial

Welcome to the Splunk Tutorial

What is Splunk?

Splunk is software that indexes IT data from any application, server or network device that makes up your IT infrastructure. It's a powerful and versatile search and analysis engine that lets you investigate, troubleshoot, monitor, alert, and report on everything that's happening in your entire IT infrastructure from one location in real time.

Want to learn more about all the kinds of data Splunk can index? Read "What is IT data?" on our website.

Who uses Splunk?

Splunk is versatile and thus has many uses and many different types of users. System administrators, network engineers, security analysts, developers, service desk, and support staff -- even Managers, VPs, and CIOs -- use Splunk to do their jobs better and faster.

- Application support staff use Splunk for end-to-end investigation and remediation across the application environment and to create alerts and dashboards that proactively monitor performance, availability, and business metrics across an entire service. They use roles to segregate data access along lines of duties and give application developers and Tier One support access to the information they need from production logs without compromising security.
- System administrators and IT staff use Splunk to investigate server problems, understand their configurations, and monitor user activity. Then, they turn the searches into proactive alerts for performance thresholds, critical system errors, and load.
- Senior network engineers use Splunk to troubleshoot escalated problems, identify events and patterns that are indicators of routine problems, such as misconfigured routers and neighbor changes, and turn searches for these events into proactive alerts.
- Security analysts and incident response teams use Splunk to investigate activity for flagged users and access to sensitive data, automatically monitor for known bad events, and use sophisticated correlation via search to find known risk patterns such as brute force attacks, data leakage, and even application-level fraud.
- Managers in all solution areas use Splunk to build reports and dashboards to monitor and summarize the health, performance, activity, and capacity of their IT infrastructure and businesses.

What's in this tutorial?

If you're new to Splunk, this tutorial will teach you what you need to know to start using Splunk, from a first-time download to creating rich, interactive dashboards.

Before you start the tutorial

Before you start the tutorial

Before you can begin to use Splunk, you need to download, install, and start up a Splunk instance. Hey, no worries -- this only takes about 5 minutes!

If you already have access to a running Splunk server, skip down to Add data to Splunk and start there.

Do you have what it takes to run Splunk?

Splunk runs on most computing platforms, but this tutorial will focus specifically on the Windows and Mac OS X versions of Splunk. Of course, whatever platform you choose to run it on, it's still Splunk, and you should be able to follow along from Start Splunk onwards.

While Splunk is software that you install on your local machine, you access Splunk through a Web browser. Splunk supports most versions of Firefox, Internet Explorer, and Safari.

Splunk is a high-performance application, but for this tutorial, you really only need an individual Windows or Mac machine that meets at least the following specifications:

Platform	Minimum supported hardware capacity
Non-Windows platforms	1x1.4 GHz CPU, 1 GB RAM
Windows platforms	Pentium 4 or equivalent at 2Ghz, 2GB RAM

For the complete list of specifications, see the system requirements in the *Installation manual*.

Which license is for you?

Splunk runs with either an Enterprise license or a Free license. When you download Splunk for the first time, you get an Enterprise trial license that expires after 60 days. This trial license enables 500 MB/day indexing and all of the Enterprise features.

Once you install Splunk, you can run with the Enterprise trial license until it expires, switch to the perpetual Free license (it's included!), or purchase an Enterprise license.

Read more about Splunk licenses and features.

Download Splunk

The Windows installer is an MSI file. There are two Mac OS X installers; for this tutorial, you'll use the DMG package.

Download the latest version of Splunk from the download page.

Log into Splunk.com to download Splunk. If you're not logged on, clicking the download package will redirect you to a registration form. If you don't already have a Splunk.com account, sign up for one.

Install Splunk

Splunk provides graphical installers for the Windows and Mac OS X platforms, though you can also install using the command line interface, or CLI.

For command line instructions and installations on other platforms, see the detailed installation procedures in the *Installation manual*.

Windows

1. To start the installer, double-click the `splunk.msi` file.

2. In the Welcome panel, click **Next**.

3. Read the licensing agreement and check the box next to "I accept the terms in the license agreement". Click **Next** to continue installing.

4. In the **Customer Information**, enter the requested details and click **Next**.

5. In the **Destination Folder** panel, click **Change...** to specify a different location to install Splunk, or click **Next** to accept the default value.

Splunk is installed by default into the `\Program Files\Splunk` directory.

The **Logon Information** panel is displayed.

6. In the **Logon Information** panel, select *Local system user* and click **Next**.

If you want to learn about the other user option, refer to the detailed instructions for installing Splunk on Windows.

7. After you specify a user, the pre-installation summary panel is displayed. Click **Install** to proceed.

8. In the **Installation Complete** panel, check the boxes to **Launch browser with Splunk** and **Create Start Menu Shortcut** now.

9. Click **Finish**.

The installation completes, Splunk starts, and Splunk Web launches in a supported browser.

Mac OS X

1. Double-click on the DMG file.

2. In the Finder window, double-click on splunk.pkg.

The Splunk installer opens and displays the **Introduction**.

3. Click Continue.

4. In the **Select a Destination** window, choose a location to install Splunk.

* To install in the default directory, `/Applications/splunk`, click on the harddrive icon.
* To select a different location, click Choose Folder...

5. Click Continue.

The pre-installation summary displays. If you need to make changes,

* Click **Change Install Location** to choose a new folder, or
* Click **Back** to go back a step.

6. Click **Install**.

The installation will begin. It may take a few minutes.

7. When your install completes, click **Finish**.

The installation completes, and now you're ready to start Splunk.

Start Splunk

Start Splunk

When you start Splunk, you're starting up two processes on your host, `splunkd` and `splunkweb`:

* `splunkd` is a distributed C/C++ server that accesses, processes and indexes streaming IT data and handles search requests.
* `splunkweb` is a Python-based application server that provides the Splunk Web interface that you use to search and navigate your IT data and manage your Splunk deployment.

Windows

To start Splunk on Windows, you have three options:

* start Splunk from the Start menu.
* use the Windows Services Manager to start and stop `splunkd` and `splunkweb`.
* open a cmd window and go to \Program Files\Splunk\bin and type

```
> splunk start
```

Mac OS X

Open a terminal or shell to access the CLI. Go to `/Applications/splunk/bin/`, and type:

```
$ ./splunk start
```

If you have administrator or root privileges you can simplify CLI usage by setting a Splunk environment variable. For more information about how to do this, read "About the CLI" in the *Admin manual*.

Accept the Splunk license

After you run the start command, Splunk displays the license agreement and prompts you to accept the license before the startup continues.

After you accept the license, the startup sequence displays. At the very end, Splunk tells you where to access Splunk Web:

```
The Splunk Web interface is at http://localhost:8000
```

If you run into any problems starting up Splunk, see Start Splunk for the first time in the *Installation manual*.

Other commands you might need

If you need to stop, restart, or check the status of your Splunk server, use these CLI commands:

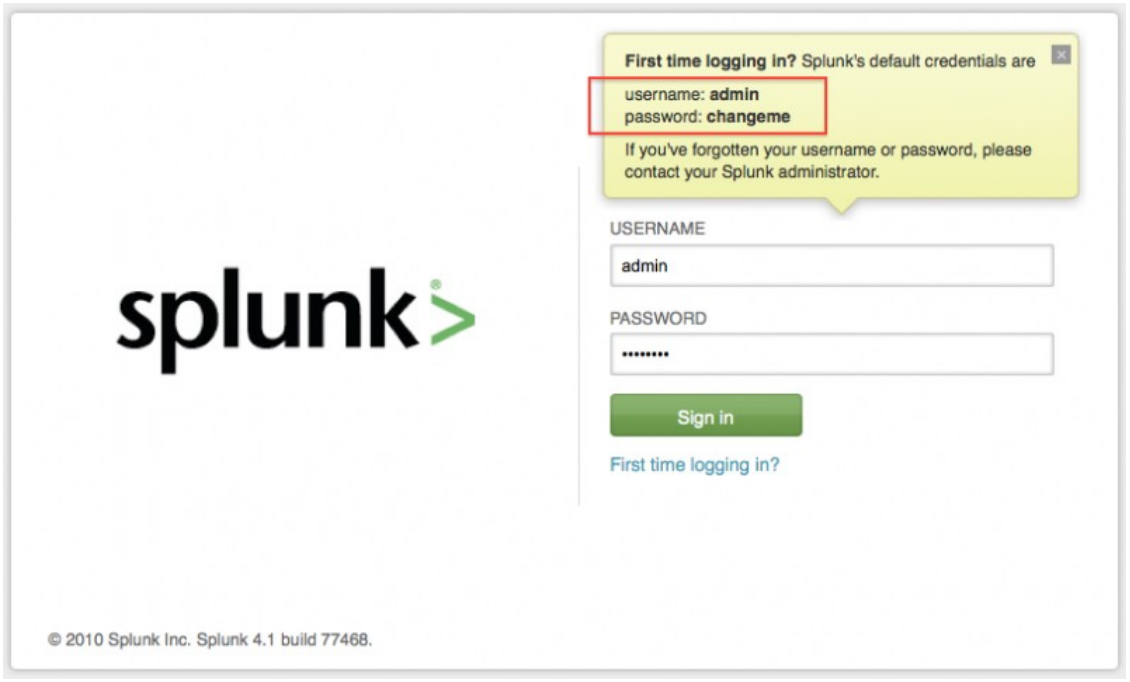
```
$ splunk stop
$ splunk restart
$ splunk status
```

Launch Splunk Web

Splunk's interface runs as a Web server and after starting up, Splunk tells you where the Splunk Web interface is. Open a browser and navigate to that location.

Splunk Web runs by default on port 8000 of the host on which it's installed. If you are using Splunk on your local machine, the URL to access Splunk Web is http://localhost:8000.

If you are using an Enterprise license, launching Splunk for the first time takes you to this login screen. Follow the message to authenticate with the default credentials:

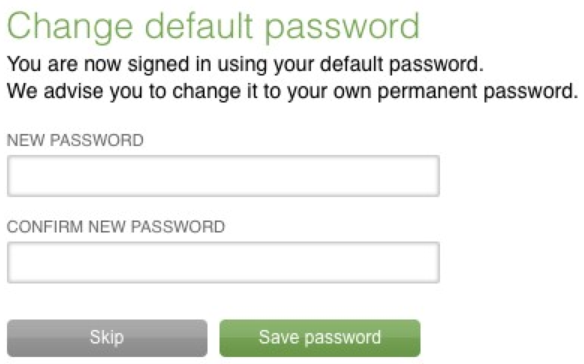


First time logging in? Splunk's default credentials are
username: **admin**
password: **changeme**
If you've forgotten your username or password, please contact your Splunk administrator.

USERNAME: admin
PASSWORD: ••••••••

If you are using a Free license, you do not need to authenticate to use Splunk. In this case, when you start up Splunk you won't see this login screen. Instead, you will be taken directly to Splunk Home or whatever is set as the default app for your account.

When you sign in with your default password, Splunk asks you to create a new password.


Change default password

You are now signed in using your default password.
We advise you to change it to your own permanent password.

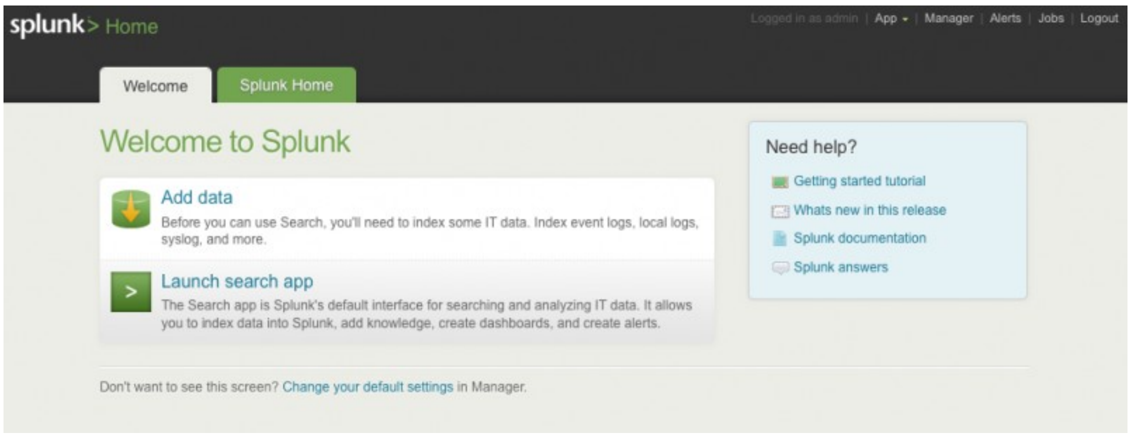
Once you've changed your password, you will be taken to Splunk Home.

Welcome to Splunk

When you log into Splunk for the first time, you should see Splunk Home. This app is designed to help you get started using Splunk. Before you can start using Splunk, you need to add some data.

The Welcome tab includes quick links to:

- *Add data*: this takes you to the interface where you can define data inputs.
- *Launch search app*: this takes you to Splunk's search interface, where you can start searching your data.


splunk> Home
Logged in as admin | App ▾ | Manager | Alerts | Jobs | Logout
Welcome
Splunk Home

Welcome to Splunk

Add data

Before you can use Search, you'll need to index some IT data. Index event logs, local logs, syslog, and more.

Launch search app

The Search app is Splunk's default interface for searching and analyzing IT data. It allows you to index data into Splunk, add knowledge, create dashboards, and create alerts.

Don't want to see this screen? Change your default settings in Manager.

Use the navigation menus at the upper right corner to access any of the apps and configuration pages for your Splunk server. This menu is available in any Splunk page, though not all of the same options will be there.

When you're ready, proceed to the next topic in this tutorial to Add data to Splunk.

Add data to Splunk

Add data to Splunk

This topic assumes that you have already downloaded, installed, and started a Splunk server. If you haven't yet, go back to the previous topic for instructions to do that.

Once you've started and logged into Splunk, you need to give it data that you can search. This topic walks you through downloading the sample dataset and adding it into Splunk.

Download the sample data file

This tutorial uses sample data from an online store, the Flower & Gift shop, to teach you about using Splunk. The sample data includes:

- Apache Web server logs
- mySQL database logs

You can feed Splunk data from files and directories, network ports, and custom scripts, but for this tutorial, you will upload a compressed file directly to Splunk.

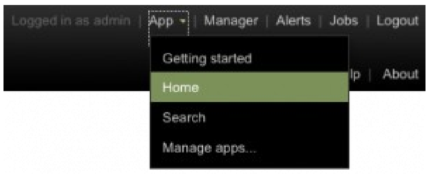
(You can of course put lots of other kinds of data into Splunk--check out "What Splunk can index" in the Getting Data In Manual for information about the different sorts of data Splunk can handle.)

To proceed with this tutorial, **download (but do not uncompress) the sample data from here: sampledata.zip** *This sample data file is updated daily*.

Important: For this tutorial, you will be adding the compressed file to Splunk, so you do not need to uncompress the sample data! Also, this tutorial is designed to be completed in a matter of hours. But, if you want to spread it out over a few days, just download a new sample data file and add it!

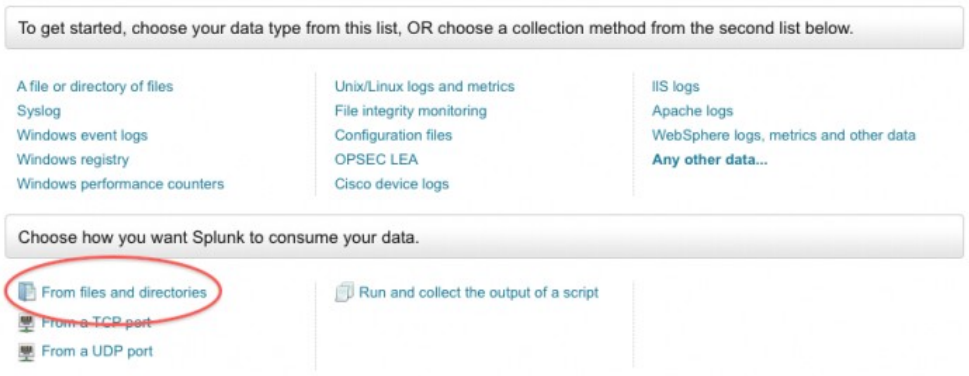
Add the sample data to Splunk

Logging into Splunk should have taken you to Splunk Home. If it isn't the first view that you see, use the App menu to select Home.



1. In Splunk Home, click **Add data**.

2. Under **Choose how you want Splunk to consume your data**, click **From files and directories**.


To get started, choose your data type from this list, OR choose a collection method from the second list below.

A file or directory of files
Syslog
Windows event logs
Windows registry
Windows performance counters

Unix/Linux logs and metrics
File integrity monitoring
Configuration files
OPSEC LEA
Cisco device logs

IIS logs
Apache logs
WebSphere logs, metrics and other data
Any other data...

Choose how you want Splunk to consume your data.

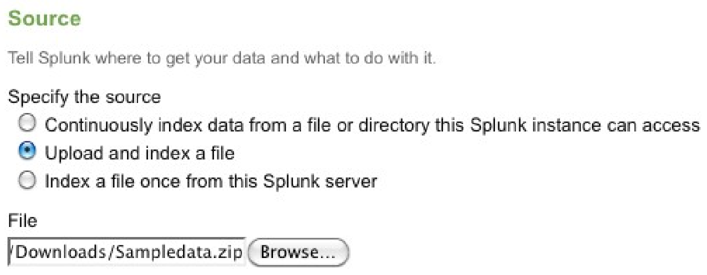
From files and directories
From a TCP port
From a UDP port

Run and collect the output of a script

This takes you to the **Home > Add data > Files & Directories > Add new** view. This is where you will upload the sample data file. Normally, this is all you need to do and Splunk handles the rest without any changes needed. For the purposes of this tutorial, however, you will also edit some of the properties.

3. Under **Source**, select **Upload and index a file** and browse for the sample data file that you just downloaded.

The `source` of an event tells you where it came from. If you collect data from files and directories, the "source" is the full pathname of the file or directory. In the case of a network-based source, the source is the protocol and port, such as UDP:514.


Source

Tell Splunk where to get your data and what to do with it.

Specify the source
○ Continuously index data from a file or directory this Splunk instance can access
● Upload and index a file
○ Index a file once from this Splunk server

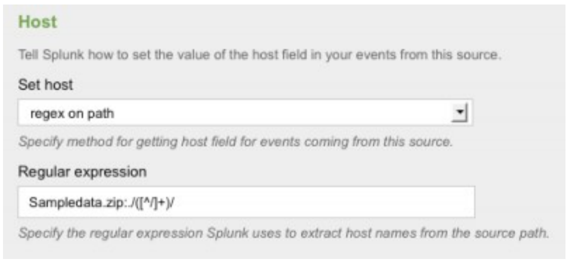
File
/Downloads/Sampledata.zip

4. Select **More settings**.

5. Under **Host** and **Set host**, choose *regex on path*.

An event's `host` value is typically the hostname, IP address, or fully qualified domain name of the network host from which the event originated.

By selecting *regex on path*, you're telling Splunk that you want to set a custom host value for your data using a regular expression (regex) to match a part of the pathname to the source file (that you browsed for earlier).


Host

Tell Splunk how to set the value of the host field in your events from this source.

Set host
regex on path

Specify method for getting host field for events coming from this source.

Regular expression
Sampledata.zip:./([^/]+)/

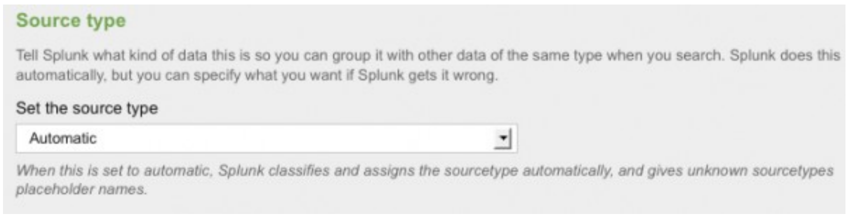
Specify the regular expression Splunk uses to extract host names from the source path.

6. Under **Regular expression**, copy and paste:

```
Sampledata.zip:./([^/]+)/
```

7. Leave the value of **Source type** as "Automatic".

The source type of an event tells you what kind of data it is, usually based on how it's formatted. Examples of source types are *access_combined* or *cisco_syslog*. This classification lets you search for the same type of data across multiple sources and hosts.



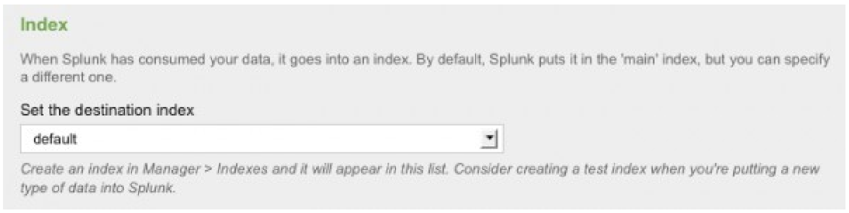
Source type

Tell Splunk what kind of data this is so you can group it with other data of the same type when you search. Splunk does this automatically, but you can specify what you want if Splunk gets it wrong.

Set the source type: Automatic

When this is set to automatic, Splunk classifies and assigns the sourcetype automatically, and gives unknown sourcetypes placeholder names.

8. Under **Index**, leave the destination index as **default**.



Index

When Splunk has consumed your data, it goes into an index. By default, Splunk puts it in the 'main' index, but you can specify a different one.

Set the destination index: default

Create an index in Manager > Indexes and it will appear in this list. Consider creating a test index when you're putting a new type of data into Splunk.

9. Click **Save**.

When it's finished, Splunk displays a message saying the upload was successful. Click **Start searching** and proceed to the next topic in this tutorial to look at your data in the Search app.

More about getting data in

This topic only discusses one type of input, uploading a local file, which is all you need to run through the tutorial. For information about all other type of data inputs Splunk can handle and how to add them, refer to the *Getting Data In Manual* beginning with the topic, "What Splunk can monitor".

The Search app

The Search app

This topic assumes you've just added the sample data for the online Flower & Gift shop. If you haven't, go back to the add data tutorial to get it before proceeding.

Once you have data in Splunk, you're ready to start searching. This topic introduces you to the Search app, which is Splunk's default interface for searching and analyzing data. If you're already familiar with the search interface, you can skip ahead and start searching.

You are a member of the Customer Support team for the online **Flower & Gift** shop. This is your first day on the job. You want to learn some more about the shop. Some questions you want answered are:

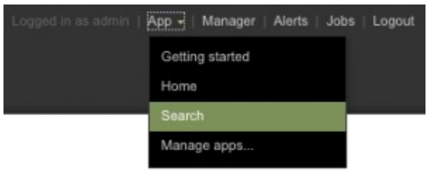
- What does the store sell? How much does each item cost?
- How many people visited the site? How many bought something today?
- What is the most popular item that is purchased each day?

The Splunk server already has data in it--let's take a look at it.

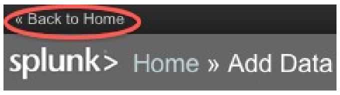
Find the Search app

You can access the Search app from anywhere in Splunk.

To get to the Search app from Home or another app or view, use the App menu at the upper right corner to select the **Search** app:



If the App menu is not available, select **Home** in the App menu or click the **<< Back to Home** link at the top left corner of the page:

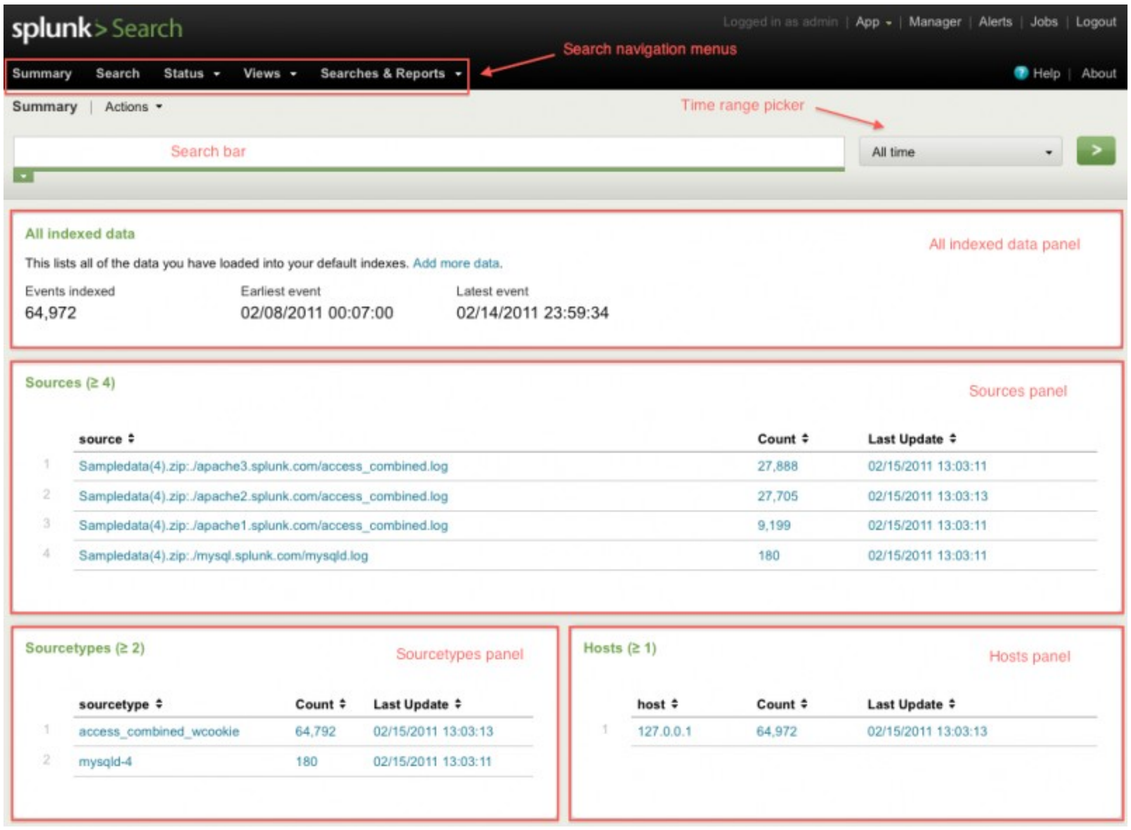


Once you're back in Home, select **Search** from the App menu.

The first view that you see in the Search app is the Summary dashboard.

The Summary dashboard

The Search app's Summary dashboard displays information about the data that you just uploaded to this Splunk server and gives you the means to start searching this data.



All indexed data

This lists all of the data you have loaded into your default indexes. Add more data.

Events indexed	Earliest event	Latest event
64,972	02/08/2011 00:07:00	02/14/2011 23:59:34

Sources (≥ 4)

	source	Count	Last Update
1	Sampledata(4).zip:./apache3.splunk.com/access_combined.log	27,888	02/15/2011 13:03:11
2	Sampledata(4).zip:./apache2.splunk.com/access_combined.log	27,705	02/15/2011 13:03:13
3	Sampledata(4).zip:./apache1.splunk.com/access_combined.log	9,199	02/15/2011 13:03:11
4	Sampledata(4).zip:./mysql.splunk.com/mysqld.log	180	02/15/2011 13:03:11

Sourcetypes (≥ 2)

	sourcetype	Count	Last Update
1	access_combined_wcookie	64,792	02/15/2011 13:03:13
2	mysqld-4	180	02/15/2011 13:03:11

Hosts (≥ 1)

	host	Count	Last Update
1	127.0.0.1	64,972	02/15/2011 13:03:13

The metrics displayed on this dashboard are generated by saved searches that run behind-the-scenes whenever you access and reload this page. (By the end of this tutorial, you'll be able to run searches, save them, and use them to build your own dashboard, much like this one.)

What's in this dashboard?

The Search app includes many different dashboards and views. For now, you really only need to know about two of them:

- *Summary*, where you are now
- *Search*, where you will do most of your searching

Use the **Search navigation menus** to locate and access the different views in the app. When you click on the links, Splunk takes you to the respective dashboards or refreshes the page if you're already there.

Other things in the Search app UI:

- **Searches & Reports:** lists all of your saved searches and reports.
- **Search bar and Time range picker:** enables you to type in your search and select different time ranges over which to retrieve events.
- **All indexed data panel:** displays metrics about your indexed event data. which include the total number of events you have in your Splunk index(es) and the timestamps of the earliest and latest indexed event. It also tells you when this data was last refreshed (or when you last reloaded this dashboard).
- **Sources panel:** displays the top sources from the data on your Splunk server.

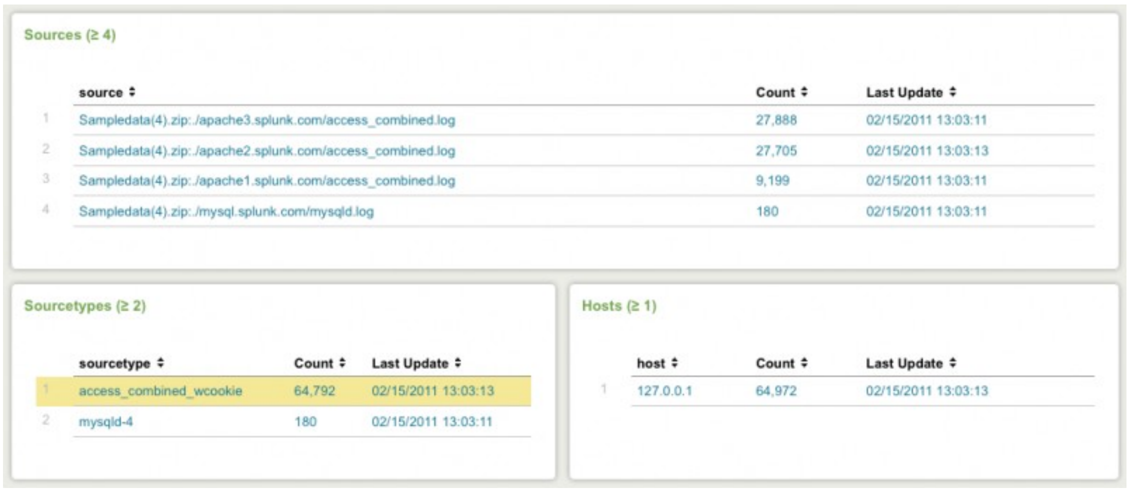
- **Sourcetypes panel:** displays the top source types from your Splunk server's data.
- **Hosts:** displays the top hosts from your Splunk server's data.

If you're using a freshly installed Splunk server for this tutorial, you'll only see the sample data files that you just uploaded. Because it's a one-time upload of a file, this data will not change. When you add more data, there will be more information on this dashboard. If you add data inputs that point to sources that are not static (such as log files that are being written to by applications), the numbers on the Summary page will change as more data comes in from your source(s).

If you're using a shared or pre-installed Splunk server that is deployed in an enterprise environment, you'll probably see much more information on this dashboard.

Kick off a search

1. Take a closer look at the **Summary** dashboard.



Sources (≥ 4)

	source	Count	Last Update
1	Sampledata(4).zip:./apache3.splunk.com/access_combined.log	27,888	02/15/2011 13:03:11
2	Sampledata(4).zip:./apache2.splunk.com/access_combined.log	27,705	02/15/2011 13:03:13
3	Sampledata(4).zip:./apache1.splunk.com/access_combined.log	9,199	02/15/2011 13:03:11
4	Sampledata(4).zip:./mysql.splunk.com/mysqld.log	180	02/15/2011 13:03:11

Sourcetypes (≥ 2)

	sourcetype	Count	Last Update
1	access_combined_wcookie	64,792	02/15/2011 13:03:13
2	mysqld-4	180	02/15/2011 13:03:11

Hosts (≥ 1)

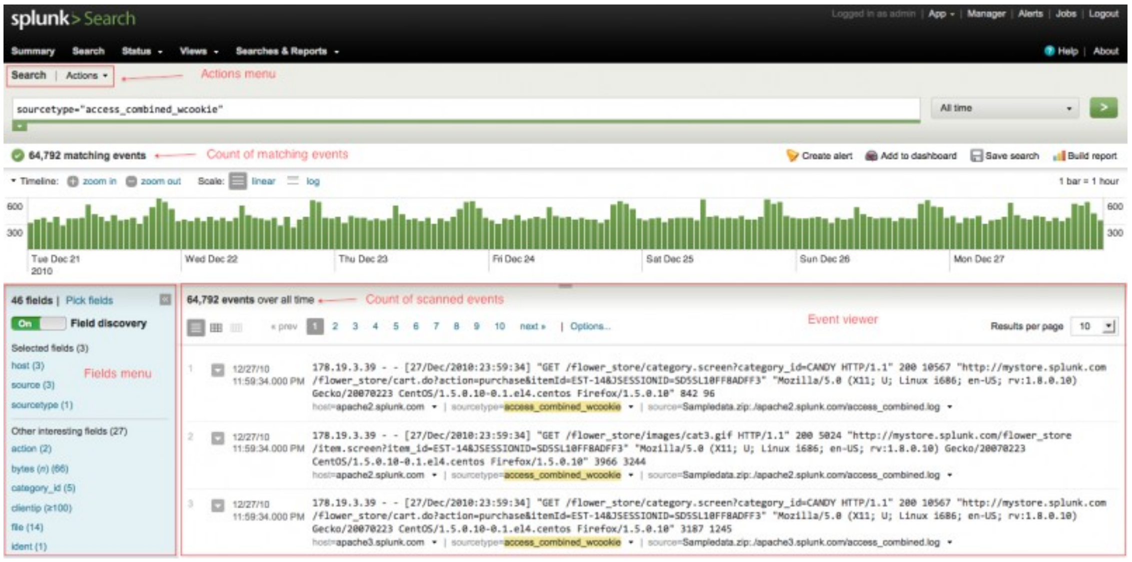
	host	Count	Last Update
1	127.0.0.1	64,972	02/15/2011 13:03:13

In the *Sources* panel, you should see three Apache Web server logs and a mySQL database log for the online Flower & Gift shop data that you just uploaded. If you're familiar with Apache Web server logs, you might recognize the *access_combined_wcookie* **Source type** as one of the log formats associated with Web access logs. All the data for this source type should give you information about people who access the Flower & Gift shop website.

Searching in Splunk is very interactive. Although you have a search bar in the Summary dashboard, you don't need to type anything into it just yet. Each of the sources, sourcetypes, and hosts listed in the **Summary** dashboard is a link that will kick off a search when you click on them.

2. In the *Sourcetypes* panel, click `access_combined_wcookie`.

Splunk takes you to the Search dashboard, where it runs the search and shows you the results:



```
sourcetype="access_combined_wcookie"
```

```
1  12/27/10         178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/category.screen?category_id=CANDY HTTP/1.1" 200 10567 "http://mystore.splunk.com
   11:59:34.000 PM  /flower_store/cart.do?action=purchase&itemId=EST-14&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
                    Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 842 96
                    host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log

2  12/27/10         178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/images/cat3.gif HTTP/1.1" 200 5024 "http://mystore.splunk.com/flower_store
   11:59:34.000 PM  /item.screen?item_id=EST-14&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223
                    CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3966 3244
                    host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log

3  12/27/10         178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/category.screen?category_id=CANDY HTTP/1.1" 200 10567 "http://mystore.splunk.com
   11:59:34.000 PM  /flower_store/cart.do?action=purchase&itemId=EST-14&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
                    Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3187 1245
                    host=apache3.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache3.splunk.com/access_combined.log
```

There are a lot of components to this view, so let's take a look at them before continuing to search.

Splunk paused my search?

If you are searching on a Splunk installation that has more data on it than just this tutorial's sample data, your search might take a bit longer. If your search takes longer than 30 seconds, Splunk will automatically pause it. If autopause pops up, click **Resume search**. You can read more about autopause in the *Admin manual*.

What's in this Search dashboard?

The search bar and time range picker should be familiar to you -- it was also in the Summary dashboard. But, now you also see a count of events, the timeline, the fields menu, and the list of retrieved events or search results.

- **Actions menu:** Use this menu to save a search or report after you run it, create a dashboard, or print your report to a PDF file.
- **Count of matching and scanned events:** As the search runs, Splunk displays two running counts of the events as it retrieves them: one is a matching event count and the other is the count of events scanned. When the search completes, the count that appears above the timeline displays the total number of matching events. The count that appears below the timeline and above the events list, tells you the number of events during the time range that you selected. As we'll see later, this number changes when you drill down into your investigations.
- **Timeline of events:** The timeline is a visual representation of the number of events that occur at each point in time. As the timeline updates with your search results, you might notice clusters or patterns of bars. The height of each bar indicates the count of events. Peaks or valleys in the timeline can indicate spikes in activity or server downtime. Thus, the timeline is useful for highlighting patterns of events or investigating peaks and lows in event activity. The timeline options are located above the timeline. You can zoom in, zoom out, and change the scale of the chart.
- **Fields menu:** We mentioned before that when you index data, Splunk by default automatically recognizes and extracts information from your data that is formatted as name and value pairs,

which we call fields. When you run a search, Splunk lists all of the fields it recognizes in the Fields menu next to your search results. You can select other fields to show in your events.
    - *Selected fields* are fields that are set to be visible in your search results. By default, host, source, and sourcetype are shown.
    - *Other interesting fields* are other fields that Splunk has extracted from your search results.
    - *Field discovery* is an on/off switch at the top of the Fields menu. Splunk's default setting is Field discovery on. If you want to speed up your search, you can turn Field discovery off, and Splunk will extract only the fields required to complete your search.
- **Event viewer:** The event viewer displays the events that Splunk retrieves to match your search. It's located below the timeline. By default the events are displayed as a list, but you can also choose to view them as a table. When you select the event table view, you will only see the *Selected fields* in your table.

When you're ready, proceed to the next topic to start searching and find out what's up at the flower shop.

Start searching

Start searching

This topic walks you through simple searches using the Search interface. If you're not familiar with the search interface, go back to the Search app tutorial before proceeding.

It's your first day of work with the Customer Support team for the online Flower & Gift shop. You're just starting to dig into the Web access logs for the shop, when you receive a call from a customer who complains about trouble buying a gift for his girlfriend--he keeps hitting a server error when he tries to complete a purchase. He gives you his IP address, 10.2.1.44.

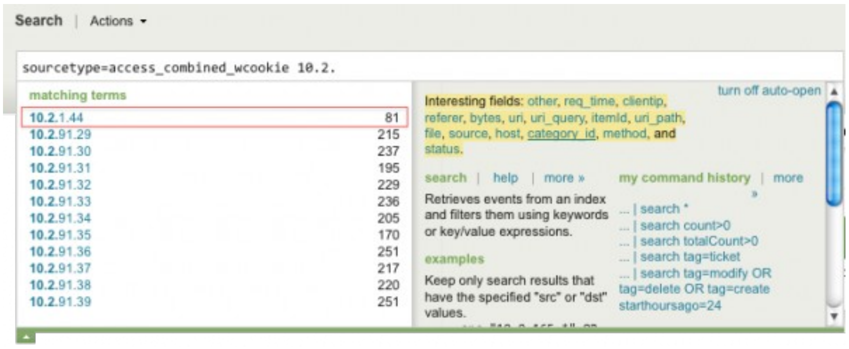
Typeahead for keywords

Everything in Splunk is searchable. You don't have to be familiar with the information in your data because searching in Splunk is free-form and as simple as typing keywords into the search bar and hitting **Enter** (or clicking that green arrow at the end of the search bar).

In the previous topic, you ran a search from the Summary dashboard by clicking on the Web access source type (`access_combined_wcookie`). Use that same search to find this customer's recent access history at the online Flower & Gift shop.

1. Type the customer's IP address into the search bar:

```
sourcetype=access_combined_wcookie 10.2.1.44
```
As you type into the search bar, Splunk's search assistant opens.



```
sourcetype=access_combined_wcookie 10.2.
```

matching terms	
10.2.1.44	81
10.2.91.29	215
10.2.91.30	237
10.2.91.31	195
10.2.91.32	229
10.2.91.33	236
10.2.91.34	205
10.2.91.35	170
10.2.91.36	251
10.2.91.37	217
10.2.91.38	220
10.2.91.39	251

Interesting fields: other, req_time, clientip, referer, bytes, uri, uri_query, itemId, uri_path, file, source, host, category_id, method, and status.

Retrieves events from an index and filters them using keywords or key/value expressions.

examples

Keep only search results that have the specified "src" or "dst" values.

```
... | search *
... | search count>0
... | search totalCount>0
... | search tag=ticket
... | search tag=modify OR tag=delete OR tag=create starthoursago=24
```

Search assistant shows you typeahead, or contextual matches and completions for each keyword as you type it into the search bar. These contextual matches are based on what's in your data. The entries under **matching terms** update as you continue to type because the possible completions for your term changes as well.

Search assistant also displays the number of matches for the search term. This number gives you an idea of how many search results Splunk will return. If a term or phrase doesn't exist in your data, you won't see it listed in search assistant.

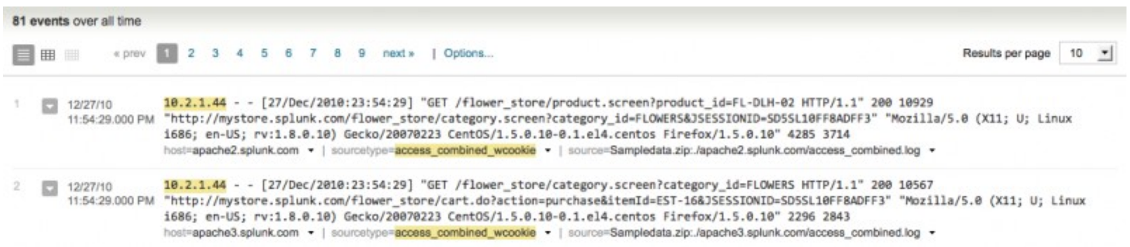
What else do you see in search assistant?

For now, ignore everything on the right panel next to the contextual help. Search assistant has more uses once you start learning the search language, as you'll see later. And, if you don't want search assistant to open, click "turn off auto-open" and close the window using the green arrow below the search bar.

More keyword searches

2. If you didn't already, run the search for the IP address. (Hit *Enter*.)

Splunk retrieves the customer's access history for the online Flower & Gift shop.



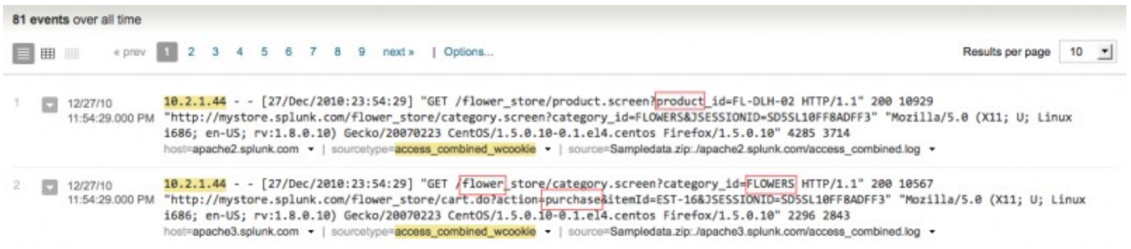
81 events over all time

```
1  12/27/10          10.2.1.44 - - [27/Dec/2010:23:54:29] "GET /flower_store/product.screen?product_id=FL-DLH-02 HTTP/1.1" 200 10929
   11:54:29.000 PM   "http://mystore.splunk.com/flower_store/category.screen?category_id=FLOWERS&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux
                     i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 4285 3714
                     host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log

2  12/27/10          10.2.1.44 - - [27/Dec/2010:23:54:29] "GET /flower_store/category.screen?category_id=FLOWERS HTTP/1.1" 200 10567
   11:54:29.000 PM   "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-16&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux
                     i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2296 2843
                     host=apache3.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache3.splunk.com/access_combined.log
```

Each time you run a search, Splunk highlights in the search results what you typed into the search bar.

3. Skim through the search results.

You should recognize words and phrases in the events that relate to the online shop (flower, product, purchase, etc.).



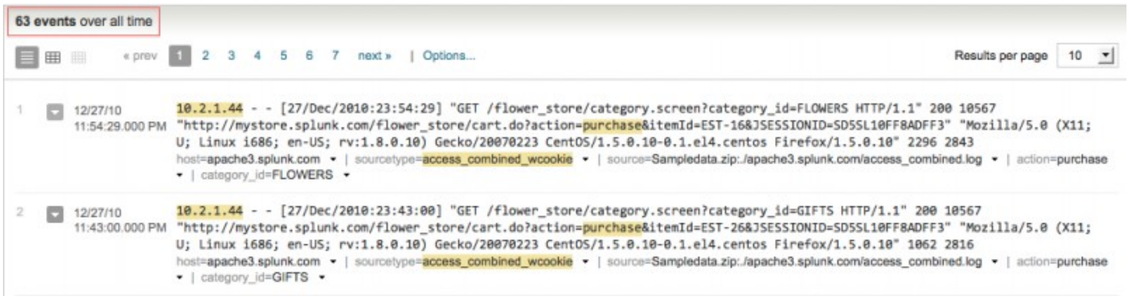
```
1  12/27/10            10.2.1.44 - - [27/Dec/2010:23:54:29] "GET /flower_store/product.screen?product_id=FL-DLH-02 HTTP/1.1" 200 10929
   11:54:29.000 PM     "http://mystore.splunk.com/flower_store/category.screen?category_id=FLOWERS&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux
                       i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 4285 3714
                       host=apache2.splunk.com ▾ | sourcetype=access_combined_wcookie ▾ | source=Sampledata.zip:./apache2.splunk.com/access_combined.log ▾

2  12/27/10            10.2.1.44 - - [27/Dec/2010:23:54:29] "GET /flower_store/category.screen?category_id=FLOWERS HTTP/1.1" 200 10567
   11:54:29.000 PM     "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-16&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux
                       i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2296 2843
                       host=apache3.splunk.com ▾ | sourcetype=access_combined_wcookie ▾ | source=Sampledata.zip:./apache3.splunk.com/access_combined.log ▾
```

The customer mentioned that he was in the middle of purchasing a gift, so lets see what we find by searching for "purchase".

4. Type *purchase* into the search bar and run the search:

```
sourcetype=access_combined_wcookie 10.2.1.44 purchase
```

When you search for keywords, your search is not case-sensitive and Splunk retrieves the events that contain those keywords anywhere in the raw text of the event's data.

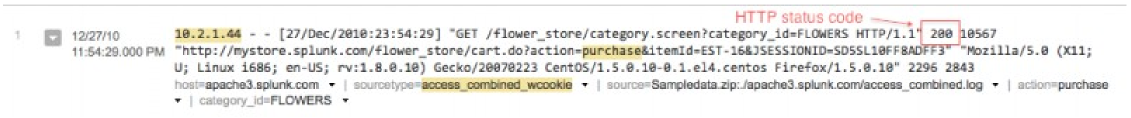


```
1  12/27/10            10.2.1.44 - - [27/Dec/2010:23:54:29] "GET /flower_store/category.screen?category_id=FLOWERS HTTP/1.1" 200 10567
   11:54:29.000 PM     "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-16&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11;
                       U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2296 2843
                       host=apache3.splunk.com ▾ | sourcetype=access_combined_wcookie ▾ | source=Sampledata.zip:./apache3.splunk.com/access_combined.log ▾ | action=purchase
                       ▾ | category_id=FLOWERS ▾

2  12/27/10            10.2.1.44 - - [27/Dec/2010:23:43:00] "GET /flower_store/category.screen?category_id=GIFTS HTTP/1.1" 200 10567
   11:43:00.000 PM     "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-26&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11;
                       U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 1062 2816
                       host=apache3.splunk.com ▾ | sourcetype=access_combined_wcookie ▾ | source=Sampledata.zip:./apache3.splunk.com/access_combined.log ▾ | action=purchase
                       ▾ | category_id=GIFTS ▾
```

Among the results that Splunk retrieves are events that show each time the customer tried to buy something from the online store. Looks like he's been busy!

Use Boolean Operators

If you're familiar with Apache server logs, in this case the access_combined format, you'll notice that most of these events have an HTTP status of 200, or Successful. These events are not interesting for you right now, because the customer is reporting a problem.

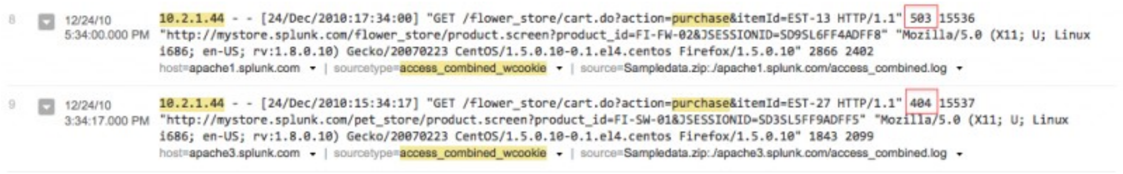


```
1  12/27/10            10.2.1.44 - - [27/Dec/2010:23:54:29] "GET /flower_store/category.screen?category_id=FLOWERS HTTP/1.1" 200 10567
   11:54:29.000 PM     "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-16&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11;
                       U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2296 2843
                       host=apache3.splunk.com ▾ | sourcetype=access_combined_wcookie ▾ | source=Sampledata.zip:./apache3.splunk.com/access_combined.log ▾ | action=purchase
                       ▾ | category_id=FLOWERS ▾
```

5. Use the Boolean NOT operator to quickly remove all of these Successful page requests. Type in:

```
sourcetype=access_combined_wcookie 10.2.1.44 purchase NOT 200
```

You notice that the customer is getting HTTP server (503) and client (404) errors.



```
8  12/24/10           10.2.1.44 - - [24/Dec/2010:17:34:00] "GET /flower_store/cart.do?action=purchase&itemId=EST-13 HTTP/1.1" 503 15536
   5:34:00.000 PM     "http://mystore.splunk.com/flower_store/product.screen?product_id=FI-FW-02&JSESSIONID=SD9SL6FF4ADFF8" "Mozilla/5.0 (X11; U; Linux
                      i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2866 2402
                      host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log

9  12/24/10           10.2.1.44 - - [24/Dec/2010:15:34:17] "GET /flower_store/cart.do?action=purchase&itemId=EST-27 HTTP/1.1" 404 15537
   3:34:17.000 PM     "http://mystore.splunk.com/pet_store/product.screen?product_id=FI-SW-01&JSESSIONID=SD3SL5FF9ADFF5" "Mozilla/5.0 (X11; U; Linux
                      i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 1843 2099
                      host=apache3.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache3.splunk.com/access_combined.log
```

But, he specifically mentioned a server error, so you want to quickly remove events that are irrelevant.

Splunk supports the Boolean operators: AND, OR, and NOT. When you include Boolean expressions in your search, the operators have to be capitalized.

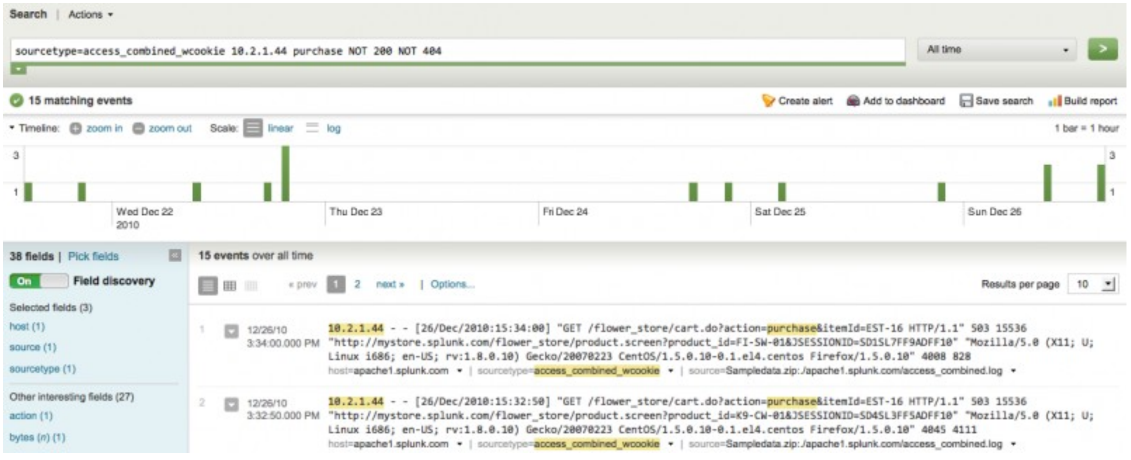
The AND operator is always implied between search terms. So the search in Step 5 is the same as:

```
sourcetype=access_combined_wcookie AND 10.2.1.44 AND purchase NOT 200
```

Another way to add Boolean clauses quickly and interactively to your search is to use your search results.

6. Mouse-over an instance of "404" in your search results and *ALT-click* it.

This updates your search string with "NOT 404" and filters out all the events that contain the term.



```
sourcetype=access_combined_wcookie 10.2.1.44 purchase NOT 200 NOT 404
```

```
1  12/26/10           10.2.1.44 - - [26/Dec/2010:15:34:00] "GET /flower_store/cart.do?action=purchase&itemId=EST-16 HTTP/1.1" 503 15536
   3:34:00.000 PM     "http://mystore.splunk.com/flower_store/product.screen?product_id=FI-SW-01&JSESSIONID=SD1SL7FF9ADFF10" "Mozilla/5.0 (X11; U;
                      Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 4008 828
                      host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log

2  12/26/10           10.2.1.44 - - [26/Dec/2010:15:32:50] "GET /flower_store/cart.do?action=purchase&itemId=EST-16 HTTP/1.1" 503 15536
   3:32:50.000 PM     "http://mystore.splunk.com/flower_store/product.screen?product_id=K9-CW-01&JSESSIONID=SD4SL3FF5ADFF10" "Mozilla/5.0 (X11; U;
                      Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 4045 4111
                      host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log
```

From these results, you see each time that the customer attempted to complete a purchase and received the server error. Now that you have confirmed what the customer reported, you can continue to drill down to find the root cause.

Interactive searching

Splunk lets you highlight and select any segment from within your search results to add, remove, and exclude them quickly and interactively using your keyboard and mouse:

- To add more search terms, highlight and click the word or phrase you want from your search results.
- To remove a term from your search, click a highlighted instance of that word or phrase in your search results.
- To exclude events from your search results, ALT-click on the term you don't want Splunk to match.

When you're ready to proceed, go to the next topic to learn how to investigate and troubleshoot interactively using the timeline in Splunk.

Use the timeline

Use the timeline

This topic assumes that you're comfortable running simple searches to retrieve events. If you're not sure, go back to the last topic where you searched with keywords, wildcards, and Booleans to pinpoint an error.

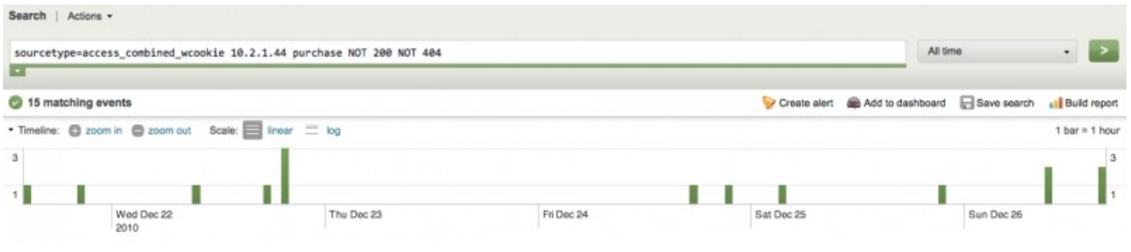
Back at the Flower & Gift shop, let's continue with the customer (10.2.1.44) you were assisting. He reported an error while purchasing a gift for his girlfriend. You confirmed his error, and now you want to find the cause of it.

Continue with the last search, which showed you the customer's failed purchase attempts.

1. Search for:

```
sourcetype=access_combined_wcookie 10.2.1.44 purchase NOT 200 NOT 404
```

In the last topic, you really just focused on the search results listed in the events viewer area of this dashboard. Now, let's take a look at the timeline.



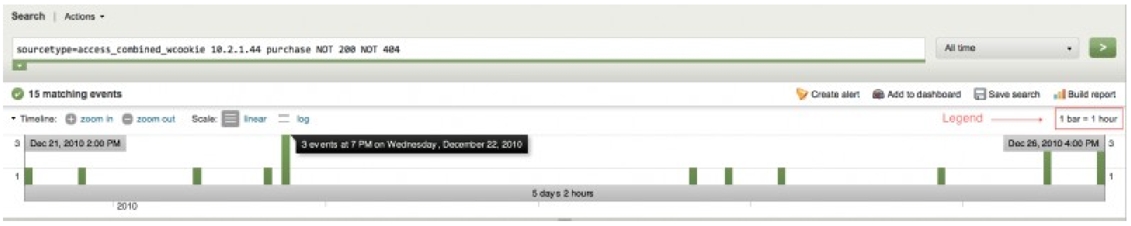
```
sourcetype=access_combined_wcookie 10.2.1.44 purchase NOT 200 NOT 404
```

15 matching events

The location of each bar on the timeline corresponds to an instance when the events that match your search occurred. If there are no bars at a time period, no events were found then.

2. Mouse over one of the bars.

A tooltip pops up and displays the number of events that Splunk found during the time span of that bar (1 bar = 1 hr).

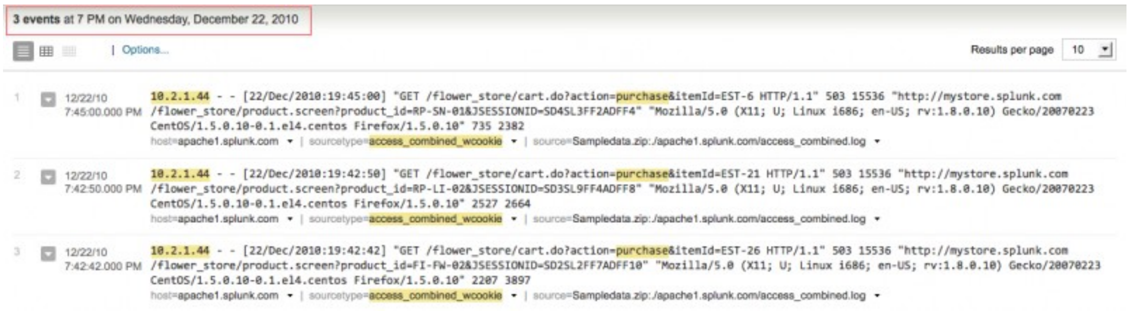


```
sourcetype=access_combined_wcookie 10.2.1.44 purchase NOT 200 NOT 404
```

The taller the bar, the more events occurred at that time. Often seeing spikes in the number of events or no events is a good indication that something has happened.

3. *Click* one of the bars, for example the tallest bar.

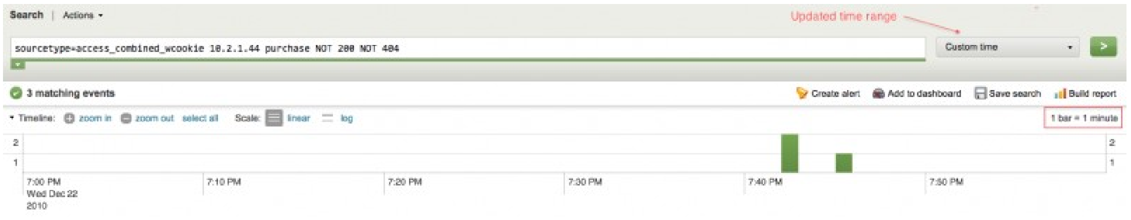
This updates your search results to show you only the events at the time span. Splunk does not run the search when you click on the bar. Instead, it gives you a preview of the results zoomed-in at the time range. You can still select other bars at this point.



```
1  12/22/10           10.2.1.44 - - [22/Dec/2010:19:45:00] "GET /flower_store/cart.do?action=purchase&itemId=EST-6 HTTP/1.1" 503 15536 "http://mystore.splunk.com
   7:45:00.000 PM     /flower_store/product.screen?product_id=RP-SN-01&JSESSIONID=SD4SL3FF2ADFF4" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223
                      CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 735 2382
                      host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log

2  12/22/10           10.2.1.44 - - [22/Dec/2010:19:42:50] "GET /flower_store/cart.do?action=purchase&itemId=EST-21 HTTP/1.1" 503 15536 "http://mystore.splunk.com
   7:42:50.000 PM     /flower_store/product.screen?product_id=RP-LI-02&JSESSIONID=SD3SL9FF4ADFF8" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223
                      CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2527 2664
                      host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log

3  12/22/10           10.2.1.44 - - [22/Dec/2010:19:42:42] "GET /flower_store/cart.do?action=purchase&itemId=EST-26 HTTP/1.1" 503 15536 "http://mystore.splunk.com
   7:42:42.000 PM     /flower_store/product.screen?product_id=FI-FW-02&JSESSIONID=SD2SL2FF7ADFF10" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223
                      CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2207 3897
                      host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log
```

4. *Double-click* on the same bar.

Splunk re-runs your search to retrieve only events during that one hour span you selected.



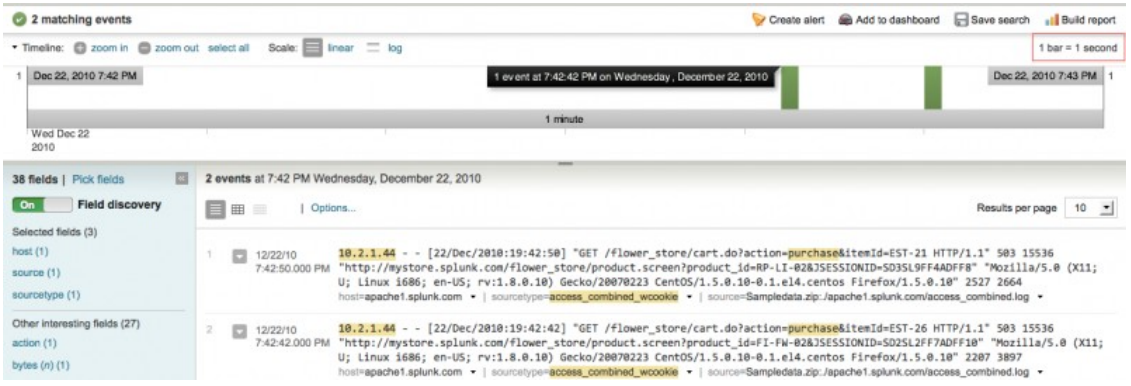
```
sourcetype=access_combined_wcookie 10.2.1.44 purchase NOT 200 NOT 404
```

You should see the same search results in the Event viewer, but, notice that the search overrides the time range picker and it now shows "Custom time". (You'll see more of the time range picker later.) Also, each bar now represents one minute of time (1 bar = 1 min).

One hour is still a wide time period to search, so let's narrow the search down more.

5. Double-click another bar.

Once again, this updates your search to now retrieve events during that one minute span of time. Each bar represents the number of events for one second of time.



2 events at 7:42 PM Wednesday, December 22, 2010

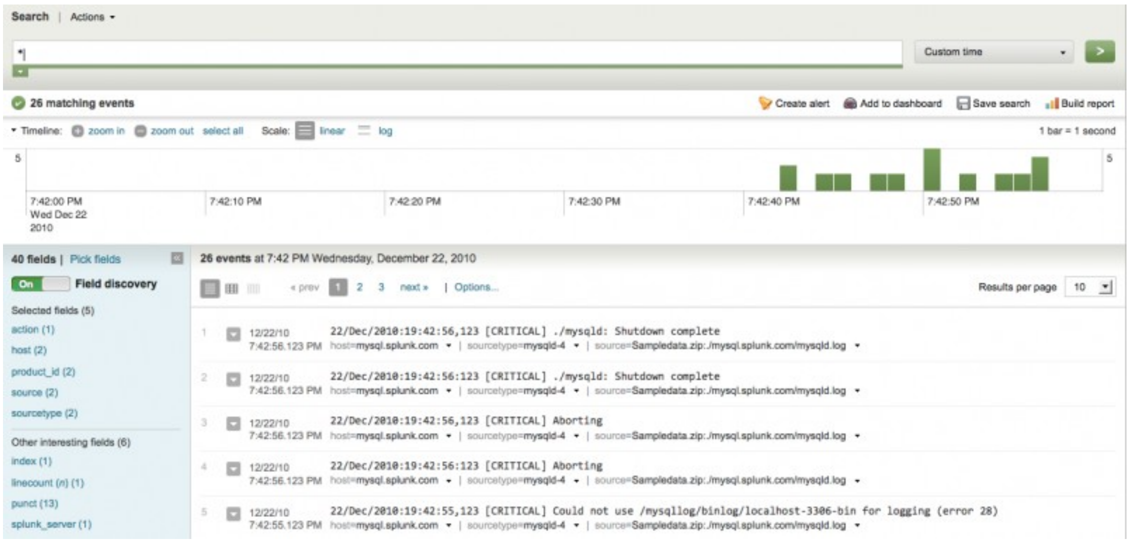
```
1  12/22/10          10.2.1.44 - - [22/Dec/2010:19:42:50] "GET /flower_store/cart.do?action=purchase&itemId=EST-21 HTTP/1.1" 503 15536
   7:42:50.000 PM    "http://mystore.splunk.com/flower_store/product.screen?product_id=RP-LI-02&JSESSIONID=SD3SL9FF4ADFF8" "Mozilla/5.0 (X11;
                     U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2527 2664
                     host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log

2  12/22/10          10.2.1.44 - - [22/Dec/2010:19:42:42] "GET /flower_store/cart.do?action=purchase&itemId=EST-26 HTTP/1.1" 503 15536
   7:42:42.000 PM    "http://mystore.splunk.com/flower_store/product.screen?product_id=FI-FW-02&JSESSIONID=SD2SL2FF7ADFF10" "Mozilla/5.0 (X11;
                     U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2207 3897
                     host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log
```

Now, you want to expand your search to see everything else, if anything happened during this minute.

6. Without changing the time range, replace your previous search in the search bar with:

```
*
```

Splunk supports using the asterisk (*) wildcard to search for "all" or to retrieve events based on parts of a keyword. Up to now, you've just searched for Web access logs. This search tells Splunk that you want to see everything that occurred at this time range:

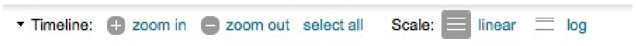


26 events at 7:42 PM Wednesday, December 22, 2010

```
1  12/22/10          22/Dec/2010:19:42:56,123 [CRITICAL] ./mysqld: Shutdown complete
   7:42:56.123 PM    host=mysql.splunk.com | sourcetype=mysqld-4 | source=Sampledata.zip:./mysql.splunk.com/mysqld.log

2  12/22/10          22/Dec/2010:19:42:56:123 [CRITICAL] ./mysqld: Shutdown complete
   7:42:56.123 PM    host=mysql.splunk.com | sourcetype=mysqld-4 | source=Sampledata.zip:./mysql.splunk.com/mysqld.log

3  12/22/10          22/Dec/2010:19:42:56,123 [CRITICAL] Aborting
   7:42:56.123 PM    host=mysql.splunk.com | sourcetype=mysqld-4 | source=Sampledata.zip:./mysql.splunk.com/mysqld.log

4  12/22/10          22/Dec/2010:19:42:56:123 [CRITICAL] Aborting
   7:42:56.123 PM    host=mysql.splunk.com | sourcetype=mysqld-4 | source=Sampledata.zip:./mysql.splunk.com/mysqld.log

5  12/22/10          22/Dec/2010:19:42:55,123 [CRITICAL] Could not use /mysqllog/binlog/localhost-3306-bin for logging (error 28)
   7:42:55.123 PM    host=mysql.splunk.com | sourcetype=mysqld-4 | source=Sampledata.zip:./mysql.splunk.com/mysqld.log
```

This search returns events from all the logs on your server. You expect to see other user's Web activity--perhaps from different hosts. But instead you see a cluster of mySQL database errors. These errors were causing your customer's purchases to fail. Now, you can report this issue to someone in the IT Operations team.

What else can you do with the timeline?



- To show all the results for the timeline again, click *select all* above the timeline.
- To lock-in the selected span of events to your search, click *zoom in*.
- To expand the timeline view to show more events, click *zoom out*.

When you're ready, proceed to the next topic to learn about searching over different time ranges.

Change the time range

Change the time range

This topic assumes that you're familiar with running ad hoc searches and using the timeline. If you're not sure, review the previous topics on searching and using the timeline.

This topic shows you how to narrow the scope of your investigative searching over any past time range. If you have some knowledge about when an event occurred, use it to target your search to that time period for faster results.

It's your second day of work with the Customer Support team for the online Flower & Gift shop. You just got to your desk. Before you make yourself a cappuccino, you decide to run a quick search to see if there were any recent issues you should be aware of.

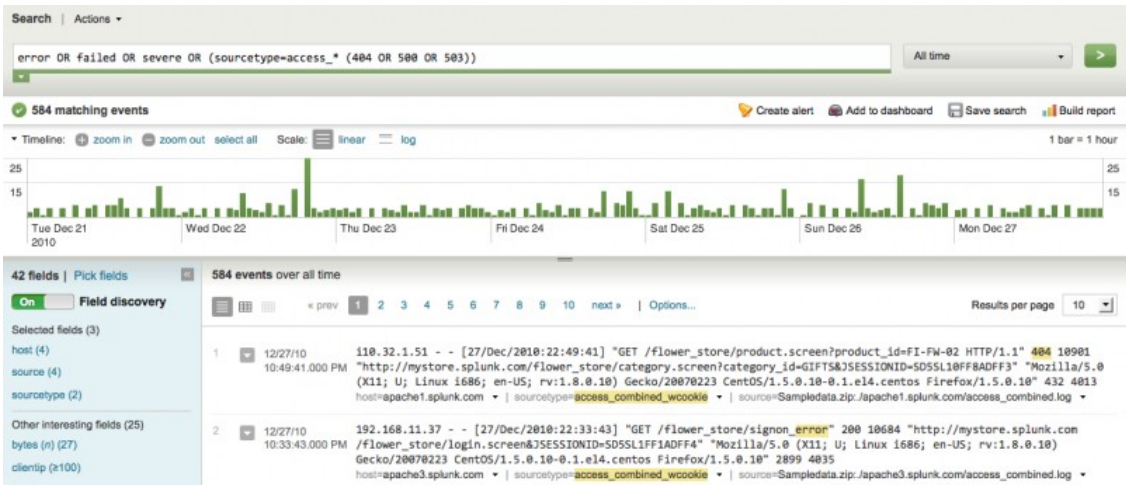
1. Return to the Search dashboard and type in the following search over all time:

```
error OR failed OR severe OR (sourcetype=access_* (404 OR 500 OR 503))
```

This search uses parentheses to group together expressions for more complicated searches. When evaluating Boolean expressions, Splunk performs the operations within the innermost parentheses first, followed by the next pair out. When all operations within parentheses are completed, Splunk evaluates OR clauses, then, AND or NOT clauses.

Also, this search uses the wildcarded shortcut, "access_*", to match the Web access logs. If you have different source types for your Apache server logs, such as `access_common` and `access_combined`, this will match them all.

This searches for general errors in your event data over the course of the last week. Instead of matching just one type of log, this searches across all the logs in your index. It matches any occurrence of the words "error", "failed", or "severe" in your event data. Additionally, if the log is a Web access log, it looks for HTTP error codes, "404", "500", or "503".



```
error OR failed OR severe OR (sourcetype=access_* (404 OR 500 OR 503))
```

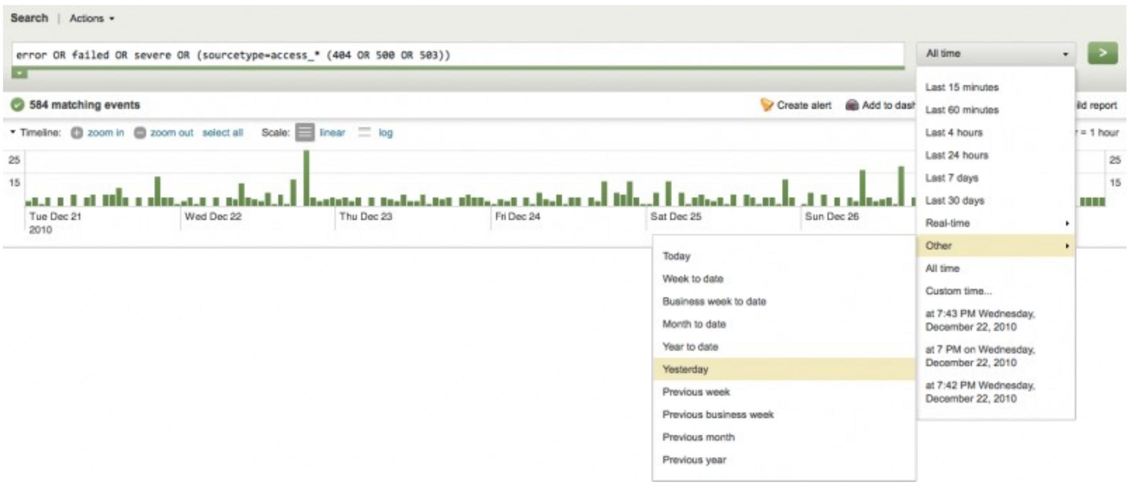
584 matching events

584 events over all time

```
1  12/27/10 10:49:41.000 PM  110.32.1.51 - - [27/Dec/2010:22:49:41] "GET /flower_store/product.screen?product_id=FI-FW-02 HTTP/1.1" 404 10901
"http://mystore.splunk.com/flower_store/category.screen?category_id=GIFTS&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0
(X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 432 4013
host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log

2  12/27/10 10:33:43.000 PM  192.168.11.37 - - [27/Dec/2010:22:33:43] "GET /flower_store/signon_error" 200 10684 "http://mystore.splunk.com
/flower_store/login.screen&JSESSIONID=SD5SL1FF1ADFF4" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2899 4035
host=apache3.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache3.splunk.com/access_combined.log
```

This search returns a significant amount of errors. You're not interested in knowing what happened over *All time*, even if it's just the course of a week. You just got into work, so you want to know about more recent activity, such as overnight or the last hour. But, because of the limitations of this dataset, let's look at yesterday's errors.

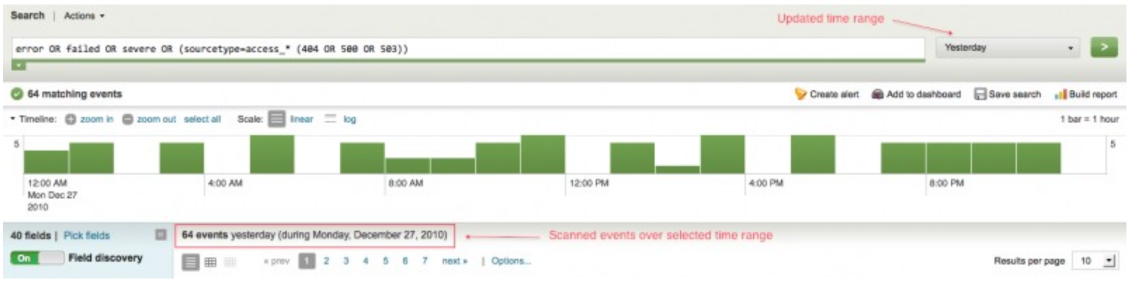
2. Drop down the time range picker and change the time range to *Other* > *Yesterday*.



```
error OR failed OR severe OR (sourcetype=access_* (404 OR 500 OR 503))
```

Out-of-the box, Splunk searches across all of your data; that is, the default time range for a search is across "All time". If you have a lot of data, searching on this time range when you're investigating an event that occurred 15 minutes ago, last night, or the previous week just means that Splunk will take a long time to retrieve the results that you want to see.

3. Selecting a time range from this list automatically runs the search for you. If it doesn't, just hit *Enter*.



```
error OR failed OR severe OR (sourcetype=access_* (404 OR 500 OR 503))
```

64 matching events

64 events yesterday (during Monday, December 27, 2010)

This search returns events for general errors across all your logs, not just Web access logs. (If your sample data file is more than a day old, you can still get these results by selecting *Custom time* and entering the last date for which you have data.) Scroll through the search results. There are more mySQL database errors and some 404 errors. You ask the intern to get you a cup of coffee while you contact the Web team about the 404 errors and the IT Operations team about the recurring server errors.

Splunk also provides options for users to define a custom time range to search or select to search a continuous stream of incoming events.

- **Real-time** enables searching forward in time against a continuous stream of live incoming event data. Because the sample data is a one-time upload, running a real-time search will not give us any results right now. We will explore this option later. Read more about real-time searches and how to run them in "Search and report in real-time"
- **Custom time...** pops up a new window and enables you to define your own time ranges based on specific dates, relative dates, real-time windows, or using the search language. Read more about how to define custom time ranges in "Change the time range of your search".

Up to now, you've run simple searches that matched the raw text in your events. You've only scratched the surface of what you can do in Splunk. When you're ready to proceed, go on to the next topic to learn about fields and how to search with fields.

Use fields to search

Use fields to search

This topic assumes you know how to run simple searches and use the time range picker and timeline. If you're not sure, review the previous topics, beginning with Start searching.

You can learn a lot about your data from just running ad hoc searches, using nothing more than keywords and the time range. But you can't take full advantage of Splunk's more advanced searching and reporting features without understanding what fields are and how to use them. This part of the tutorial will familiarize you with:

- default fields and other fields that Splunk automatically extracts

- using the fields menu and fields picker to find helpful fields
- searching with fields

Let's return to the happenings at the online Flower and Gift shop. It's your second day as a member of the Customer Support team. You spent the morning investigating some general issues and reporting the problems you found to other teams. You feel pretty good about what you've learned about the online shop and its customers, but you want to capture this and share it with your team. When you ask a coworker how you can do this, he recommends that you use fields.

What are fields?

Fields are searchable name/value pairings in event data. All fields have names and can be searched with those names. Some examples of fields are `clientip` for IP addresses accessing your Web server, `_time` for the timestamp of an event, and `host` for domain name of a server.

Fields distinguish one event from another because not all events will have the same fields and field values. Fields enable you to write more tailored searches to retrieve the specific events that you want. Fields also enable you to take advantage of the search language, create charts, and build reports.

Most fields in your data exist as name and value pairs where there is one single value to each field name. But, you'll also see fields that appear more than once in an event and have a different value for each appearance.

One of the more common examples of multivalue fields is email address fields. While the "From" field will contain only a single email address, the "To" and "Cc" fields may have one or more email addresses associated with them.

For more information, read About fields in the Knowledge Manager manual.

The fields menu and fields picker

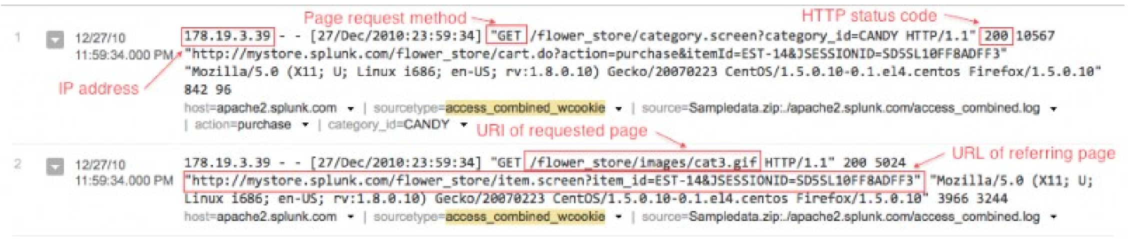
1. Go back to the Search dashboard and search for Web access activity. Select *Other > Yesterday* from the time range picker:

```
sourcetype="access_*"
```

2. Scroll through the search results.

If you're familiar with the access_combined format of Apache logs, you will recognize some of the information in each event, such as:

- IP addresses for the users accessing the website.
- URIs and URLs for the page request and referring page.
- HTTP status codes for each page request.
- Page request methods.



```
1  12/27/10          178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/category.screen?category_id=CANDY HTTP/1.1" 200 10567
   11:59:34.000 PM   "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-14&JSESSIONID=SD5SL10FF8ADFF3"
                     "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10"
                     842 96
                     host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log
                     | action=purchase | category_id=CANDY

2  12/27/10          178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/images/cat3.gif HTTP/1.1" 200 5024
   11:59:34.000 PM   "http://mystore.splunk.com/flower_store/item.screen?item_id=EST-14&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U;
                     Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3966 3244
                     host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log
```

As Splunk retrieves these events, the Fields menu updates with *Selected fields* and *Other interesting fields*. These are the fields that Splunk extracted from your data.

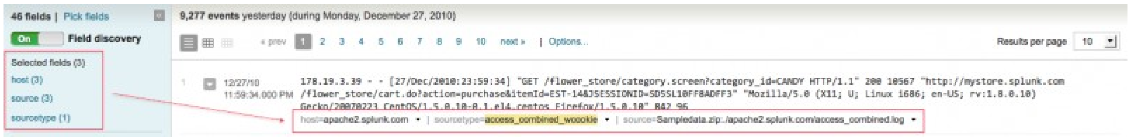
Default and automatically extracted fields

Splunk extracts fields from event data twice. It extracts default and other indexed fields during event processing when that data is indexed. And it extracts a different set of fields at search time, when you run a search. Read more about "Index time versus search time" in the Admin manual.

At index time, Splunk automatically finds and extracts *default fields* for each event it processes. These fields include `host`, `source`, and `sourcetype` (which you should already be familiar with). For a complete list of the default fields, see "Use default fields" in the User manual.

Splunk also extracts certain fields at search time--when you run a search. You'll see some examples of these searches later. For more information, read the "Overview of search-time field extractions" in the *Knowledge Manager manual*.

Notice that default fields host, source, and sourcetype are *Selected fields* and are included in your search results:



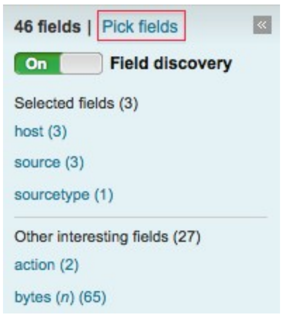
9,277 events yesterday (during Monday, December 27, 2010)

```
1  12/27/10          178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/category.screen?category_id=CANDY HTTP/1.1" 200 10567 "http://mystore.splunk.com
   11:59:34.000 PM   /flower_store/cart.do?action=purchase&itemId=EST-14&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
                     Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 842 96
                     host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log
```

3. Scroll through *Other interesting fields* to see what else Splunk extracted.

You should recognize the field names that apply to the Web access logs. For example, there's `clientip, method, and status`. These are *not* default fields; they have (most likely) been extracted at search time.

4. Click the *Pick fields* link in the fields menu.


46 fields | Pick fields

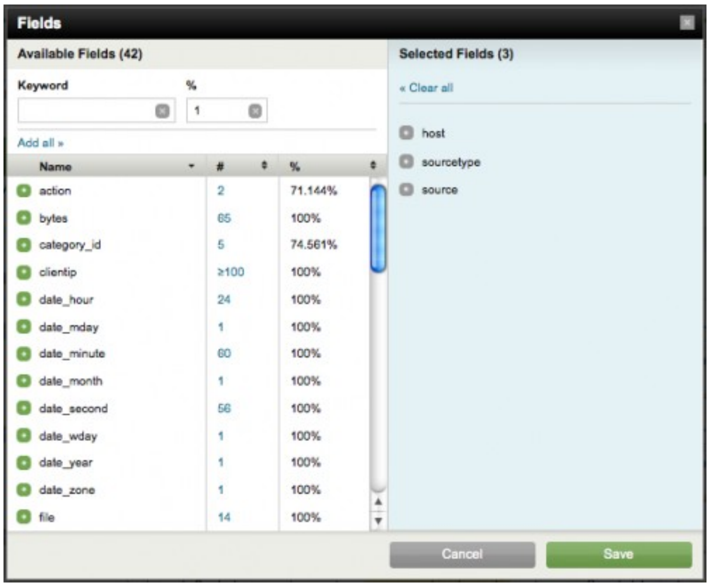
On Field discovery

Selected fields (3)
host (3)
source (3)
sourcetype (1)

Other interesting fields (27)
action (2)
bytes (n) (65)

The Fields picker opens and displays all the fields that Splunk extracted.

- *Available fields* are the fields that Splunk identified from the events in your current search (some of these fields were listed under **Other interesting fields**).
- *Selected fields* are the fields you picked (from the available fields) to show in your search results (by default, `host, source, and sourcetype` are selected).


Fields

Available Fields (42)

Keyword | % 1

Add all »

Name	#	%
action	2	71.144%
bytes	65	100%
category_id	5	74.561%
clientip	≥100	100%
date_hour	24	100%
date_mday	1	100%
date_minute	60	100%
date_month	1	100%
date_second	56	100%
date_wday	1	100%
date_year	1	100%
date_zone	1	100%
file	14	100%

Selected Fields (3)

« Clear all

host
sourcetype
source

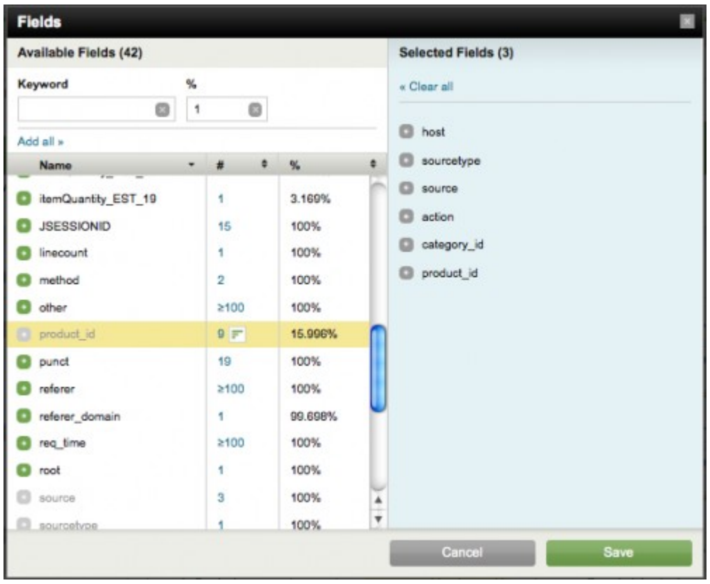
5. Scroll through the list of *Available fields*.

You're already familiar with the fields that Splunk extracted from the Web access logs based on your search. You should also see other default fields that Splunk defined--some of these fields are based on each event's `timestamp` (everything beginning with `date_*`), punctuation (`punct`), and location (`index`).

But, you should also notice other extracted fields that are related to the online store. For example, there are `action`, `category_id`, and `product_id`. From conversations with your coworker, you may know that these fields are:

field name	description
action	what a user does at the online shop.
category_id	the type of product a user is viewing or buying.
product_id	the catalog number of the product the user is viewing or buying.

6. From the *Available fields* list, select action, category_id, and product_id.



Fields

Available Fields (42)

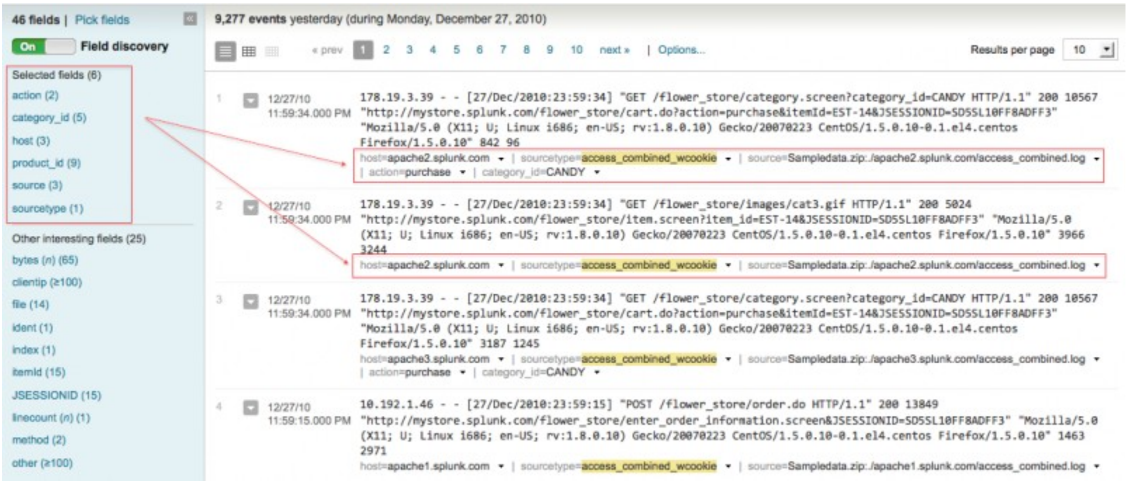
Name	#	%
itemQuantity_EST_19	1	3.169%
JSESSIONID	15	100%
linecount	1	100%
method	2	100%
other	≥100	100%
product_id	9	15.996%
punct	19	100%
referer	≥100	100%
referer_domain	1	99.698%
req_time	≥100	100%
root	1	100%
source	3	100%
sourcetype	1	100%

Selected Fields (3)

- host
- sourcetype
- source
- action
- category_id
- product_id

7. Click *Save*.

When you return to the Search view, the fields you selected will be included in your search results *if they exist in that particular event*. Different events will have different fields.



46 fields | Pick fields

On Field discovery

Selected fields (6)
action (2)
category_id (5)
host (3)
product_id (9)
source (3)
sourcetype (1)

Other interesting fields (25)
bytes (n) (65)
clientip (≥100)
file (14)
ident (1)
index (1)
itemId (15)
JSESSIONID (15)
linecount (n) (1)
method (2)
other (≥100)

9,277 events yesterday (during Monday, December 27, 2010)

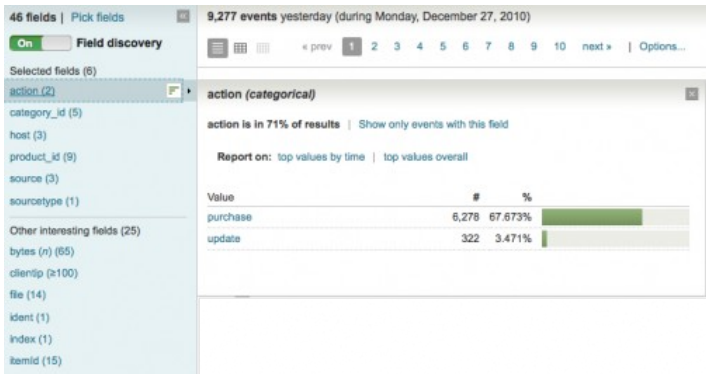
```
1  12/27/10 11:59:34.000 PM
178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/category.screen?category_id=CANDY HTTP/1.1" 200 10567
"http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-14&JSESSIONID=SD5SL10FF8ADFF3"
"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos
Firefox/1.5.0.10" 842 96
host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log
| action=purchase | category_id=CANDY

2  12/27/10 11:59:34.000 PM
178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/images/cat3.gif HTTP/1.1" 200 5024
"http://mystore.splunk.com/flower_store/item.screen?item_id=EST-14&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0
(X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3966
3244
host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log

3  12/27/10 11:59:34.000 PM
178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/category.screen?category_id=CANDY HTTP/1.1" 200 10567
"http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-14&JSESSIONID=SD5SL10FF8ADFF3"
"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos
Firefox/1.5.0.10" 3187 1245
host=apache3.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache3.splunk.com/access_combined.log
| action=purchase | category_id=CANDY

4  12/27/10 11:59:15.000 PM
10.192.1.46 - - [27/Dec/2010:23:59:15] "POST /flower_store/order.do HTTP/1.1" 200 13849
"http://mystore.splunk.com/flower_store/enter_order_information.screen&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0
(X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 1463
2971
host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log
```

The fields menu doesn't just show you what fields Splunk has captured from your data. It also displays how many values exist for each of these fields. For the fields you just selected, there are 2 for action, 5 for category_id, and 9 for product_id. This doesn't mean that these are all the values that exist for each of the fields--these are just the values that Splunk knows about from the results of your search.

What are some of these values?

8. Under **Selected fields**, click **action** for the `action` field.

This opens the interactive menu for the action field.



9,277 events yesterday (during Monday, December 27, 2010)

action *(categorical)*

action is in 71% of results

Value	#	%
purchase	6,278	67.673%
update	322	3.471%

This window tells you that, in this set of search results, Splunk found two values for `action` and they are `purchase` and `update`. Also, it tells you that the `action` field appears in 71% of your search results. This means that three-quarters of the Web access events are related to the purchase of an item or an update (of the item quantity in the cart, perhaps).

9. Close this window and look at the other two fields you selected, `category_id` (what types of products the shop sells) and `product_id` (specific catalog names for products).

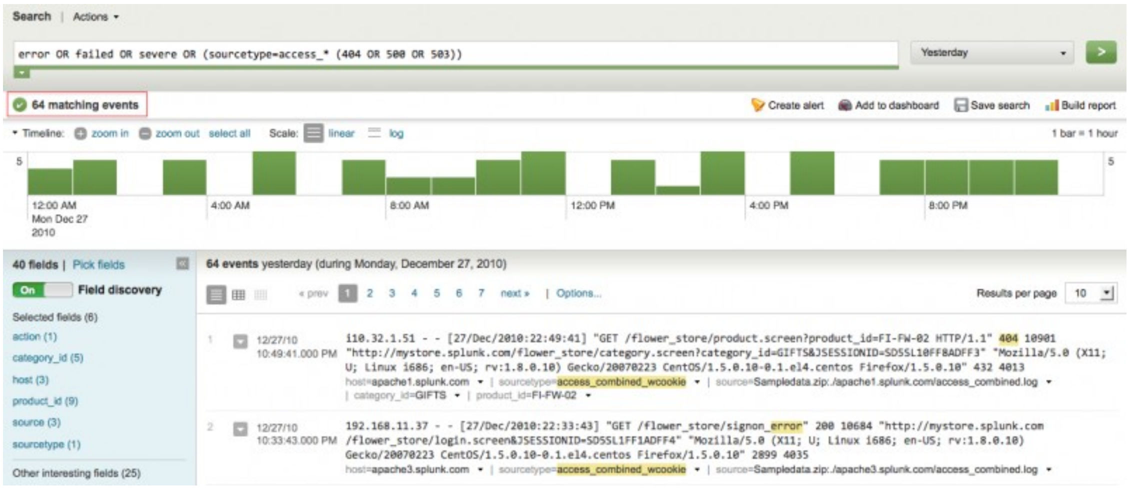
Now you know a little bit more about the information in your data relating to the online Flower and Gift shop. Let's use these fields to see what people are buying. For example, the online shop sells a selection of flowers, gifts, plants, candy, and balloons.

Use fields to run more targeted searches

These next two examples illustrate the difference between searching with keywords and using fields.

Example 1: Return to the search you ran to check for errors in your data. Select *Other > Yesterday* from the time range picker:

```
error OR failed OR severe OR (sourcetype=access_* (404 OR 500 OR 503))
```



```
error OR failed OR severe OR (sourcetype=access_* (404 OR 500 OR 503))
```

64 matching events

40 fields

Selected fields (6)
action (1)
category_id (5)
host (3)
product_id (9)
source (3)
sourcetype (1)
Other interesting fields (25)

64 events yesterday (during Monday, December 27, 2010)

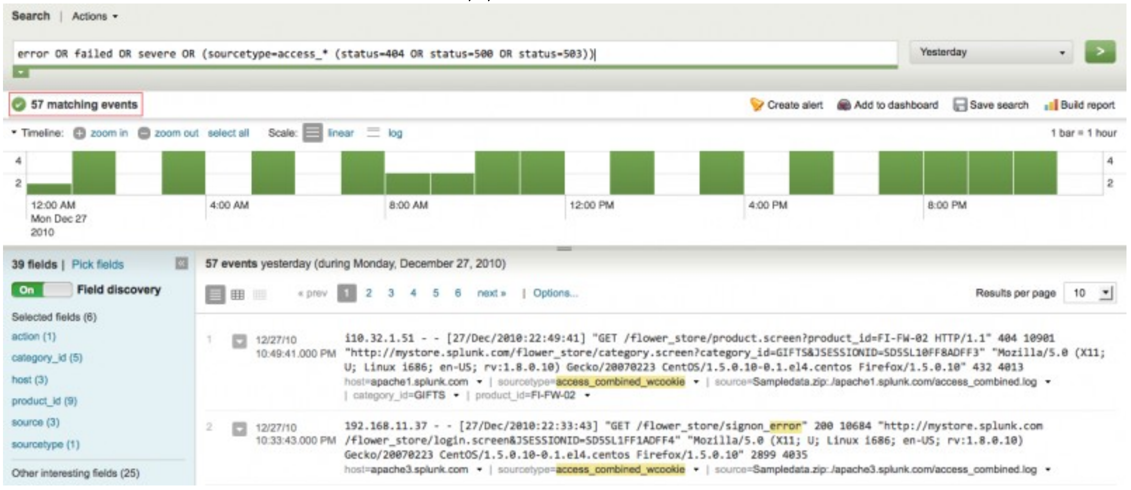
```
1  12/27/10           i10.32.1.51 - - [27/Dec/2010:22:49:41] "GET /flower_store/product.screen?product_id=FI-FW-02 HTTP/1.1" 404 10901
   10:49:41.000 PM    "http://mystore.splunk.com/flower_store/category.screen?category_id=GIFTS&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11;
                      U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 432 4013
                      host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log
                      | category_id=GIFTS | product_id=FI-FW-02

2  12/27/10           192.168.11.37 - - [27/Dec/2010:22:33:43] "GET /flower_store/signon_error" 200 10684 "http://mystore.splunk.com
   10:33:43.000 PM    /flower_store/login.screen&JSESSIONID=SD5SL1FF1ADFF4" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
                      Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2899 4035
                      host=apache3.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache3.splunk.com/access_combined.log
```

Run this search again, but this time, use fields in your search.

To search for a particular field, just type the field name and value into the search bar: `fieldname=fieldvalue`

The HTTP error codes are values of the `status` field. Now your search looks like this:

```
error OR failed OR severe OR (sourcetype=access_* (status=404 OR
status=500 OR status=503))
```



```
error OR failed OR severe OR (sourcetype=access_* (status=404 OR status=500 OR status=503))
```

57 matching events

39 fields

Selected fields (6)
action (1)
category_id (5)
host (3)
product_id (9)
source (3)
sourcetype (1)
Other interesting fields (25)

57 events yesterday (during Monday, December 27, 2010)

```
1  12/27/10           i10.32.1.51 - - [27/Dec/2010:22:49:41] "GET /flower_store/product.screen?product_id=FI-FW-02 HTTP/1.1" 404 10901
   10:49:41.000 PM    "http://mystore.splunk.com/flower_store/category.screen?category_id=GIFTS&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11;
                      U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 432 4013
                      host=apache1.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache1.splunk.com/access_combined.log
                      | category_id=GIFTS | product_id=FI-FW-02

2  12/27/10           192.168.11.37 - - [27/Dec/2010:22:33:43] "GET /flower_store/signon_error" 200 10684 "http://mystore.splunk.com
   10:33:43.000 PM    /flower_store/login.screen&JSESSIONID=SD5SL1FF1ADFF4" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
                      Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2899 4035
                      host=apache3.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache3.splunk.com/access_combined.log
```

Notice the difference in the count of events between the two searches--because it's a more targeted search, the second search returns fewer events.

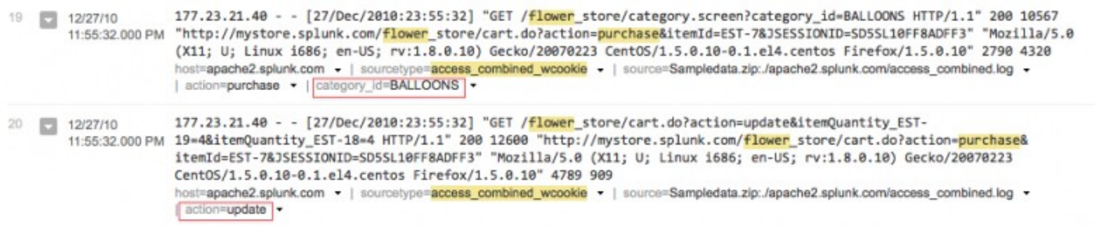
When you run simple searches based on arbitrary keywords, Splunk matches the raw text of your data. When you add fields to your search, Splunk looks for events that have those specific field/value pairs.

Also, you were actually using fields all along! Each time you searched for `sourcetype=access_*`, you told Splunk to only retrieve events from your Web access logs and nothing else.

Example 2: Before you learned about the fields in your data, you might have run this search to see how many times flowers were purchased from the online shop:

```
sourcetype=access_* purchase flower*
```

As you typed in "flower", search assistant shows you both "flower" and "flowers' in the typeahead. Since you don't know which is the one you want, you use the wildcard to match both.

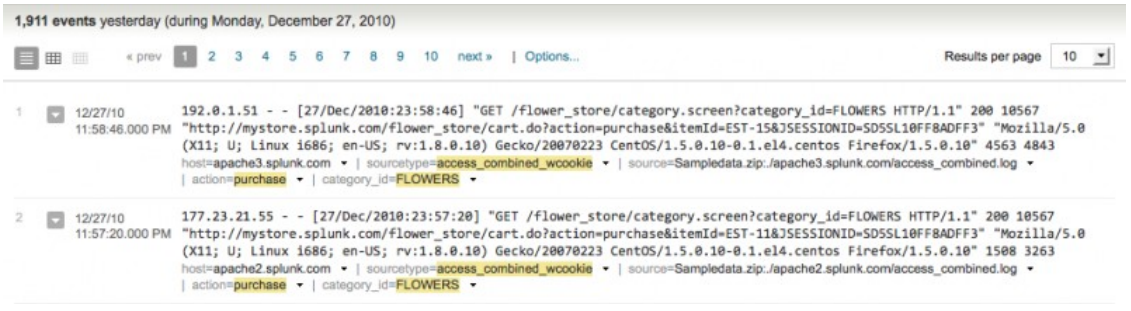


```
19  12/27/10            177.23.21.40 - - [27/Dec/2010:23:55:32] "GET /flower_store/category.screen?category_id=BALLOONS HTTP/1.1" 200 10567
    11:55:32.000 PM     "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-7&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0
                        (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2790 4320
                        host=apache2.splunk.com ▾ | sourcetype=access_combined_wcookie ▾ | source=Sampledata.zip:./apache2.splunk.com/access_combined.log ▾
                        | action=purchase ▾ | category_id=BALLOONS ▾

20  12/27/10            177.23.21.40 - - [27/Dec/2010:23:55:32] "GET /flower_store/cart.do?action=update&itemQuantity_EST-
    11:55:32.000 PM     19=4&itemQuantity_EST-18=4 HTTP/1.1" 200 12600 "http://mystore.splunk.com/flower_store/cart.do?action=purchase&
                        itemId=EST-7&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223
                        CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 4789 909
                        host=apache2.splunk.com ▾ | sourcetype=access_combined_wcookie ▾ | source=Sampledata.zip:./apache2.splunk.com/access_combined.log ▾
                        | action=update ▾
```

If you scroll through the (many) search results, you'll see that some of the events have `action=update` and `category_id` that have a value other than `flowers`. These are not events that you wanted!

Run this search instead. Select *Other* > *Yesterday* from the time range picker:

```
sourcetype=access_* action=purchase category_id=flower*
```



1,911 events yesterday (during Monday, December 27, 2010)

« prev 1 2 3 4 5 6 7 8 9 10 next » | Options... Results per page 10

```
1   12/27/10            192.0.1.51 - - [27/Dec/2010:23:58:46] "GET /flower_store/category.screen?category_id=FLOWERS HTTP/1.1" 200 10567
    11:58:46.000 PM     "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-15&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0
                        (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 4563 4843
                        host=apache3.splunk.com ▾ | sourcetype=access_combined_wcookie ▾ | source=Sampledata.zip:./apache3.splunk.com/access_combined.log ▾
                        | action=purchase ▾ | category_id=FLOWERS ▾

2   12/27/10            177.23.21.55 - - [27/Dec/2010:23:57:20] "GET /flower_store/category.screen?category_id=FLOWERS HTTP/1.1" 200 10567
    11:57:20.000 PM     "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-11&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0
                        (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 1508 3263
                        host=apache2.splunk.com ▾ | sourcetype=access_combined_wcookie ▾ | source=Sampledata.zip:./apache2.splunk.com/access_combined.log ▾
                        | action=purchase ▾ | category_id=FLOWERS ▾
```

For the second search, even though you still used the wildcarded word "flower*", there is only one value of `category_id` that it matches (`FLOWERS`).

Notice the difference in the number of events that Splunk retrieved for each search; the second search returns significantly fewer events. Searches with fields are more targeted and retrieves more exact matches against your data.

As you run more searches, you want to be able to save them and reuse them or share them with your teammates. When you're ready, proceed to the next topic to learn how to save your search and share it it with others.

Save a search

Save a search

This topic assumes you're comfortable running searches with fields. If you're not, go back to the previous topic and review how to *Use fields to search*.

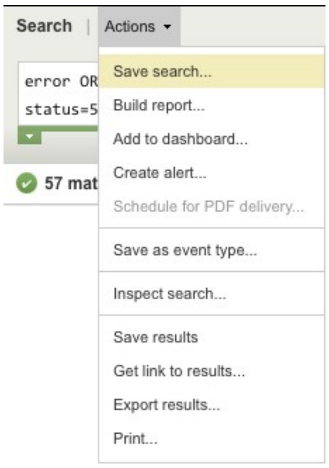
This topic walks you through the basics of saving a search and how you can use that search again later.

Back at the Flower & Gift shop, you just ran a search to see if there were any errors yesterday. This is a search you will run every morning. Rather than type it in manually every day, you decide to save this search.

Example 1. Run the search for all errors seen yesterday:

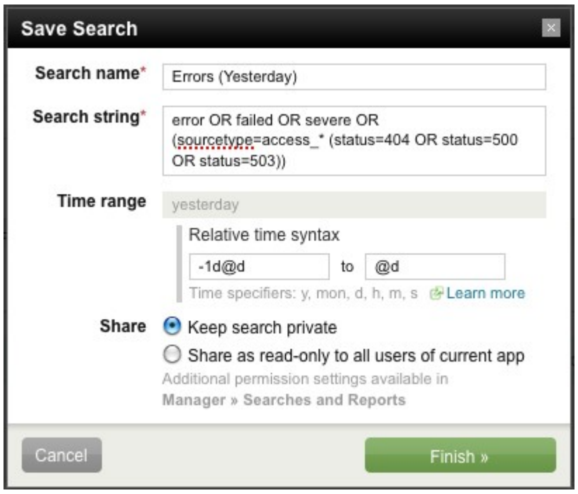
```
error OR failed OR severe OR (sourcetype=access_* (status=404 OR
status=500 OR status=503))
```

1. Click **Actions** above the search bar.



2. Select *Save search...* from the drop down list.

The *Save search* dialog box opens.


Save Search

Search name* Errors (Yesterday)

Search string* error OR failed OR severe OR (sourcetype=access_* (status=404 OR status=500 OR status=503))

Time range yesterday

Relative time syntax -1d@d to @d

Time specifiers: y, mon, d, h, m, s Learn more

Share [✓ Keep search private] Share as read-only to all users of current app

Additional permission settings available in Manager » Searches and Reports

Cancel Finish »

At a minimum, a saved search includes the search string and the time range associated with the search, as well as the name of the search.

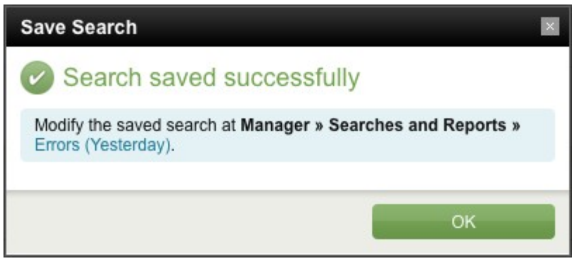
3. Name the search, *Errors (Yesterday)*

4. Under **Search**, don't modify the search string.

5. Leave the **Time range** as it is.

6. Leave **Share** set as it is.

7. Click **Finish**. Splunk confirms that your search was saved:

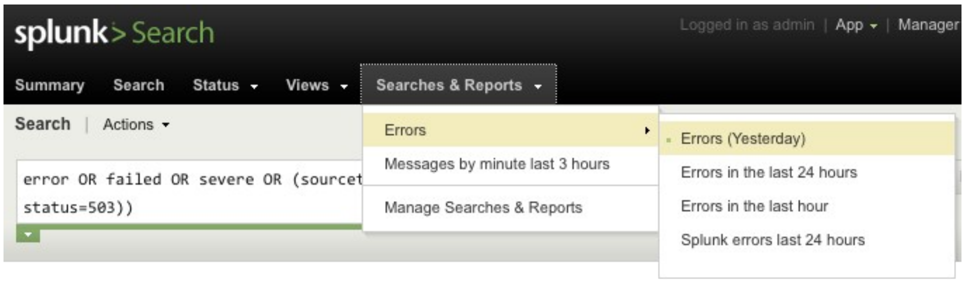

Save Search

Search saved successfully

Modify the saved search at **Manager » Searches and Reports »** Errors (Yesterday).

OK

8. Find your saved search in the **Searches & Reports** list:


splunk> Search
Logged in as admin | App | Manager
Summary Search Status Views Searches & Reports
Search | Actions
Errors
Messages by minute last 3 hours
Manage Searches & Reports
Errors (Yesterday)
Errors in the last 24 hours
Errors in the last hour
Splunk errors last 24 hours

```
error OR failed OR severe OR (sourcet
status=503))
```

Because the saved search's name contained the word "Error," Splunk lists it in the saved search submenu for Errors.

The green dot next to your saved search means that it's local to your Splunk account; right now you are the only one that is authorized to access this saved search. Since this is a search that others on your team may want to run, you can set it as a global saved search that they can access. To do this, read more about saving searches and sharing search results.

Manage searches and reports

If you want to modify a search that you saved, use the **Searches & Reports** menu to select *Manage Searches & Reports*. This takes you the Splunk Manager page for all the searches and reports you're allowed to access (if you're allowed to access them). From here you can select your search from the list. This take you to the searches edit window where you can then change or update the search string, description, time range, and schedule options.

Schedule saved searches and alerts

If you have an Enterprise license, Splunk also lets you configure the searches you saved to run on a schedule and to set alerts based off the scheduled searches. When you download Splunk for the first time, you're given an Enterprise trial license that expires after 60 days. If you're using the Free license, you do not have the capability to schedule a saved search. Read more about scheduling saved searches and setting alerts in the *Monitoring recurring situations* chapter of the User manual.

Save and share search results

Saving the results of a search is different from saving the search itself. You do this when you want to be able to review the outcome of a particular run of a search at a later time. Read more about this in saving searches and sharing search results

Now, you can save your searches after you run them. When you're ready, proceed to the next topic to learn more ways to search.

Use Splunk's search language

Use Splunk's search language

This topic assumes that you are familiar with running simple searches using keywords and field/value pairs. If you're not sure, go back and read "Use fields to search".

Back at the online Flower & Gift shop Customer Support office, the searches you've run to this point have only retrieved matching events from your Splunk index.

In a previous topic, when you ran this search for purchases of flowers:

```
sourcetype=access_* action=purchase category_id=flowers
```

The search results told you approximately how many flowers were bought. But, this doesn't help you answer questions, such as:

- What items were purchased most at the online shop?
- How many customers bought flowers? How many flowers did each customer buy?

To answer these questions, you need to use Splunk's search language, which includes an extensive library of commands, arguments, and functions that enables you to filter, modify, reorder, and group your search results. For this tutorial you'll only use a few of them.

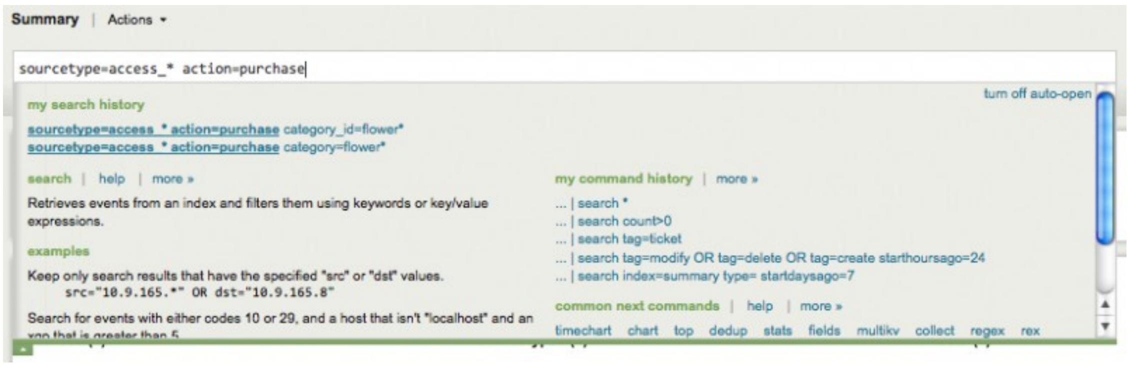
More search assistant

Example 1. What items were purchased most at the online shop?

1. Return to the search dashboard and restrict your search to purchases over *Yesterday*:

```
sourcetype=access_* action=purchase
```

As you type in the search bar, search assistant opens with syntax and usage information for the `search` command (on the right side). If search assistant doesn't open, click the green arrow under the left side of the search bar.



```
sourcetype=access_* action=purchase
```

my search history

sourcetype=access_* action=purchase category_id=flower*
sourcetype=access_* action=purchase category=flower*

search | help | more »

Retrieves events from an index and filters them using keywords or key/value expressions.

examples

Keep only search results that have the specified "src" or "dst" values.

```
    src="10.9.165.*" OR dst="10.9.165.8"
```

Search for events with either codes 10 or 29, and a host that isn't "localhost" and an xqp that is greater than 5

my command history | more »

... | search *
... | search count>0
... | search tag=ticket
... | search tag=modify OR tag=delete OR tag=create starthoursago=24
... | search index=summary type= startdaysago=7

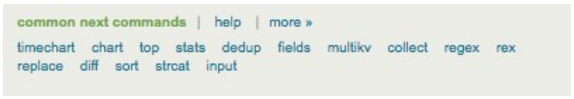
common next commands | help | more »

timechart chart top dedup stats fields multikv collect regex rex

You've seen before that search assistant displays typeahead for keywords that you type into the search bar. It also attempts to help you construct your search string by showing your **search history**, your **command history**, and suggesting other search commands you may want to use next based on your history (**common next commands**).

Search assistant shows you the `search` command because each time you ran a search to now, you were using the `search` command--it's implied.

2. In search assistant, scroll down to **common next commands**. If you don't see this section, type a pipe character into the search bar.



common next commands | help | more »

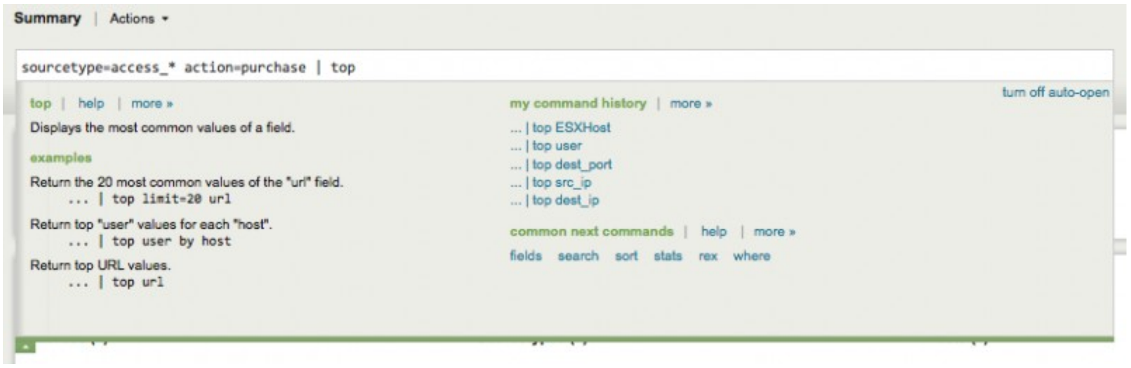
timechart chart top stats dedup fields multikv collect regex rex
replace diff sort strcat input

You want Splunk to give you the most popular items bought at the online store--the `top` command looks promising.

Construct a search pipeline

3. Under **common next commands**, click **top**.

Splunk appends the top command to your search string with a pipe "|" character.


Summary | Actions ▾
turn off auto-open
top | help | more »
my command history | more »
common next commands | help | more »
fields search sort stats rex where

```
sourcetype=access_* action=purchase | top
```

Displays the most common values of a field.

examples

Return the 20 most common values of the "url" field.
```
... | top limit=20 url
```
Return top "user" values for each "host".
```
... | top user by host
```
Return top URL values.
```
... | top url
```

```
... | top ESXHost
... | top user
... | top dest_port
... | top src_ip
... | top dest_ip
```

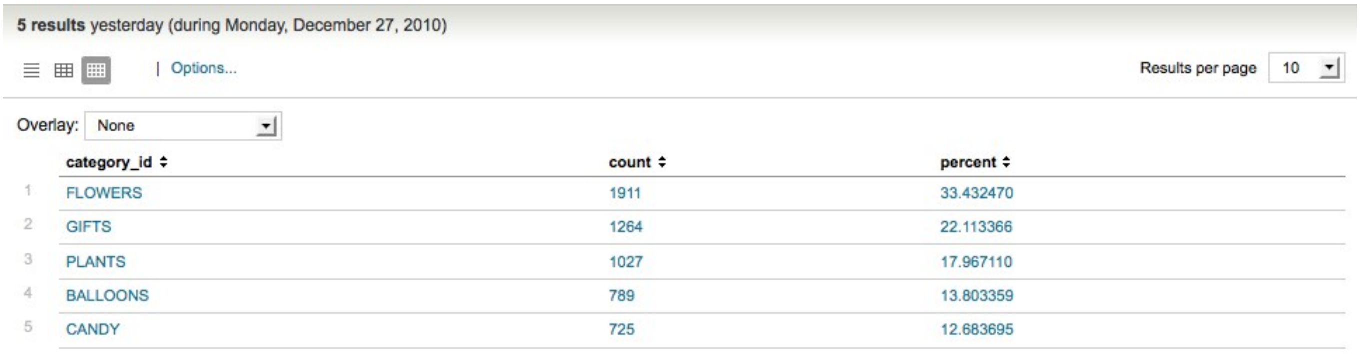
The pipe indicates to Splunk that you want to take the results of the search to the left of the pipe and use that as the input to the command after the pipe. You can pass the results of one command into another command in a series, or pipeline, of search commands.

According to search assistant's description and usage examples, the **top** command "displays the most common values of a field"--exactly what you wanted.

You wanted to know what types of items were being bought at the online shop, not just flowers.

4. Complete your search with the `category_id` field:

```
sourcetype=access_* action=purchase | top category_id
```
This gives you a table of the top or most common values of category_id. By default, the `top` command returns ten values, but you only have five different types of items. So, you should see all five, sorted in descending order by the count of each type:



5 results yesterday (during Monday, December 27, 2010)

Overlay: None

	category_id	count	percent
1	FLOWERS	1911	33.432470
2	GIFTS	1264	22.113366
3	PLANTS	1027	17.967110
4	BALLOONS	789	13.803359
5	CANDY	725	12.683695

The top command also returns two new fields: `count` is the number of times each value of the field occurs, and `percent` is how large that count is compared to the total count. Read more about the top command in the Search reference manual.

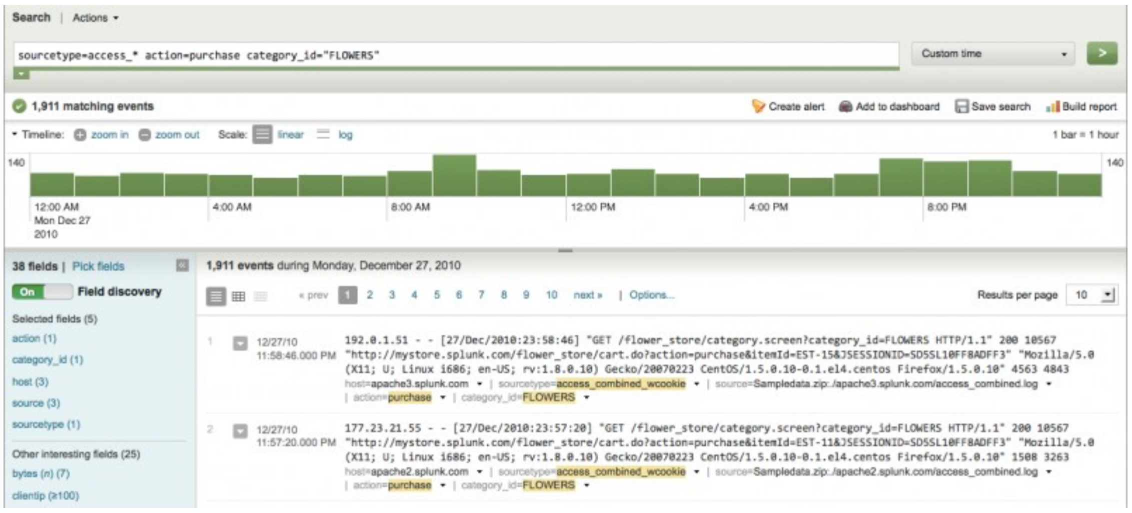
Drill down into search results

The last search returned a table that showed you what items the online shop sells and how many of those items were purchased. But, you want to know more about an individual item, for example, flowers.

Example 2: How many flowers were bought?

1. Click the row in the result table for Flowers.

This kicks off a new search. Splunk updates your search, to include the filter for the field/value pair `category=flowers`, which was the row item you clicked in the result table from the search in Example 2.



```
sourcetype=access_* action=purchase category_id="FLOWERS"
```

1,911 matching events

1,911 events during Monday, December 27, 2010

```
1  12/27/10 11:58:46.000 PM
192.0.1.51 - - [27/Dec/2010:23:58:46] "GET /flower_store/category.screen?category_id=FLOWERS HTTP/1.1" 200 10567 "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-15&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 4563 4843
host=apache3.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache3.splunk.com/access_combined.log | action=purchase | category_id=FLOWERS

2  12/27/10 11:57:20.000 PM
177.23.21.55 - - [27/Dec/2010:23:57:20] "GET /flower_store/category.screen?category_id=FLOWERS HTTP/1.1" 200 10567 "http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-11&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 1508 3263
host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log | action=purchase | category_id=FLOWERS
```

Splunk's **drilldown actions** enable you to delve deeper into the details of the information presented to you in the tables and charts that result from your search. Read more about drilldown actions in the User manual.

The number of events returned tells you how many times flowers were purchased, but you want to know how many different customers bought the flowers.

Example 3: How many different customers purchased the flowers?

1. You're looking specifically for the purchase of flowers, so continue with the search from the previous example:

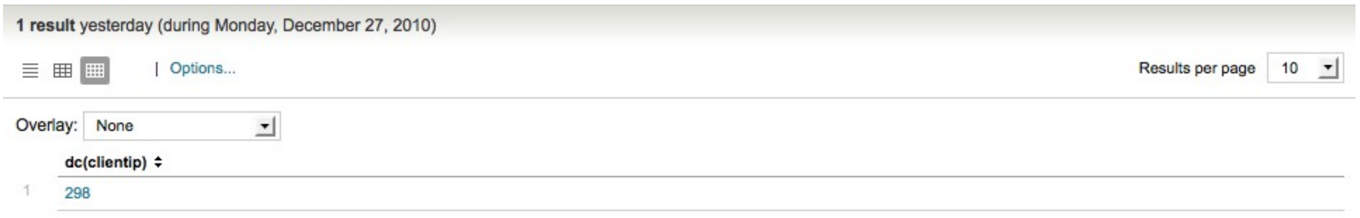
```
sourcetype=access_* action=purchase category_id=flowers
```

The customers who access the Flower & Gift shop are distinguished by their IP addresses, which are values of the `clientip` field.

2. Use the stats command and the distinct_count() or dc() function:

```
sourcetype=access_* action=purchase category_id=flowers | stats
dc(clientip)
```

You piped the search results into the `stats` command and used the `distinct_count()` function to count the number of unique `clientip` values that it finds in those events. This returns a single value:



1 result yesterday (during Monday, December 27, 2010)

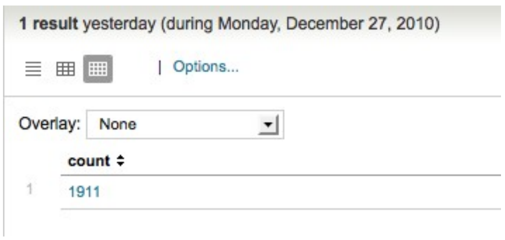
	dc(clientip)
1	298

This tells you that there were approximately 300 different people who bought flowers from the online shop.

Example: 4 In the last example, you calculated how many different customers bought flowers. How do you find the number of flowers that each customer bought?

1. Use the stats command:

```
sourcetype=access_* action=purchase category_id=flowers | stats count
```



1 result yesterday (during Monday, December 27, 2010)

	count
1	1911

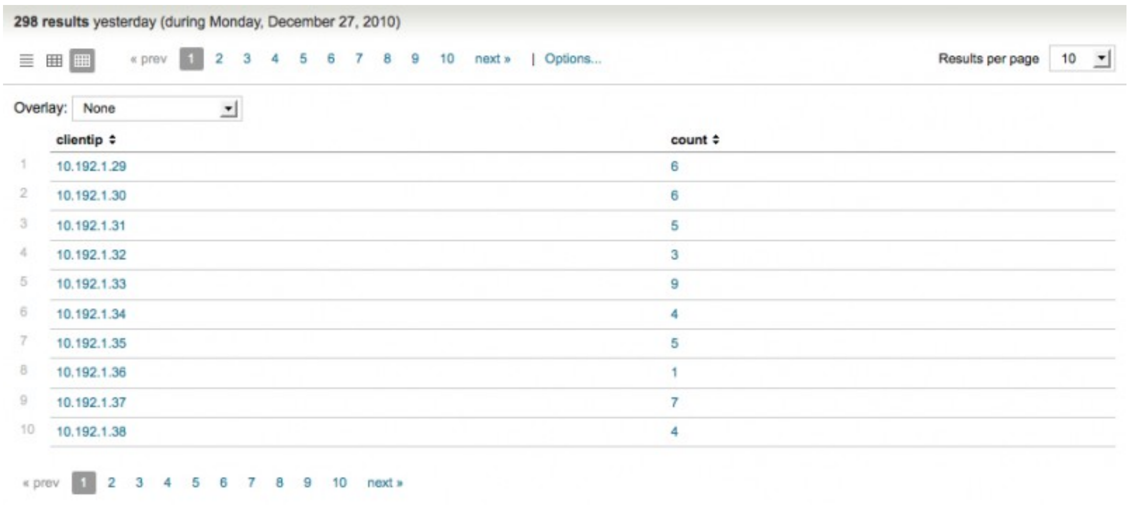
The `count()` function returns a single value, the count of your events. (This should match your result from Example 2.)

Now, break this count down to see how many flowers each customer bought.

2. Add a *by* clause to the `stats` command:

```
sourcetype=access_* action=purchase category_id=flowers | stats count BY
clientip
```

This search gives you a table of the different customers (clientip) and the number of flowers purchased (count).



298 results yesterday (during Monday, December 27, 2010)

	clientip	count
1	10.192.1.29	6
2	10.192.1.30	6
3	10.192.1.31	5
4	10.192.1.32	3
5	10.192.1.33	9
6	10.192.1.34	4
7	10.192.1.35	5
8	10.192.1.36	1
9	10.192.1.37	7
10	10.192.1.38	4

Reformat the search results

You might know what the header for this table represents, but anyone else wouldn't know at a glance. You want to show off your results to your boss and other members of your team. Let's reformat it a little:

3. First, let's rename the `count` field:

```
sourcetype=access_* action=purchase category_id=flowers | stats count AS
"# Flowers Purchased" by clientip
```

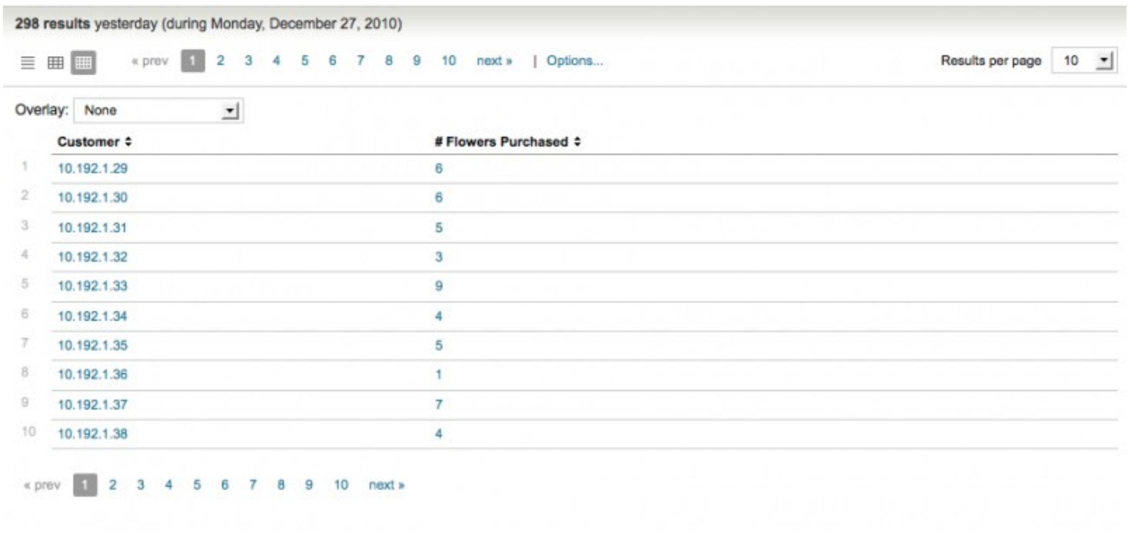
The syntax for the `stats` command enables you to rename the field inline using an "AS" clause. If your new field name is a phrase, use double quotes.

The syntax for the `stats` command doesn't allow field renaming in the "by" clause.

4. Use the `rename` command to change the `clientip` name:

```
sourcetype=access_* action=purchase category_id=flowers | stats count AS
"# Flowers Purchased" by clientip | rename clientip AS Customer
```

This formats the table to rename the headers, clientip and count, with *Customer* and *# Flowers purchased*:


298 results yesterday (during Monday, December 27, 2010)

« prev 1 2 3 4 5 6 7 8 9 10 next » | Options... Results per page 10

Overlay: None

	Customer	# Flowers Purchased
1	10.192.1.29	6
2	10.192.1.30	6
3	10.192.1.31	5
4	10.192.1.32	3
5	10.192.1.33	9
6	10.192.1.34	4
7	10.192.1.35	5
8	10.192.1.36	1
9	10.192.1.37	7
10	10.192.1.38	4

« prev 1 2 3 4 5 6 7 8 9 10 next »

For more information about the stats command and its usage, arguments, and functions, see the stats command in the Search reference manual and the list of stats functions. For more information about the rename command, see the rename command in the Search reference manual.

In this last search, you found how many flowers each customer to the online shop bought. But what if you were looking for the one customer who buys the most items on any given day? When you're ready, continue on to the next topic to learn another way to search, this time using subsearches.

Use a subsearch

Use a subsearch

The last topic introduced search commands, the search pipeline, and drilldown actions. If you're not familiar with them, review more ways to search.

This topic walks you through another search example and shows you two approaches to getting the results that you want.

Back at the Flower & Gift shop, your boss asks you to put together a report that shows the customer who bought the most items yesterday and what he or she bought.

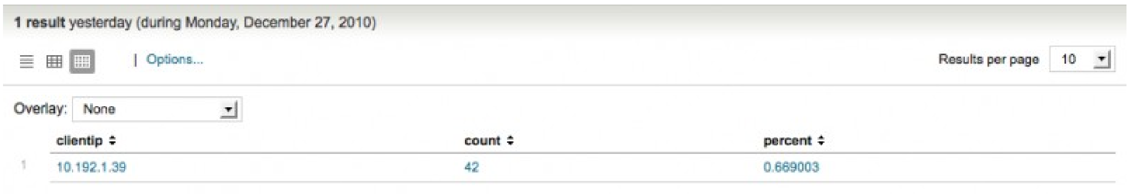
Part 1: Break the search down.

Let's see which customer accessed the online shop the most yesterday.

1. Use the `top` command and limit the search to *Yesterday*:

```
sourcetype=access_* action=purchase | top limit=1 clientip
```
You limit the `top` command to return only one result for the `clientip`. If you wanted to see more

than one "top purchasing customer", you would change this limit value.

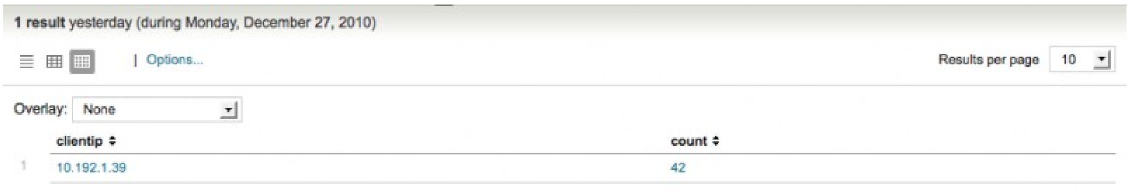

1 result yesterday (during Monday, December 27, 2010)

	clientip	count	percent
1	10.192.1.39	42	0.669003

Now, use the `clientip` value to complete your search.

2. Use the `stats` command to count the VIP customer's purchases:

```
sourcetype=access_* action=purchase clientip=10.192.1.39 | stats count by
clientip
```

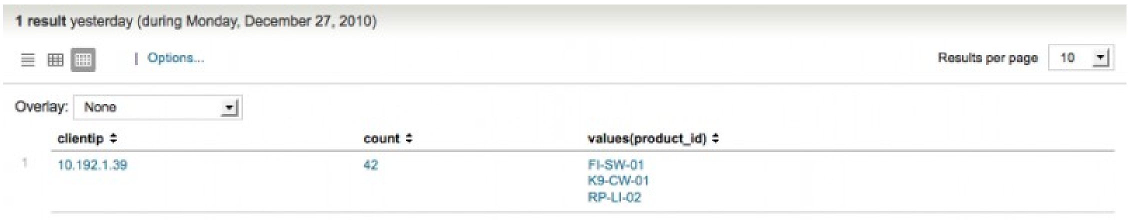

1 result yesterday (during Monday, December 27, 2010)

	clientip	count
1	10.192.1.39	42

This only returns the count of purchases for the clientip. You also want to know what he bought.

3. One way to do this is to use the `values()` function:

```
sourcetype=access_* action=purchase clientip=10.192.1.39 | stats count,
values(product_id) by clientip
```
This adds a column to the table that lists what he bought by product ID.


1 result yesterday (during Monday, December 27, 2010)

	clientip	count	values(product_id)
1	10.192.1.39	42	FI-SW-01 K9-CW-01 RP-LI-02

The drawback to this approach is that you have to run two searches each time you want to build this table. The top purchaser is not likely to be the same person at any given time range.

Part 2: Let's use a subsearch instead.

A subsearch is a search with a search pipeline as an argument. Subsearches are contained in square brackets and evaluated first. The result of the subsearch is then used as an argument to the primary search. Read more about "How subsearches work" in the User manual.

1. Use a subsearch to run the searches from Part 1 inline. Type or copy/paste in:

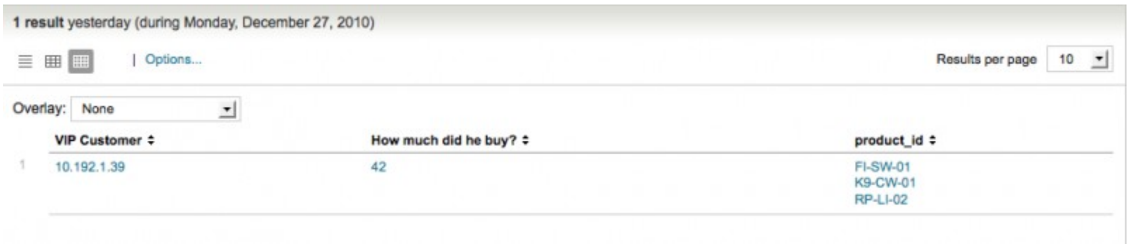
```
sourcetype=access_* action=purchase [search sourcetype=access_*
action=purchase | top limit=1 clientip | table clientip] | stats count,
values(product_id) by clientip
```

Because the `top` command returns `count` and `percent` fields as well, you use the `table` command to keep only the `clientip` value.

These results should match the previous result, if you run it on the same time range. But, if you change the time range, you might see different results because the top purchasing customer will be different!

2. Reformat the results so that it's easier to read:

```
sourcetype=access_* action=purchase [search sourcetype=access_*
action=purchase | top limit=1 clientip | table clientip] | stats count,
values(product_id) as product_id by clientip | rename count AS "How much
did he buy?", product_id AS "What did he buy?", clientip AS "VIP Customer"
```


1 result yesterday (during Monday, December 27, 2010)

| Options...   Results per page 10

Overlay: None

	VIP Customer	How much did he buy?	product_id
1	10.192.1.39	42	FI-SW-01 K9-CW-01 RP-LI-02

For more information about the usage and syntax for the sort command, see the sort command in the Search Reference manual.

While this report is perfectly acceptable, you want to make it better. For example, you don't expect your boss to know the shop items by their product ID numbers. You want to display the VIP customer's purchases by the product names, rather than the cryptic product ID. When you're ready continue on to the next topic to learn about adding more information to your events using field lookups.

Use field lookups

Use field lookups

The last topic walked you through using a subsearch. If you're not familiar with it, go back and review how to Use a subsearch.

This topic walks you through using field lookups.

What are field lookups?

Field lookups enable you to reference fields in an external CSV file that match fields in your event data. Using this match, you can enrich your event data by adding more meaningful information and searchable fields to them.

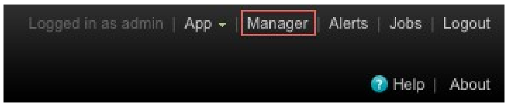
For an example that shows you how to use field lookups to add HTTP status code descriptions to your Web access event data, see this User manual topic.

In the previous example, you created a report table that listed how many items the top purchasing customer bought and which items they were. The items were listed by a product ID number that, on it's own, is pretty meaningless because you don't know what it refers to. Before you show this report to your boss and coworkers, you want to add the actual product name. This field doesn't exist in your data, so you need to use an external file.

To proceed, **download and uncompress this CSV file: product_lookup.csv.zip**

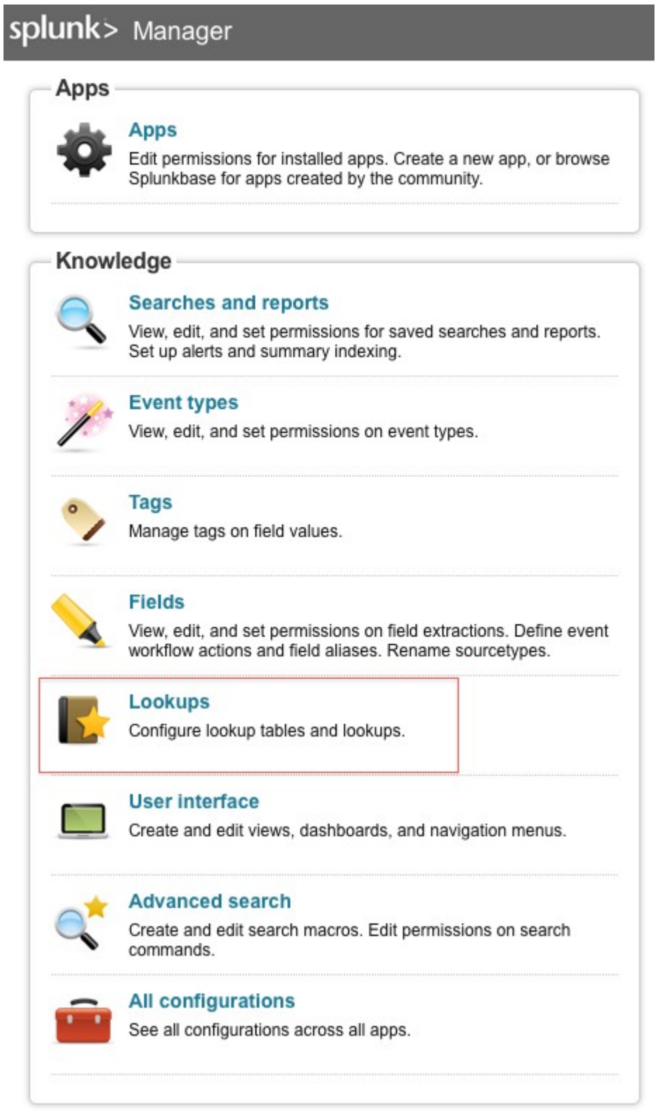
Find the Lookups manager

1. In the Splunk navigation menus, on the upper right corner, click on Manager.



This takes you to Splunk's Manager view.

2. Under **Apps and knowledge**, click *Lookups*.


splunk> Manager

Apps

Apps
Edit permissions for installed apps. Create a new app, or browse Splunkbase for apps created by the community.

Knowledge

Searches and reports
View, edit, and set permissions for saved searches and reports. Set up alerts and summary indexing.

Event types
View, edit, and set permissions on event types.

Tags
Manage tags on field values.

Fields
View, edit, and set permissions on field extractions. Define event workflow actions and field aliases. Rename sourcetypes.

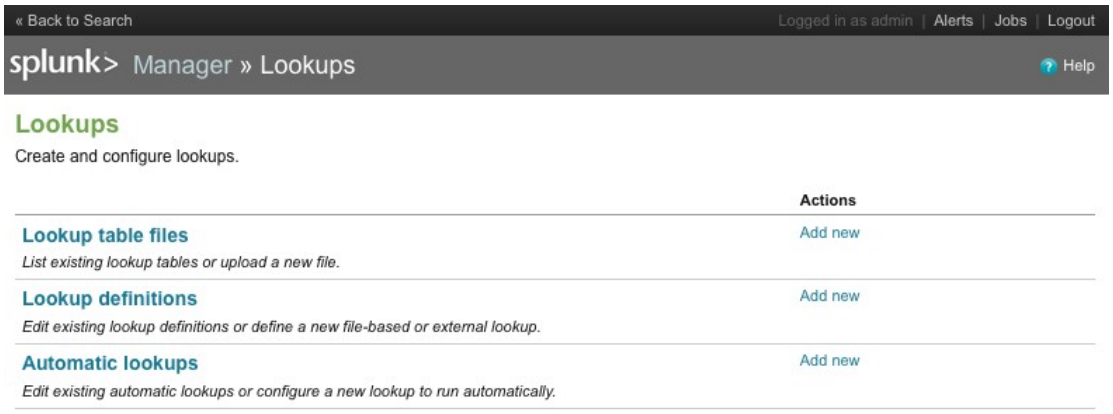
Lookups
Configure lookup tables and lookups.

User interface
Create and edit views, dashboards, and navigation menus.

Advanced search
Create and edit search macros. Edit permissions on search commands.

All configurations
See all configurations across all apps.

This takes you to the **Manager > Lookups** view.


« Back to Search   Logged in as admin | Alerts | Jobs | Logout

splunk> Manager » Lookups   Help

Lookups
Create and configure lookups.

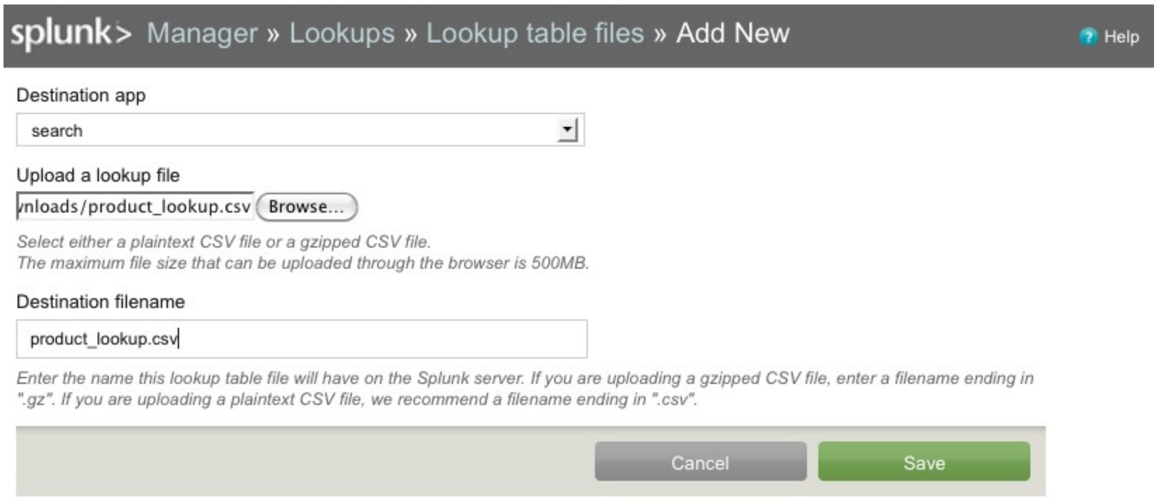
	Actions
Lookup table files *List existing lookup tables or upload a new file.*	Add new
Lookup definitions *Edit existing lookup definitions or define a new file-based or external lookup.*	Add new
Automatic lookups *Edit existing automatic lookups or configure a new lookup to run automatically.*	Add new

Upload the lookup file

In the **Manager > Lookups** view:

1. Under *Actions* for **Lookup table files**, click *Add New*.

This takes you to the **Manager > Lookups > Lookup table files** view where you upload CSV files to use in your definitions for field lookups.


splunk> Manager » Lookups » Lookup table files » Add New

Destination app: search

Upload a lookup file: wnloads/product_lookup.csv

Select either a plaintext CSV file or a gzipped CSV file.
The maximum file size that can be uploaded through the browser is 500MB.

Destination filename: product_lookup.csv

Enter the name this lookup table file will have on the Splunk server. If you are uploading a gzipped CSV file, enter a filename ending in ".gz". If you are uploading a plaintext CSV file, we recommend a filename ending in ".csv".

2. Leave the **Destination app** as *search*.

This tells Splunk to save your lookup table file in the Search app.

3. Under **Upload a lookup file**, browse for the CSV file (product_lookup.csv) to upload.

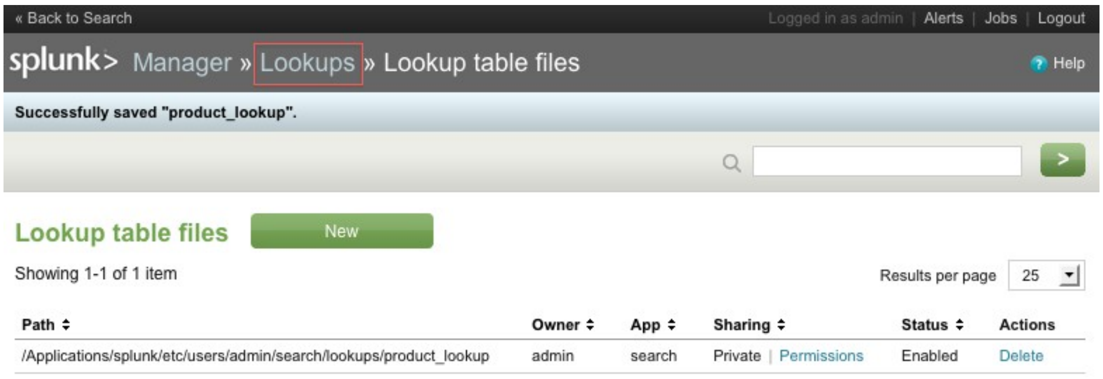
4. Under **Destination filename**, name the file **product_lookup.csv**.

This will be the name you use to refer to the file in a lookup definition.

5. Click *Save*.

This uploads your lookup file to Splunk to the Search app, but now you need to define the type of lookup you want to set up.

6. Return to **Manager > Lookups** by clicking the breadcrumb:


« Back to Search | Logged in as admin | Alerts | Jobs | Logout

splunk> Manager » Lookups » Lookup table files | Help

Successfully saved "product_lookup".

Lookup table files | New

Showing 1-1 of 1 item | Results per page 25

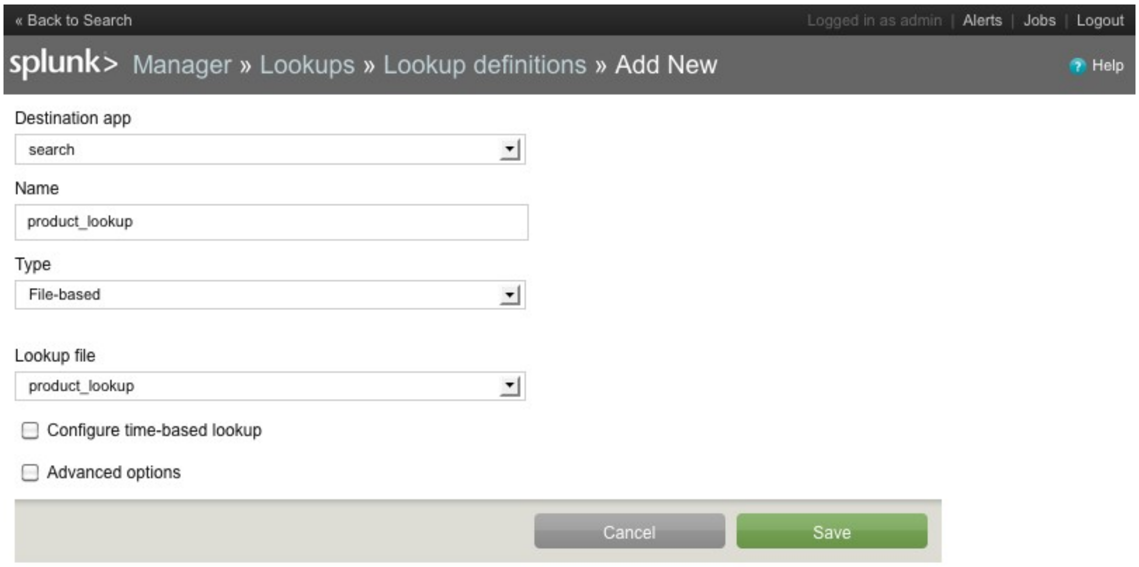
Path	Owner	App	Sharing	Status	Actions
/Applications/splunk/etc/users/admin/search/lookups/product_lookup	admin	search	Private | Permissions	Enabled	Delete

Define the field lookup

In the **Manager > Lookups** view:

1. Under *Actions* for **Lookup definitions**, click *Add New*.

This takes you to the **Manager > Lookups > Lookup table files** view where you define your field lookup.


« Back to Search | Logged in as admin | Alerts | Jobs | Logout

splunk> Manager » Lookups » Lookup definitions » Add New | Help

Destination app: search

Name: product_lookup

Type: File-based

Lookup file: product_lookup

☐ Configure time-based lookup

☐ Advanced options

Cancel | Save

2. Leave the **Destination app** as *search*.

3. Name your lookup **product_lookup**.

4. Under **Type**, select *File-based*.

5. Under **Lookup file**, select *product_lookup* (the name of your lookup table).

6. Leave **Configure time-based lookup** and **Advanced options** unchecked.

7. Click *Save*.

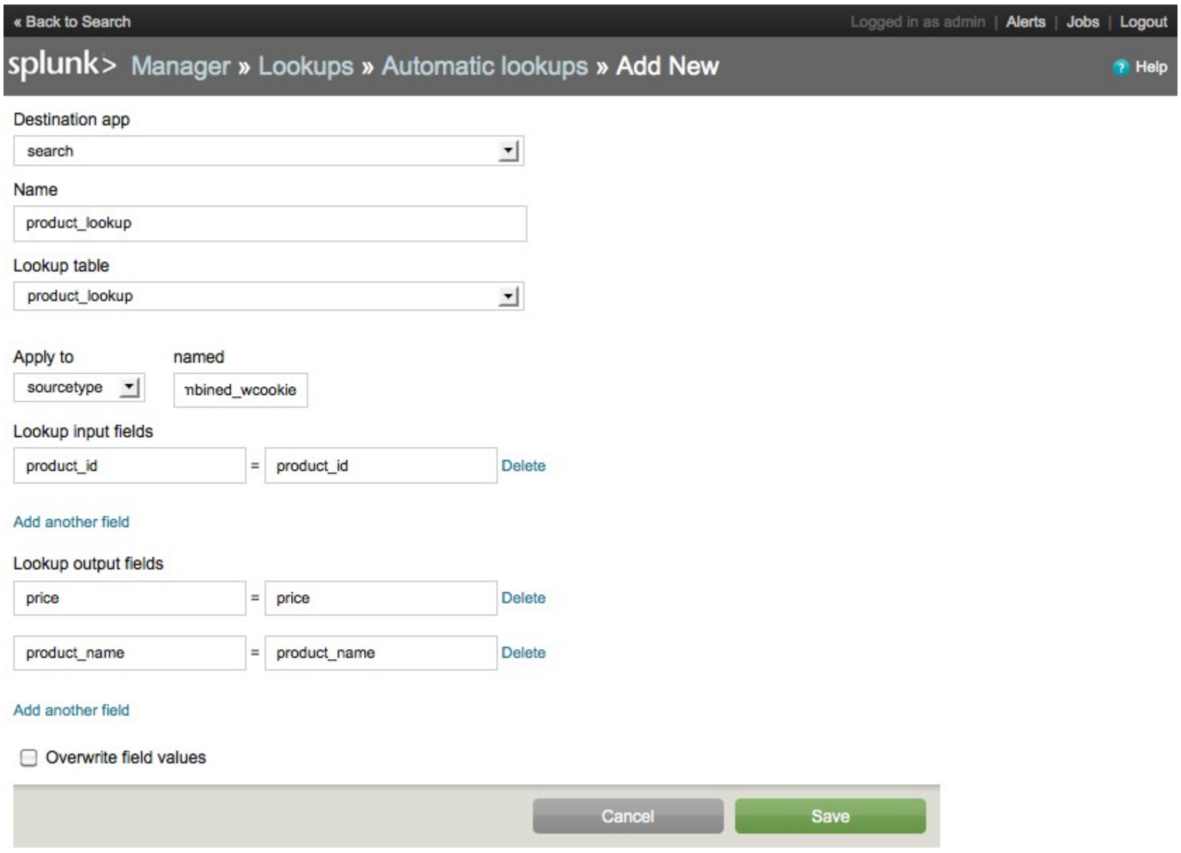
Now Splunk knows that product_lookup is a file-based lookup.

Make the lookup automatic

In the **Manager** > **Lookups** view:

1. Under *Actions* for **Automatic lookups**, click *Add New*.

This takes you to the **Manager** > **Lookups** > **Automatic lookups** >> **Add New** view where you configure the lookup to run automatically.


« Back to Search   Logged in as admin | Alerts | Jobs | Logout

splunk> Manager » Lookups » Automatic lookups » Add New   Help

Destination app: search

Name: product_lookup

Lookup table: product_lookup

Apply to: sourcetype   named: nbined_wcookie

Lookup input fields

product_id = product_id Delete

Add another field

Lookup output fields

price = price Delete

product_name = product_name Delete

Add another field

☐ Overwrite field values

Cancel Save

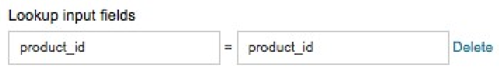
2. Leave the **Destination app** as *search*.

3. Name your automatic lookup **product_lookup**.

4. Under **Lookup table**, select *product_lookup*.

5. Under **Apply to** and **named**, select *sourcetype* and type in *access_combined_wcookie*.

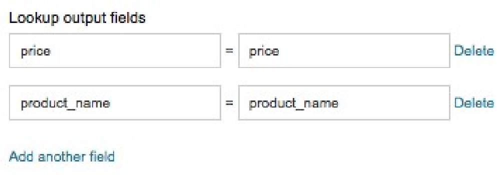
6. Under **Lookup input fields** type in:


Lookup input fields

product_id = product_id Delete

The input field is the field in your event data that you are using to match the field in the lookup table.

7. Under **Lookup output fields**, type in the following. Use the **Add another field** link to add more fields after the first one:



The output fields are the field(s) in the lookup table that you want to add to your event data based on the input field matching. Here, you are adding the fields: `price`, which contains the price for each `product_id`, and `product_name`, which contains the descriptive name for each `product_id`.

8. Leave **Overwrite field values** unchecked.

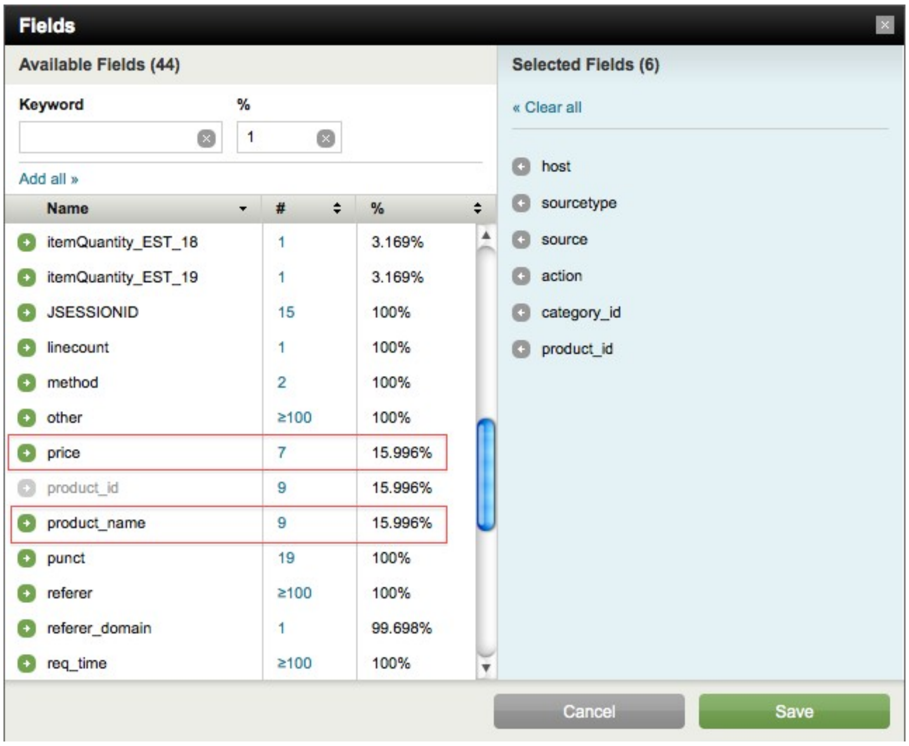
If you check this box, Splunk will overwrite any fields that exist in your event data with values from the corresponding field that you map to it from the lookup table. Since you are adding two new fields, you don't need to worry about this option.

9. Click *Save*.

Return to the Search dashboard (click **<< Back to Search**) and run the search for Web access activity over the time range, Yesterday:

```
sourcetype=access_*
```

When you scroll through the Fields menu or Fields picker, you should see the new fields that you added.



Fields

Available Fields (44)

Keyword:

%: 1

Add all »

Name	#	%
itemQuantity_EST_18	1	3.169%
itemQuantity_EST_19	1	3.169%
JSESSIONID	15	100%
linecount	1	100%
method	2	100%
other	≥100	100%
price	7	15.996%
product_id	9	15.996%
product_name	9	15.996%
punct	19	100%
referer	≥100	100%
referer_domain	1	99.698%
req_time	≥100	100%

Selected Fields (6)

« Clear all

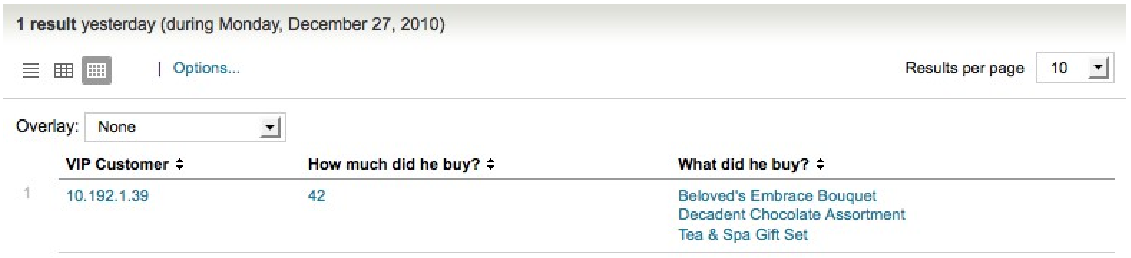
- host
- sourcetype
- source
- action
- category_id
- product_id

Search examples

Now you can run the previous subsearch example to see what the VIP customer bought. This time, replace the `product_id` field with the more readable `product_name`:

```
sourcetype=access_* action=purchase [search sourcetype=access_*
action=purchase | top limit=1 clientip | table clientip] | stats count,
values(product_name) AS product_name by clientip | sort - count | rename
count AS "How much did he buy?", product_name AS "What did he buy?",
clientip AS "VIP Customer"
```

The result is exactly the same as in the previous subsearch example, except that the VIP customer's purchases are more meaningful.



1 result yesterday (during Monday, December 27, 2010)

	VIP Customer	How much did he buy?	What did he buy?
1	10.192.1.39	42	Beloved's Embrace Bouquet Decadent Chocolate Assortment Tea & Spa Gift Set

Save the search as "VIP customer".

When you're ready, proceed to the next topic where you will run more searches.

More search examples

More search examples

In the last topic, you enriched your event data with new fields using a lookup table. If you didn't add those fields, go back and review how to use field lookups.

This topic walks you through more searches using what you learned from previous topics.

The search reference manual

These examples use only a handful of the search commands and functions available to you. For complete syntax and descriptions of usage of all the search commands, see the **Search reference manual**.

- The complete list of search commands
- The list of functions for the eval command
- The list of functions for the stats command

Back at the Flower & Gift shop, you're running searches to gather information to build a report for your boss about yesterday's purchase records:

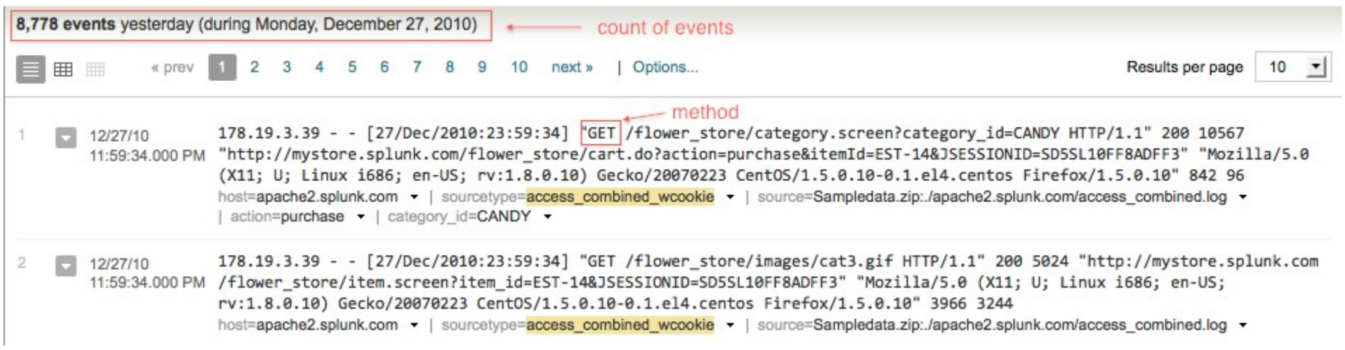
- How many page views were requested?
- How many page views were there compared to the number of purchases made?
- What was purchased and how much was made?
- How many purchase attempts failed?

Example 1

How many times did someone view a page on the website, yesterday?

1. Start with a search for all page views. Select the time range, *Other* > *Yesterday*:

```
sourcetype=access_* method=GET
```



8,778 events yesterday (during Monday, December 27, 2010)

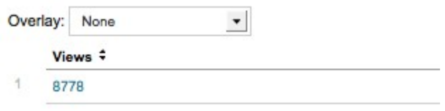
```
1  12/27/10 11:59:34.000 PM
178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/category.screen?category_id=CANDY HTTP/1.1" 200 10567
"http://mystore.splunk.com/flower_store/cart.do?action=purchase&itemId=EST-14&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0
(X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 842 96
host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log
| action=purchase | category_id=CANDY

2  12/27/10 11:59:34.000 PM
178.19.3.39 - - [27/Dec/2010:23:59:34] "GET /flower_store/images/cat3.gif HTTP/1.1" 200 5024 "http://mystore.splunk.com
/flower_store/item.screen?item_id=EST-14&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U; Linux i686; en-US;
rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3966 3244
host=apache2.splunk.com | sourcetype=access_combined_wcookie | source=Sampledata.zip:./apache2.splunk.com/access_combined.log
```

Next you want to count the number of page views (characterized by the `method` field).

2. Use the stats command:

```
sourcetype=access_* method=GET | stats count AS Views
```

Here, you use the `stats` command's `count()` function to count the number of "GET" events in your Web access logs. This is the total number of events returned by the search, so it should match the count of retrieved events. This search essentially captures that count and saves it into a field that you can use.



Overlay: None

	Views
1	8778

Here, renaming the `count` field as `Views` isn't necessary, but you're going to use it again later and this helps to avoid confusion.

3. Save this search as *Pageviews (Yesterday)*.

Example 2

From Example 1, you have the total number of views. How many visitors who viewed the site purchased an item? What is the percentage difference between views and purchases?

1. Start with the search from Example 1. Select the *Other > Yesterday* from the time range picker:

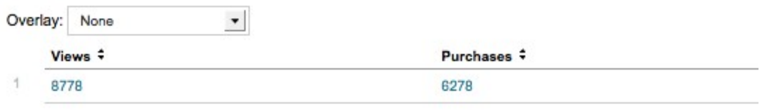
```
sourcetype=access_* method=GET | stats count AS views
```

2. Use stats to count the number of purchases (characterized by the `action` field):

```
sourcetype=access_* method=GET | stats count AS Views,
count(eval(action="purchase")) AS Purchases
```

You also use the count() function again, this time with an `eval()` function, to count the number of purchase actions and rename the field as `Purchases`.

Here, the renaming is required--the syntax for using an `eval()` function with the `stats` command requires that you rename the field.



Overlay: None

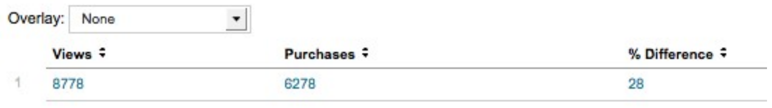
	Views	Purchases
1	8778	6278

Now you just need to calculate the percentage, using the total views and the purchases.

3. Use the `eval` command and pipe the results to `rename`:

```
sourcetype=access_* method=GET | stats count AS Views,
count(eval(action="purchase")) as Purchases | eval
percentage=round(100-(Purchases/Views*100)) | rename percentage AS "%
Difference"
```

The `eval` command enables you to evaluate an expression and save the result into a field. Here, you use the `round()` function to round the calculated percentage of `Purchases` to `Views` to the nearest integer.


Overlay: None

	Views	Purchases	% Difference
1	8778	6278	28

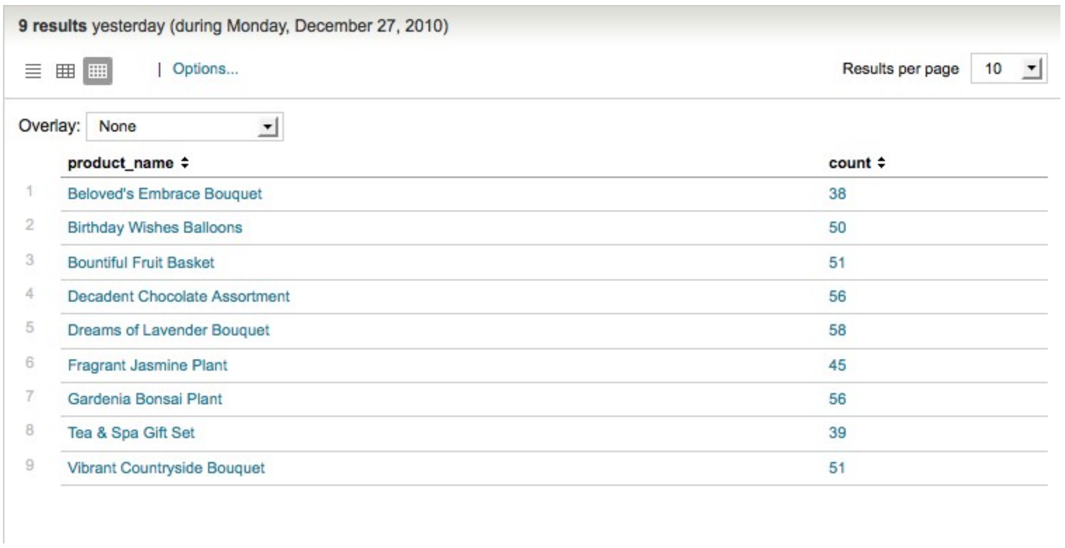
5. Save your search as "% Difference Purchases/Views".

Example 3

In the fields lookup example, you added the `product_name` and `price` fields to your data. Use this information to build a table to show what products were purchased yesterday, how many of each item was bought, and the calculated revenue for each product.

1. Start with a search for all purchases by the product name. Change the time range to *Other > Yesterday*:

```
sourcetype=access_* action=purchase | stats count by product_name
```


9 results yesterday (during Monday, December 27, 2010)

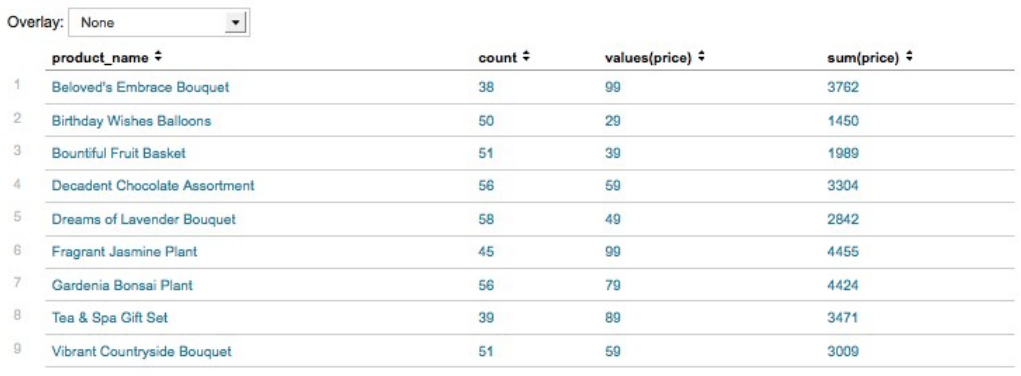
| Options... Results per page 10

Overlay: None

	product_name	count
1	Beloved's Embrace Bouquet	38
2	Birthday Wishes Balloons	50
3	Bountiful Fruit Basket	51
4	Decadent Chocolate Assortment	56
5	Dreams of Lavender Bouquet	58
6	Fragrant Jasmine Plant	45
7	Gardenia Bonsai Plant	56
8	Tea & Spa Gift Set	39
9	Vibrant Countryside Bouquet	51

2. Use `stats` functions to include the count of products purchased, price of each product, and the total revenue made for each product.

```
sourcetype=access_* action=purchase | stats count, values(price),
sum(price) by product_name
```


Overlay: None

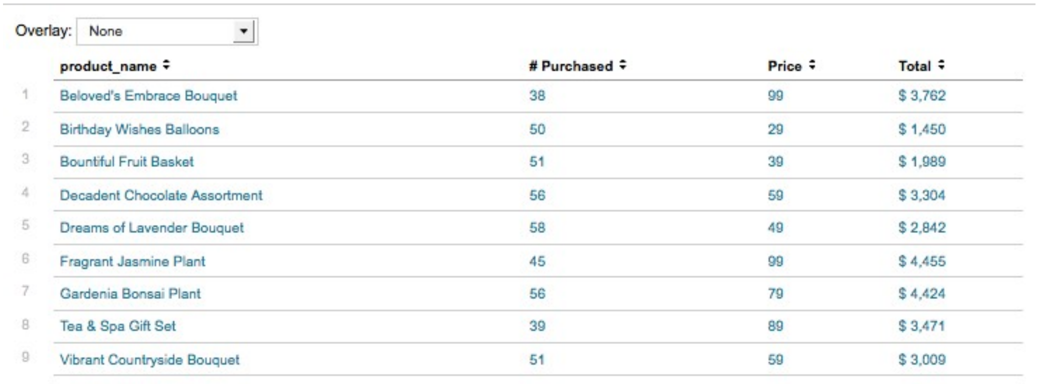
	product_name	count	values(price)	sum(price)
1	Beloved's Embrace Bouquet	38	99	3762
2	Birthday Wishes Balloons	50	29	1450
3	Bountiful Fruit Basket	51	39	1989
4	Decadent Chocolate Assortment	56	59	3304
5	Dreams of Lavender Bouquet	58	49	2842
6	Fragrant Jasmine Plant	45	99	4455
7	Gardenia Bonsai Plant	56	79	4424
8	Tea & Spa Gift Set	39	89	3471
9	Vibrant Countryside Bouquet	51	59	3009

The `count()` function just counts the number of events. The `values()` function returns the value of `price` for each `product_name`. And the `sum()` function adds together all the values of `price` for each `product_name`.

3. Now, you just need to rename the fields to make the table more readable:

```
sourcetype=access_* action=purchase | stats count AS "# Purchased",
values(price) AS Price, sum(price) AS Total by product_name | eval
Total="$ ".tostring(Total, "commas")
```

Here, you used the `eval` command's `tostring()` function to convert the price values to a string and reformat the values to include a dollar sign "$" and commas. Then, you used the `rename` command to rename the table headers.


Overlay: None

	product_name	# Purchased	Price	Total
1	Beloved's Embrace Bouquet	38	99	$ 3,762
2	Birthday Wishes Balloons	50	29	$ 1,450
3	Bountiful Fruit Basket	51	39	$ 1,989
4	Decadent Chocolate Assortment	56	59	$ 3,304
5	Dreams of Lavender Bouquet	58	49	$ 2,842
6	Fragrant Jasmine Plant	45	99	$ 4,455
7	Gardenia Bonsai Plant	56	79	$ 4,424
8	Tea & Spa Gift Set	39	89	$ 3,471
9	Vibrant Countryside Bouquet	51	59	$ 3,009

5. Save your search as *Purchases and Revenue (Yesterday)*.

Example 4

In the previous examples you searched for successful purchases, but you also want to know the count of purchase attempts that failed!

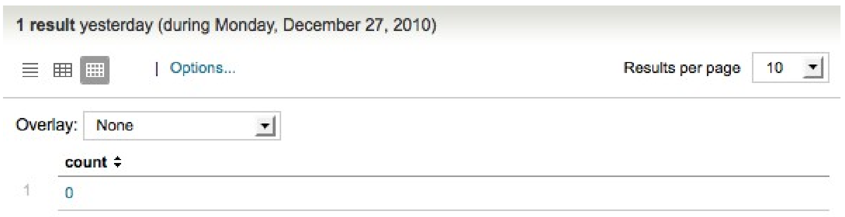
1. Run the search for failed purchase attempts, selecting *Yesterday* from the time range picker:

```
sourcetype=access_* action=purchase status=503
```
(You should recognize this search from the Start searching topic, earlier in this tutorial.)

This search returns the events list, so let's count the number of results.

2. Use the `stats` command:

```
sourcetype=access_* action=purchase status=503 | stats count
```
This returns a single value:


1 result yesterday (during Monday, December 27, 2010)

	count
1	0

This means that there were no failed purchases yesterday!

3. Save this search as *Failed purchases (Yesterday)*.

Now you should be comfortable using the search language and search commands. When you're ready, proceed to the next topic to learn how to create charts and build reports.

Reporting examples

Reporting examples

This topic builds on the searches that you ran and saved in the previous search examples.

This topic walks you through creating charts and building reports.

Splunk can dynamically update generated charts as it gathers search results. When you initiate a search, you can start building your report before the search completes. You can use the fields menu to quickly build simple pre-defined reports or use the **Report Builder**, which lets you define, generate, and fine-tune the format of your report, from the type of chart you want to create to the contents you want to display on this chart.

To learn more about using the report builder to define basic report parameters, format charts, and export or print finished reports, see "Define reports and generate charts" in this manual.

Back at the Flower & Gift shop, you're still building your reports. The previous searches you ran returned either a single value (for example, a count of failed errors) or a table of results (a table of products that were purchased). Now, you want to also add some pretty charts to your reports for yesterday's activities:

- The count of products purchased over time
- The count of purchases and views for each product category

Chart of purchases and views for each product

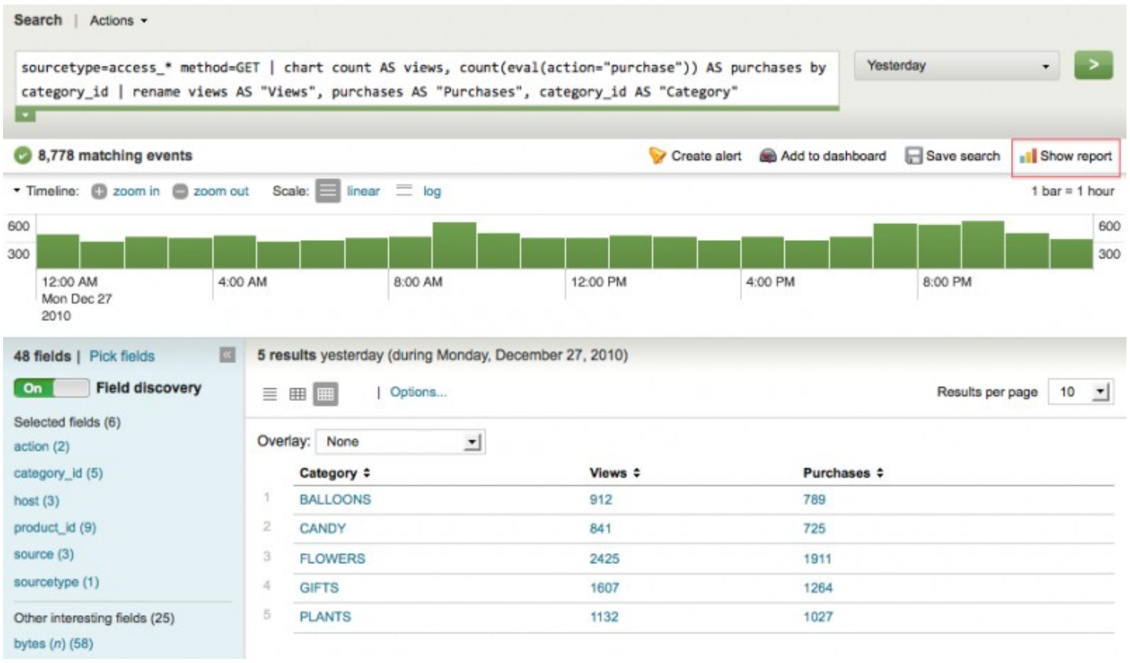
In this example, you want to build a chart of the number of views and number of purchases for each type of product. Recall that you saved a similar search in a previous topic.

Let's modify it a little.

1. Run this search over the time range, *Yesterday*:

```
sourcetype=access_* method=GET | chart count AS views,
count(eval(action="purchase")) AS purchases by category_id | rename views
AS "Views", purchases AS "Purchases", category_id AS "Category"
```

Here, you use the `chart` command instead of the `stats` command. The `chart` command enables you to create charts and specify the x-axis with the *by clause*.



```
sourcetype=access_* method=GET | chart count AS views, count(eval(action="purchase")) AS purchases by
category_id | rename views AS "Views", purchases AS "Purchases", category_id AS "Category"
```

8,778 matching events

5 results yesterday (during Monday, December 27, 2010)

	Category	Views	Purchases
1	BALLOONS	912	789
2	CANDY	841	725
3	FLOWERS	2425	1911
4	GIFTS	1607	1264
5	PLANTS	1132	1027

2. Click **Show report**.

Because you use the `chart` command and have already defined your report, this opens the **Format report** page of the Report Builder.

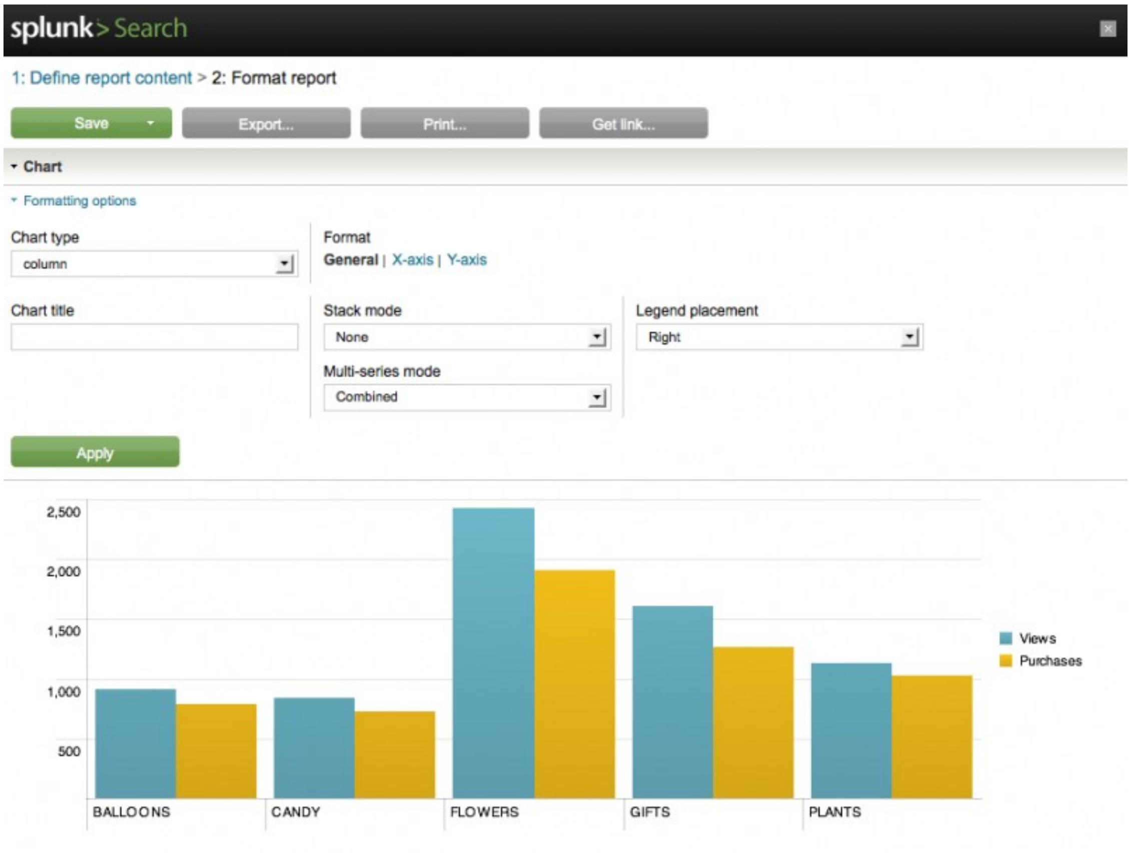


Chart type: column
Format: General | X-axis | Y-axis
Chart title:
Stack mode: None
Legend placement: Right
Multi-series mode: Combined

If you see something different in this window, for example a different chart type, it's probably because you're not looking at the default settings. You don't need to worry about this though.

If your search string includes reporting commands, you access the Report Builder by clicking Show report. Splunk will jump you directly to the formatting stage of the report-building process, since your reporting commands have already defined the report.

You don't need to have a strong understanding of reporting commands to use the Report Builder, but if you do have this knowledge the range of things you can do with the Report builder is increased.

3. Under Formatting options:

- Leave the chart type set to *column*.
- Name the chart, *Purchases and Views by Product Type*.

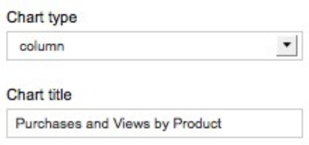
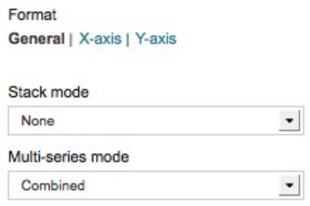


Chart type
column

Chart title
Purchases and Views by Product

Because you're using the `chart` command, you have to define the axes of the chart.

4. Under **General**, leave the settings as it is.


Format
General | X-axis | Y-axis

Stack mode
None

Multi-series mode
Combined

5. Under **Format**, click **X-axis**:

Type in "Product type" for the **X-axis title**.

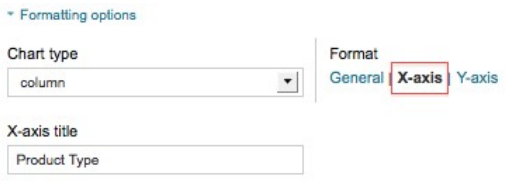


Chart type
column

Format
General | X-axis | Y-axis

X-axis title
Product Type

6. Under **Format**, click **Y-axis**:

Type in "Count of events" for the **y-axis title**.

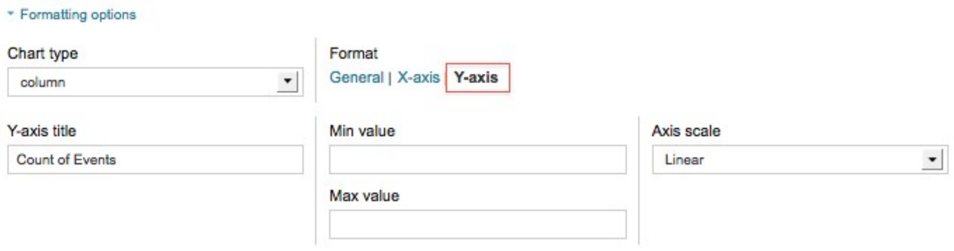


Chart type
column

Format
General | X-axis | Y-axis

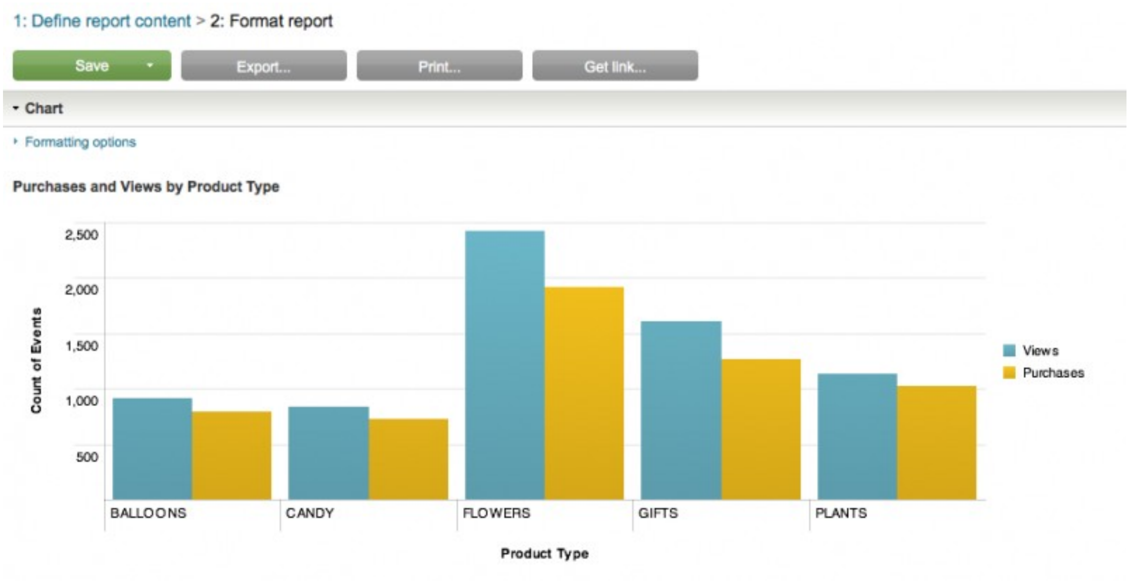
Y-axis title
Count of Events

Min value

Max value

Axis scale
Linear

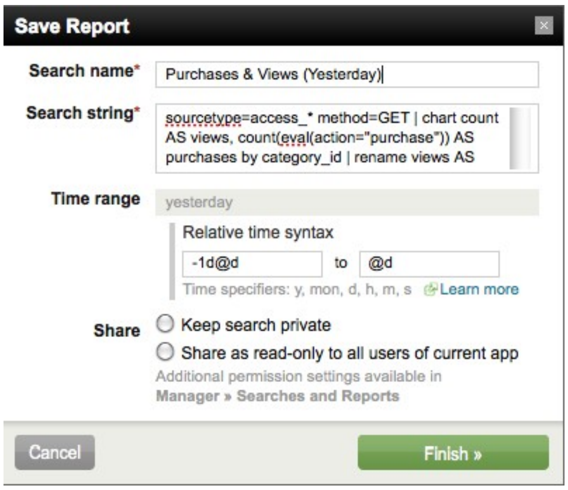
7. Click **Apply**.



Now you should see your chart of purchases and views formatted as a column chart with the types of products on the X-axis.

7. Click **Save** and select *Save report...*

The **Save report** dialog window opens:



Search name*: Purchases & Views (Yesterday)

Search string*:

```
sourcetype=access_* method=GET | chart count
AS views, count(eval(action="purchase")) AS
purchases by category_id | rename views AS
```

Time range: yesterday

Relative time syntax

-1d@d to @d

Time specifiers: y, mon, d, h, m, s Learn more

Share: ○ Keep search private ○ Share as read-only to all users of current app

Additional permission settings available in Manager » Searches and Reports

- Name your report *Purchases & Views (Yesterday)*.
- Click **Finish >>**.

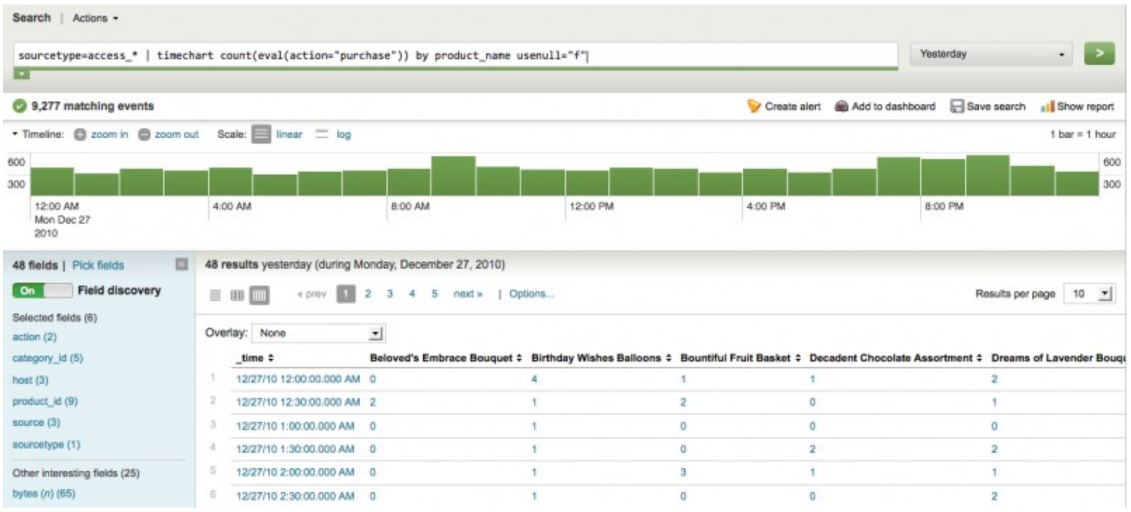
Top purchases by product name

For this report, you want to chart the number of purchases that were completed for each item yesterday.

1. Search for:

```
sourcetype=access_* | timechart count(eval(action="purchase")) by
product_name usenull="f"
```

Once again, use the `count()` function. But also, use the `usenull` argument to make sure the chart only counts events that have a value for `product_name`.



Search | Actions

```
sourcetype=access_* | timechart count(eval(action="purchase")) by product_name usenull="f"
```

Yesterday

9,277 matching events

Create alert | Add to dashboard | Save search | Show report

Timeline: zoom in | zoom out | Scale: linear | log | 1 bar = 1 hour

48 fields | Pick fields

On | Field discovery

Selected fields (6)
action (2)
category_id (5)
host (3)
product_id (9)
source (3)
sourcetype (1)

Other interesting fields (25)
bytes (n) (65)

48 results yesterday (during Monday, December 27, 2010)

« prev 1 2 3 4 5 next » | Options... Results per page 10

Overlay: None

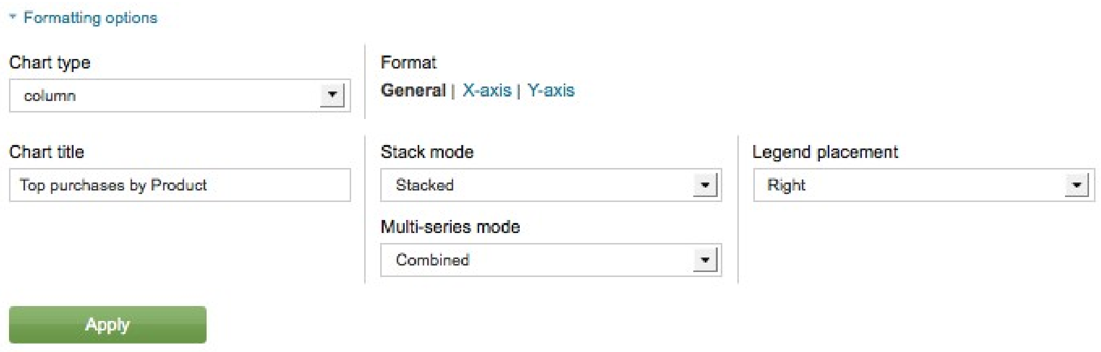
	_time	Beloved's Embrace Bouquet	Birthday Wishes Balloons	Bountiful Fruit Basket	Decadent Chocolate Assortment	Dreams of Lavender Bouq
1	12/27/10 12:00:00.000 AM	0	4	1	1	2
2	12/27/10 12:30:00.000 AM	2	1	2	0	1
3	12/27/10 1:00:00.000 AM	0	1	0	0	0
4	12/27/10 1:30:00.000 AM	0	1	0	2	2
5	12/27/10 2:00:00.000 AM	0	1	3	1	1
6	12/27/10 2:30:00.000 AM	0	1	0	0	2

2. Click **Show report**.

Because you used the `timechart` command in your search string, this takes you directly to Step 2 of report builder, where you Format your report.

3. Under Formatting options:

- Change the chart type to *column*.
- Name the chart, *Top purchases by Product*.
- Change the Stack mode to *Stacked*.



Formatting options

Chart type: column

Format: General | X-axis | Y-axis

Chart title: Top purchases by Product

Stack mode: Stacked

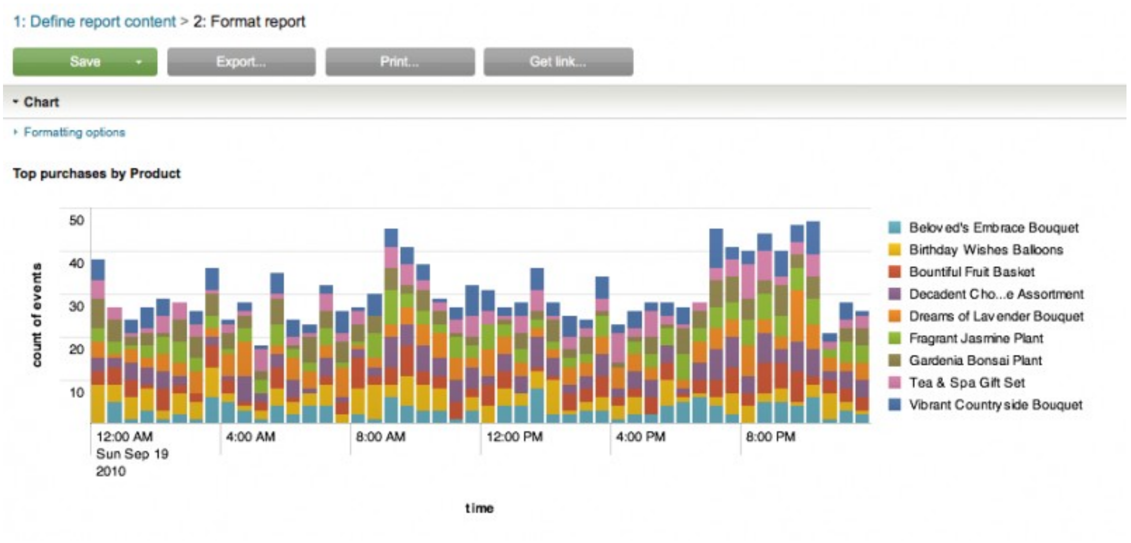
Legend placement: Right

Multi-series mode: Combined

Apply

Because you used the `timechart` command, the axes are already named: the x-axis is *time* and the y-axis is *count of events*.

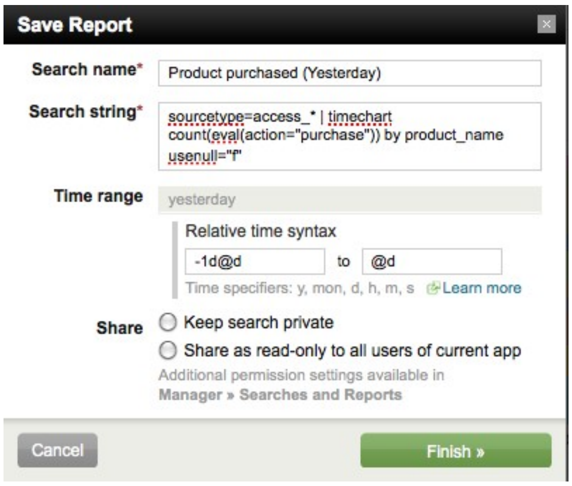
4. Click **Apply**.



Each of the columns represents the different products bought in that half-hour period.

5. Click **Save** and select *Save report*.

- Name your report *Products purchased (Yesterday)*.
- Add a description.
- Click Save.


Save Report

Search name* Product purchased (Yesterday)

Search string* sourcetype=access_* | timechart count(eval(action="purchase")) by product_name usenull="f"

Time range yesterday

Relative time syntax

-1d@d to @d

Time specifiers: y, mon, d, h, m, s Learn more

Share
- Keep search private
- Share as read-only to all users of current app

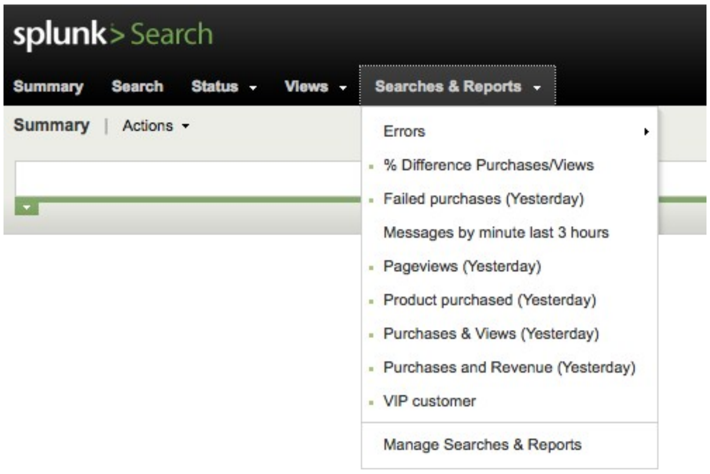
Additional permission settings available in Manager » Searches and Reports

There are alternate ways to access the Report builder:

- Click *Build report* in the **Actions** dropdown menu after you initiate a new search or run a saved search.
- Click a field in the search results sidebar to bring up the interactive menu for that field. Depending on the type of field you've clicked, you'll see links to reports in the interactive menu such as **average over time**, **maximum value over time**, and **minimum value over time** (if you've selected a numerical field) or **top values over time** and **top values overall** (if you've selected a non-numerical field). Click on one of these links, and Splunk opens the Format report page of the Report Builder, where it generates the chart described by the link.

Access saved reports

After you save a report, go **<< back to Search**. Splunk lists all your saved reports in the **Searches & Reports** menu on the search dashboard:


splunk> Search
Summary
Search
Status
Views
Searches & Reports
Summary
Actions
Errors
% Difference Purchases/Views
Failed purchases (Yesterday)
Messages by minute last 3 hours
Pageviews (Yesterday)
Product purchased (Yesterday)
Purchases & Views (Yesterday)
Purchases and Revenue (Yesterday)
VIP customer
Manage Searches & Reports

Save and share reports

When you're happy with the report you've created, you have a number of options for saving it and sharing it with others. Read more about saving your reports in "Save reports and share them with others".

You can also design specialized views and dashboards that include reports that you've defined. Dashboards can be made up of multiple panels that each display charts, lists, and other data that are generated by hidden, predefined searches. When you're ready, proceed to the next topic which walks you through creating and sharing a dashboard.

Build and share a dashboard

Build and share a dashboard

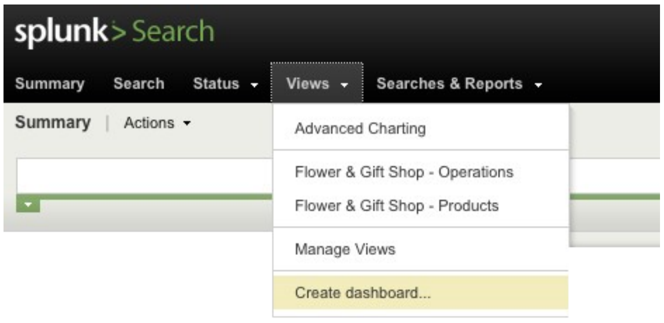
Before you proceed with this topic you should review Reporting on field values, where you have already built and saved a few reports. This topic walks you through creating simple dashboards that use the same searches and reports that you saved in the previous topics.

Back at the Flower & Gift Shop, your boss asks you to put together a dashboard to show metrics on the products sold at the shop. You also decide to build yourself a dashboard to help you find and troubleshoot any problems with the shop.

Flower & Gift Shop Products

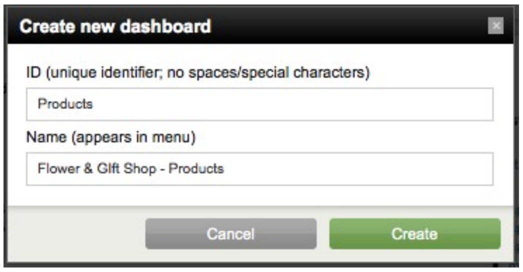
The first dashboard will show metrics related to the day-to-day purchase of different products at the Flower & Gift shop.

1. To start, open the **Views** drop-down list and click **Create dashboard...**.



This opens the **Create new dashboard** window.

2. Name your new dashboard.



Create new dashboard

ID (unique identifier; no spaces/special characters)

Products

Name (appears in menu)

Flower & Gift Shop - Products

2a. Designate a unique **ID** for it. This ID is the name you use to refer to the dashboard.

'*2b. **Give your dasboard a** Name*. This Name is the label that you will see listed in the navigation menus and at the top of your dashboard.

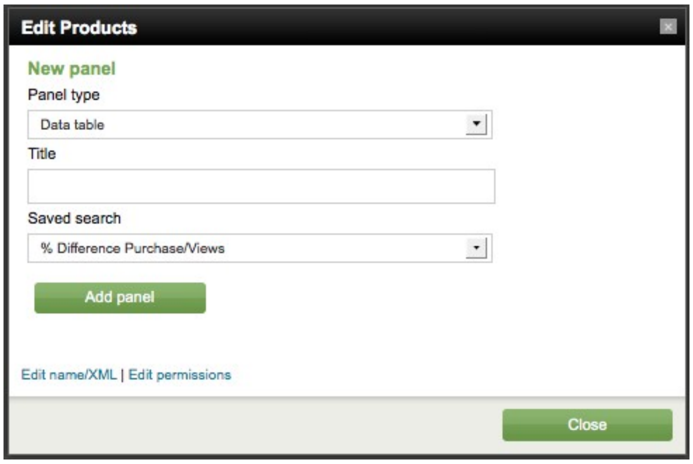
3. Click **Create** to create your new dashboard.

When your new dashboard appears, it is empty.

4. To start defining panels for it, click **Edit the dashboard** to open the visual dashboard editor.

Splunk's visual dashboard editor enables you to create simple dashboards quickly. All you need to get going is a set of saved searches and saved reports that Splunk can use to populate dashboard panels with useful metrics and charts.

The **Edit** window for a **New panel** opens.



Edit Products

New panel

Panel type: Data table

Title:

Saved search: % Difference Purchase/Views

The visual dashboard editor enables you to create four types of dashboard panels: data tables, charts, event lists, and single value panels.

All dashboard panels are associated with searches. You can determine whether a panel runs off of a predefined, saved search, or whether it uses a search that has been specifically designed for the panel and associated with it in an "inline" manner. For these dashboards, you'll just use saved searches and reports.

5. Add a **Chart** panel for the products purchased yesterday. (Purchases by Product Name)

The **Chart** panel type displays report results as a chart.

- Under **Panel type**, select *Chart*.
- Under **Title**, name the panel *Products purchased (Yesterday)*.
- Under **Saved search**, select the search "Products purchased (Yesterday)".
- Click **Add panel**.

The window expands to show the **Panel layout** section. This section shows a box which represents that panel you just defined. As you add more panels, you'll see more boxes here. This editor enables you to reorder and change the layout of the panels by dragging the boxes around.

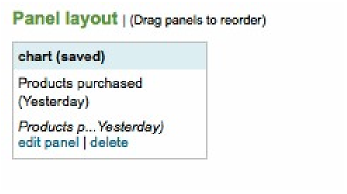

Panel layout | (Drag panels to reorder)

chart (saved)
Products purchased (Yesterday)
Products p... Yesterday)
edit panel | delete

Before you click **Save**, add a couple more panels, a chart and a data table:

6. Add a **Chart** panel for the count of purchases and views made yesterday. (# Purchases and Views)

- Under **Title**, name the panel *Purchases & Views (Yesterday)*.
- Under **Saved search**, select the saved search "Purchases & Views (Yesterday)".
- Click **Add Panel**.

7. Add a **Data table** panel for the products that were sold yesterday and how much was made yesterday.

The **Data table** panel type presents report results in tabular format.

- Under **Panel type**, select *Data table*.
- Under **Title**, name the panel *Products & Revenue (Yesterday)*.
- Under **Saved search**, select the saved search "Purchases and Revenue (Yesterday)".
- Click **Add Panel**.

8. Rearrange the boxes so that your layout looks like this:

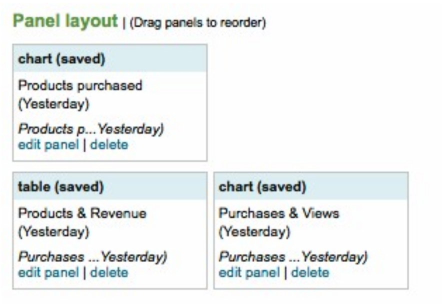
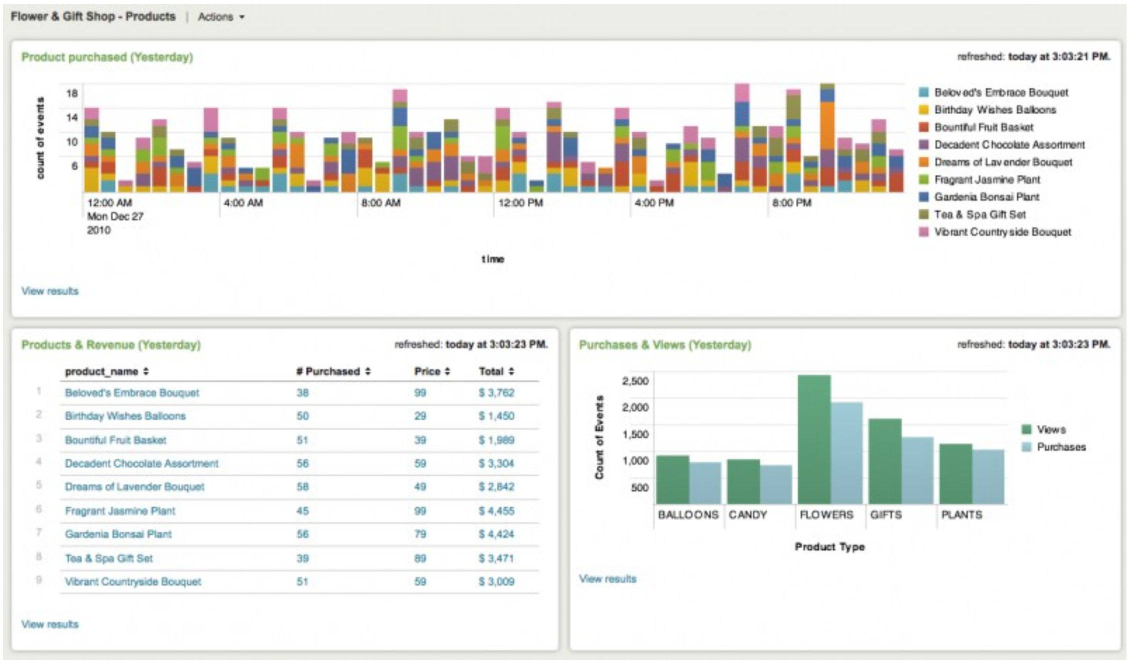

Panel layout | (Drag panels to reorder)

chart (saved)
Products purchased (Yesterday)
Products p... Yesterday)
edit panel | delete

table (saved)
Products & Revenue (Yesterday)
Purchases ... Yesterday)
edit panel | delete

chart (saved)
Purchases & Views (Yesterday)
Purchases ... Yesterday)
edit panel | delete

9 Click **Close**.



Products & Revenue (Yesterday)

refreshed: today at 3:03:23 PM.

	product_name	# Purchased	Price	Total
1	Beloved's Embrace Bouquet	38	99	$ 3,762
2	Birthday Wishes Balloons	50	29	$ 1,450
3	Bountiful Fruit Basket	51	39	$ 1,989
4	Decadent Chocolate Assortment	56	59	$ 3,304
5	Dreams of Lavender Bouquet	58	49	$ 2,842
6	Fragrant Jasmine Plant	45	99	$ 4,455
7	Gardenia Bonsai Plant	56	79	$ 4,424
8	Tea & Spa Gift Set	39	89	$ 3,471
9	Vibrant Countryside Bouquet	51	59	$ 3,009

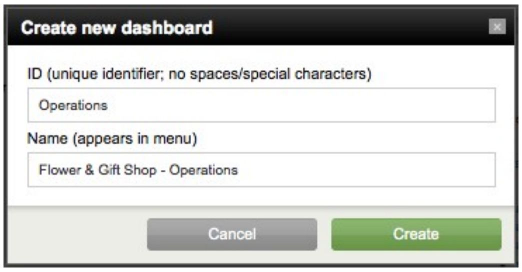
View results

Flower & Gift Shop Operations

The second dashboard includes simple reports that you can view at the start of your day to give you some information about recent Web access activity.

1. Return to the Search app. Select *Create new dashboard...* from the **Actions** menu.

2. Name your new dashboard:



ID (unique identifier; no spaces/special characters)

Operations

Name (appears in menu)

Flower & Gift Shop - Operations

3. Click **Edit Dashboard**.

The new panel edit window opens.

4. Add a *single value* panel for total number of website views yesterday.

The single value panel type displays a single numerical value as its result.

- Under **Panel type**, select *Single value*.
- Under **Title**, name the panel *Total views (Yesterday)*.
- Under **Saved search**, select the "Pageviews (Yesterday)".

- Click **Add panel**.

5. Add another single value panel to search for failed purchase attempts yesterday.

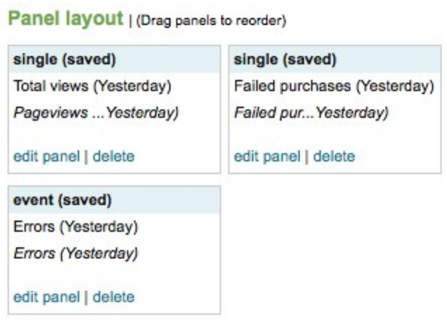
- Under **Title**, name the panel *Failed purchases (Yesterday)*.
- Under **Saved search**, select the "Failed purchases (Yesterday)".
- Click **Add panel**.

6. Add an *event listing* panel to search for any errors that occurred yesterday.

The event listing panel type displays your list of search results.

- Under **Panel type**, select *Event listing*.
- Under **Title**, name the panel *Errors (Yesterday)*.
- Under **Saved search**, select the "Errors (Yesterday)".
- Click **Add panel**.

7. Reorganize your panels to look like this:



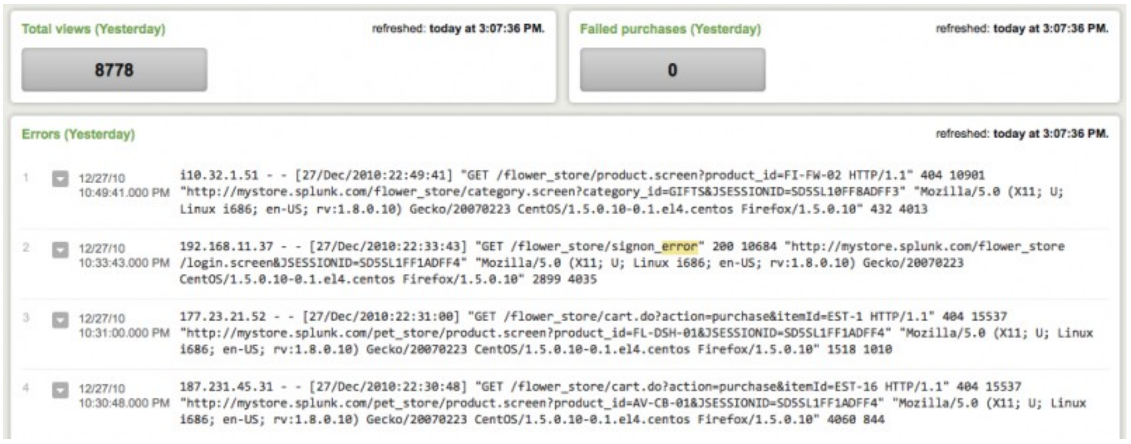
Panel layout | (Drag panels to reorder)

single (saved)
Total views (Yesterday)
Pageviews ... Yesterday)
edit panel | delete

single (saved)
Failed purchases (Yesterday)
Failed pur... Yesterday)
edit panel | delete

event (saved)
Errors (Yesterday)
Errors (Yesterday)
edit panel | delete

8. Click **Close**.



Total views (Yesterday) refreshed: **today at 3:07:36 PM.**
8778

Failed purchases (Yesterday) refreshed: **today at 3:07:36 PM.**
0

Errors (Yesterday) refreshed: **today at 3:07:36 PM.**

```
1  12/27/10          i10.32.1.51 - - [27/Dec/2010:22:49:41] "GET /flower_store/product.screen?product_id=FI-FW-02 HTTP/1.1" 404 10901
   10:49:41.000 PM   "http://mystore.splunk.com/flower_store/category.screen?category_id=GIFTS&JSESSIONID=SD5SL10FF8ADFF3" "Mozilla/5.0 (X11; U;
                     Linux i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 432 4013

2  12/27/10          192.168.11.37 - - [27/Dec/2010:22:33:43] "GET /flower_store/signon_error" 200 10684 "http://mystore.splunk.com/flower_store
   10:33:43.000 PM   /login.screen&JSESSIONID=SD5SL1FF1ADFF4" "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10) Gecko/20070223
                     CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 2899 4035

3  12/27/10          177.23.21.52 - - [27/Dec/2010:22:31:00] "GET /flower_store/cart.do?action=purchase&itemId=EST-1 HTTP/1.1" 404 15537
   10:31:00.000 PM   "http://mystore.splunk.com/pet_store/product.screen?product_id=FL-DSH-01&JSESSIONID=SD5SL1FF1ADFF4" "Mozilla/5.0 (X11; U; Linux
                     i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 1518 1010

4  12/27/10          187.231.45.31 - - [27/Dec/2010:22:30:48] "GET /flower_store/cart.do?action=purchase&itemId=EST-16 HTTP/1.1" 404 15537
   10:30:48.000 PM   "http://mystore.splunk.com/pet_store/product.screen?product_id=AV-CB-01&JSESSIONID=SD5SL1FF1ADFF4" "Mozilla/5.0 (X11; U; Linux
                     i686; en-US; rv:1.8.0.10) Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 4060 844
```

Share the dashboard

Select **Edit permissions** to expand or restrict the role-based read and write permissions for the dashboard. When you set dashboard permissions you can also define the app availability of the app. The dashboard can be:

- A private view available only to yourself.
- Available to one app only (the app it was designed in) and the people who have permission to use it.
- "Globally" available to all Splunk apps in your system (and therefore all of your Splunk users).

Index New Data

About data and indexes

About data and indexes

When you use Splunk, you are working with data in a Splunk index. In general, this manual assumes that a Splunk admin has already added data to your Splunk index. If this is the case, you can skip right to the "Search and investigate" chapter in this manual.

Read on to:

- Learn about the types of data Splunk indexes.
- See how to add new data to your Splunk index.

What types of data does Splunk index

Splunk can index any IT data from any source in real time. Point your servers or network devices' syslog at Splunk, set up WMI polling, monitor any live logfiles, enable change monitoring on your filesystem or the Windows registry, schedule a script to grab system metrics, and more. No matter how you get the data, or what format it's in, Splunk will index it the same way — without any specific parsers or adapters to write or maintain. It stores both the raw data and the rich index in an efficient, compressed, filesystem-based datastore — with optional data signing and auditing if you need to prove data integrity.

Ways to get data into Splunk

When adding data to Splunk, you have a variety of flexible input methods to choose from: Splunk Web, Splunk's CLI, and the inputs.conf configuration file.

You can add most data sources using Splunk Web. If you have access to the configuration files, you can use inputs.conf, which has more extensive configuration options. Any changes you make using Splunk Web or the Splunk CLI are written to inputs.conf.

The "Add data to your indexes" topic briefly outlines the general procedure for using Splunk Web to add new data. For more specifc information about configuring inputs, see the "What Splunk can index" chapter in the Getting Data In manual.

Where does Splunk store the data

You'll notice that we use the term "index" to refer to a couple of different things. First and foremost, when Splunk indexes new data, it processes the raw data to make it searchable. Second, when we talk about Splunk indexes, we mean the data store where Splunk stores all or parts of the data. So, when you *index* new data, Splunk stores the data in *indexes*. Additionally, when you search, you're matching against data in one or multiple indexes.

Apps and inputs

When you add an input to Splunk, that input gets added relative to the app you're in. Some apps, like the *nix and Windows apps that ship with Splunk, write input data to a specific index (in the case of *Nix and Windows, that is the 'os' index). If you're not finding data that you're certain is in Splunk, be sure that you're searching the right index.

For the Splunk user, this is all you need to know before you begin searching and learning more about your data. If you want to read more about managing the data in your indexes, see the "Manage indexes" chapter in the Admin manual.

Add data to your indexes

Add data to your indexes

As you read in the "About data and indexes" topic, Splunk can index logs, configuration files, traps and alerts, messages, scripts and code, and performance data from all your applications, servers and network devices. You can add most of these data sources via Splunk Web.

Access the data inputs configuration page

If you have the appropriate permissions, you can view and manage all of the data in your indexes from Splunk Manager's data inputs configuration page. To access this page:

1. Click the **Manager** link on the upper right hand corner of the screen. This link should always be available, regardless of the app you are currently in.

2. From the list of Splunk system configuration pages, click **Data inputs**. The data inputs configuration page displays a table listing the type of data and a count of the existing inputs for each type.

To add new data from files and directories, via TCP or UDP, or using a script, click the appropriate **Add new** link.

For more specifics about data inputs and how to add them, see "What Splunk can index" in the Getting Data In Manual.

Can't find the data you know is in Splunk?

When you add an input to Splunk, that input gets added relative to the app you're in. Some apps, like the *nix and Windows apps that ship with Splunk, write input data to a specific index (in the case of *Nix and Windows, that is the 'os' index). If you're not finding data that you're certain is in Splunk, be sure that you're searching the right index.

If you add an input, **Splunk adds that input to a copy of `inputs.conf` that belongs to the app you're in when you add that input.** This means that if you navigated to Splunk Manager, directly from the Launcher your input will be added to
`$SPLUNK_HOME/etc/apps/launcher/local/inputs.conf`.

Search and Investigate

About search

About search

Now you've got all that data in your system...what do you want to do with it? Start by using Splunk's powerful search functionality to look for anything, not just a handful of predetermined fields. Combine time and term searches. Find errors across every tier of your IT infrastructure and track down configuration changes in the seconds before a system failure occurs.

Splunk identifies fields from your records as you search, providing flexibility unparalleled by solutions that require setup of rigid field mapping rulesets ahead of time. Even if your system contains terabytes of data, Splunk enables you to search across it with precision.

In this chapter, you will:

- Start searching with simple terms, Booleans, wildcards, and fields.
- Learn how to search interactively with Splunk Web.
- Learn how to search across one or multiple indexes.
- Learn how to search across one or multiple Splunk servers.
- Perform actions on running and completed searches.
- See how to narrow your search by changing the time range.
- Use the timeline to investigate patterns of events.
- Learn how search commands work on your data.
- Search your data in real-time and preview reports.

Note: If you want to just jump right in and start searching, see the Search command cheat sheet for a quick reference complete with descriptions and examples.

Event data, fields, and search

When you search in Splunk, you're matching search terms against segments of your event data. We generally use the phrase *event data* to refer to your data after it has been added to Splunk's index. Events, themselves, are a single record of activity or instance of this event data. For example, an event might be a single log entry in a log file. Because Splunk breaks out individual events by their time information, an event is distinguished from other events by a timestamp.

Here's a sample event:

```
172.26.34.223 - - [01/Jul/2005:12:05:27 -0700] "GET
/trade/app?action=logout HTTP/1.1" 200 2953
```

Events contain pairs of information, or fields. When you add data and it gets indexed, Splunk automatically extracts some useful fields for you, such as the host the event came from and the type of data source it is.

You can use field names (sometimes called attributes or keys) and field values to narrow your search for specific event data. For more information about fields, see the **Data interpretation: Fields and**

field extractions chapter in the Knowledge Manager manual, beginning with the "About fields" topic.

Search and knowledge

As you search, you may begin to recognize patterns and identify more information that could be useful as searchable fields. You can configure Splunk to recognize these new fields as you index new data or you can create new fields as you search. Whatever you learn, you can use, add, and edit this knowledge about fields, events, and transactions to your event data. This capturing of knowledge helps you to construct more efficient searches and build more detailed reports.

For more information about capturing knowledge from your event data and adding information from external sources, see the "Capture knowledge" chapter in this manual.

Use the CLI to search

This chapter discusses search using Splunk Web. You can also execute searches on your Splunk server using the command line interface (CLI). For more information, you can read "About the CLI" and "Get help with the CLI" in the Admin manual.

Searching in Splunk

Searching in Splunk

The first time you use Splunk, you'll probably start by just searching the raw data to investigate problems — whether it's an application error, network performance problem, or security alert. Searching in Splunk is free form -- you can use familiar Boolean operators, wildcards and quoted strings to construct your searches. Type in keywords, such as a username, an IP address, a particular message... You're never limited to a few predetermined fields and you don't need to confront a complicated query builder, learn a query language, or know what field to search on. You can search by time, host and source.

This topic discusses how to start searching in Splunk Web from the Search app and by typing into the search bar. The following examples use Web access logs that contain the following information: IP addresses, browser versions, Web request protocols, HTTP status codes, website URLs, etc. Each of these examples stand alone; however, you can read the "Start searching tutorial" and "Use fields to search tutorial" for a more complete example.

Check the **Search Reference Manual** if you're looking for a reference for the commands in the Splunk search language or the Search command cheatsheet.

Go to the Search app

After logging into Splunk, you will see either the **Welcome** view or **Splunk Home** view.

- If you're in the **Welcome** view, select **Launch search app**.
- If you're in **Splunk Home**, select **Search**.
- If you are in another app, select the **Search** app from the **App** menu, which is located in the upper right corner of the window.

This takes you to the Summary dashboard of the Search app. For more information about what you will find in the Search App, read the Search App tutorial before you continue.

Start with simple terms

To begin your Splunk search, type in terms you might expect to find in your event data. For example, if you want to find events that might be HTTP 404 errors, type in the keywords:

```
http 404
```

Your search results are all events that have both HTTP and 404 in the raw text; this may or may not be exactly what you want to find. For example, your search results will include events that have website URLs, which begin with "http://", and any instance of "404", including a string of characters like "ab/404".

You can narrow the search by adding more keywords:

```
http 404 "not found"
```

Enclosing keywords in quotes tells Splunk to search for literal, or exact, matches. If you search for "not" and "found" as separate keywords, Splunk returns events that have both keywords, though not necessarily the phrase "not found".

You can also use Boolean expressions to narrow your search further.

Add Boolean expressions

Splunk supports the Boolean operators: `AND`, `OR`, and `NOT`; **the operators have to be capitalized**. You can use parentheses to group Boolean expressions. For example, if you wanted all events for HTTP client errors not including 404 or 403, search with:

```
http client error NOT (403 OR 404)
```

In a Splunk search, **the AND operator is implied**; the previous search is the same as:

```
http AND client AND error NOT (403 OR 404)
```

This search returns all events that have the terms "HTTP", "client", and "error" and do not have the terms "403" or "404". Once again, the results may or may not be exactly what you want to find. Just as the earlier search for `http 404` may include events you don't want, this search may both include events you don't want and exclude events you want.

Note: Splunk evaluates Boolean expressions in the following order: first, expressions within parentheses; then, `OR` clauses; finally, `AND` or `NOT` clauses.

Search with wildcards

Splunk supports the asterisk (`*`) wildcard for searching. Searching for `*` by itself means "match all" and returns all events up to the maximum limit. Searching for `*` as part of a word matches based on that word.

The simplest beginning search is the search for `*`. Because this searches your entire index and returns an unlimited number of events, it's also not an efficient search. We recommend that you begin with a more specific search on your index.

If you wanted to see only events that matched HTTP client and server errors, you might search for:

```
http error (40* OR 50*)
```

This indicates to Splunk that you want events that have "HTTP" and "error" and 4xx and 5xx classes of HTTP status codes. Once again, though, this will result in many events that you may not want. For more specific searches, you can extract information and save them as fields.

Search with fields

When you index data, Splunk automatically adds fields to your event data for you. You can use these fields to search, edit the fields to make them more useful, extract additional knowledge and save them as custom fields. For more information about fields and how to use, edit, and add fields, read the "Capture Knowledge" chapter in this manual.

Splunk lists fields that it has extracted in the Field Picker to the left of your search results in Splunk Web. Click a field name to see information about that field, add it to your search results, or filter your search to display only results that contain that field. When you filter your search with a field from the Field Picker, Splunk edits the search bar to include the selected field.

Alternately, you can type the field name and value directly into your search bar. A field name and value pair can be expressed in two ways: `fieldname="fieldvalue"` or `fieldname=fieldvalue`.

Note: Field names are case sensitive.

Let's assume that the event type for your Web access logs is `eventtype=webaccess` and you saved a field called `status` for the HTTP status codes in your event data. Now, if you wanted to search for HTTP 404 errors, you can restrict your search to the specific `field`:

```
status=404
```

Use wildcards to match multiple field values

If you're interested in seeing multiple values for the `status` field, you can use wildcards. For example, to search for Web access events that are HTTP client errors (4xx) or HTTP server errors (5xx), type:

```
eventtype=webaccess status=40* OR status=50*
```

Use comparison operators to match field values

You can use comparison operators to match a specific value or a range of field values.

Operator	Example	Result
=	field=foo	Field values that exactly match "foo".
!=	field!=foo	Field values that don't exactly match "foo".
<	field<x	Numerical field values that are less than x.
>	field>x	Numerical field values that are greater than x.
<=	field<=x	Numerical field values that are less than and equal to x.

>=	field>=x	Numerical field values that are greater than and equal to x.

Note: You can only use <, >, <=, and >= with numerical field values, and you can only use = and != with multi-valued fields.

Use tags to group field values

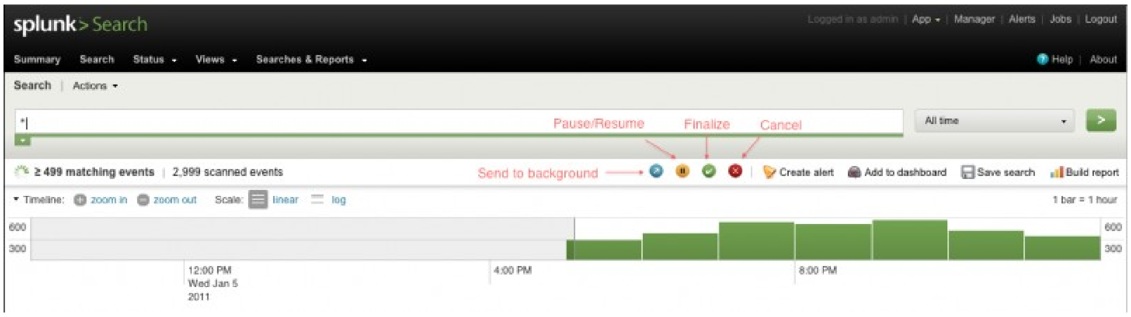
You can also use tags to group similar field values and search for fields based on these tags. For more information about tags, how to add them to field values, and examples of how to search for tagged field values, read "Tag and alias field values" in this manual.

Perform actions on running searches

Perform actions on running searches

Splunk provides a set of controls that you can use to manage "in process" searches. It displays these controls below the search bar while a search is running. The controls include:

- **Send to background** (blue button): Sends a search "to the background" while you work on other projects in the foreground, and has the system notify you when a backgrounded search is complete. You can use the Jobs page to access backgrounded **search jobs** and review their results.
- **Pause**/**Resume** (yellow button): Pauses a search in progress. Useful when you're running a long search but want to put it on hold momentarily. Click **Resume** to keep searching or **Finalize** to finalize the search (see below).
- **Finalize** (green button): Stops a search before it completes. Splunk will display the results that it has retrieved up to that point. You can use the finalized results to build a **report**.
- **Cancel** (red button): Cancels searches in progress and deletes all results. Splunk lists recently canceled searches in the Jobs page, but, because their results are deleted, it does not provide a **view** link for them.
- **Create alert**: Click to define an **alert** based on your search. Alerts run saved searches in the background (either on a **schedule** or in **real time**). When the search returns results that meet a condition you have set in the alert definition, the alert is triggered. For more information, see "Create an alert" in this manual.
- **Add to dashboard**: Click this if you'd like to generate a dashboard **panel** based on your search and add it to a new or existing **dashboard**. Learn more about dashboards in "Create and edit simple dashboards" in this manual.
- **Save search**: Saves the search, so you can easily run the search again without having to retype the search string. For more information, see "Save searches and share search results" in this manual.
- **Build report**: If you're dealing with a long search and don't want to wait until the search completes to start defining a **report** based on it, click this to launch the **Report Builder** and give yourself a head start. The search continues running after the Report Builder is launched, and the finished report covers the full range of the event data returned. For more information, see "Define reports and generate charts" in this manual.



For more information about using the Jobs page to track searches that have been backgrounded, canceled, or which are running for alerting purposes see "Supervise Your Search Jobs" in this manual.

Search interactively with Splunk Web

Search interactively with Splunk Web

Both the raw results and timeline are interactive, so you can click to drill down to events of interest, focus on anomalies, or eliminate noise to find the needle in a haystack. Whether you're troubleshooting a customer problem or investigating a security alert, you'll get to the bottom of what happened in minutes rather than hours or days.

In this topic, you'll learn how to:

- Use search results to narrow your search.
- Use the fields picker to add search terms.
- Use the Search assistant to help construct your searches.
- Use the Show source window to view raw events.

Use search results to narrow your search

Anytime after you run a search, you can highlight and select segments from your search results to add, remove, and exclude those keywords quickly and interactively.

Add new terms to your search

Fields and terms that you include in your search string will appear highlighted in the list of search results. Certain segments (words or phrases) in your search results will highlight as you move your mouse over the list; this indicates that you can add these terms to your search. To add any one of these other segments into your search, **click** it. Your search updates and filters out all the previous results that don't match.

For example, while searching for Web access events that are errors:

```
eventtype=webaccess errors
```
Perhaps, you notice one particular host machine, alpha, appears more frequently than others. You decide to just focus on this host. Instead of typing `host=alpha` into your search bar, you highlight one occurrence of the field value and click.

Your search string automatically appends the new filter and your search results update to reflect the new search:

```
eventtype=webaccess errors host=alpha
```

Remove existing terms from your search

Just as easily as you can add new search terms to your search, you can remove search terms. To do so, **click** on any of the highlighted segments in your list of search results.

For example, if you searched for Web access errors on a machine called alpha:

```
eventtype=webaccess errors host=alpha
```

Then, as you scroll through your results, you decide that you want to see what other Web access activity has occurred on alpha. To do this quickly and without having to edit your search string, you click on one highlighted occurrence of the term "errors" in your results. Your search string and results automatically update to match:

```
eventtype=webaccess host=alpha
```

Exclude terms from your search

As you scroll through the list of your search results, you may also notice events that are of no interest to you in your current investigation. To eliminate this noise without manually typing anything into your search bar, use **alt-click** (for Windows, use **ctrl-click**). Splunk updates your search to exclude the term you selected.

For example, if you searched for all Web access errors:

```
eventtype=webaccess errors
```

Then, you decide that you don't want to see any events from alpha; alt-click (or ctrl-click) on the host value in your results. Your search bar updates to read:

```
eventtype=webaccess errors NOT host=alpha
```

All events from alpha are removed from your list of search results.

Add search terms from available fields

Splunk automatically extracts fields from your data when you add it to your index. After you run a search, you'll notice that only three of these default fields display in your event data: host, sourcetype, and source. You can view all the other fields that Splunk identified (if they exist in these search results) and select to make them visible in your event data as well.

In the **Search view**, the field sidebar is on the left and underneath the Timeline. After you run a search, this sidebar contains the list of fields that are visible in your search results.

Click "Pick fields" underneath the Timeline, to open the **Fields** popup window. In the Fields window, you can view all the fields that are available in your search results. Select fields from this list to make visible in your search results.

To hide fields (that are already visible), you can click on them in the "Available Fields" list or the "Selected Fields" list. Click "Save" and you'll see your changes applied to the event data in your search results.

Use Search assistant to help construct your searches

Search assistant is a quick reference for users who are constructing searches. By default, search assistant is active; whenever you type terms into the search bar, it will give you typeahead information. When you type in search commands, it will give you descriptions and examples of usage for the command. You can access the search assistant within Splunk Web; click the green down-arrow under the search bar.

The default view displays a short description, some examples, common usage, and common next command. If the search bar is empty (there is no search command in it), Search assistant displays information for the `search` command.

You can also see lists of **common usage** and **common next commands** and expand the lists by clicking the **more** links next to the headers. When you click on any item in the lists, Splunk appends it to your search.

To see more information, click the **more >>** link at the end of the short description. This detailed view contains a longer description, command syntax, and related commands (if they are relevant). To return to the default view, click **<< less**.

Note: You can use the search assistant to quickly access the search command documentation; just click the **help** link next to the search command. This opens the search command's reference page in a new browser tab.

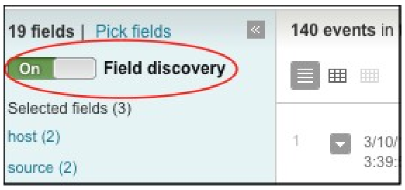
Use show source to view the raw event

After you run a search, you may want to view a particular result's raw format. To do this, click on the dropdown arrow at the left of the search result and select "Show Source". The **Show source** window opens and displays the raw data for the event you selected and some surrounding events.

You can also use the **Show source** window to view the validity of your indexed data. When you open **Show source**, the event that you selected to view is highlighted in yellow. Events highlighted in pink contain gaps in the data. Events highlighted in red may have been tampered with and are not valid. For example, if your data was tampered with, it may be indexed out of order and thus contain gaps.

Turn field discovery off to improve search performance

If you are running a search that ordinarily takes a long time to complete, you can set the **Field discovery** toggle to *Off* to make the search run faster.



The tradeoff is that turning off **Field discovery** disables automatic **field extraction**, except for fields that are required to fulfill your search (such as fields that you are specifically searching on) and **default fields** such as `_time`, `host`, `source`, and `sourcetype`. The search runs faster because Splunk is no longer trying to extract every field possible from your events.

Field discovery is set to *On* by default. You should leave it on if you don't know what fields exist in your data and think you might need them to help you narrow down your search in some way.

For more information about searching with fields, see the Capture Knowledge chapter of this manual. For general information about fields and field extraction, see "About fields" in the Knowledge Manager Manual.

Change the time range to narrow your search

Change the time range to narrow your search

With more flexible time range options, you can build more useful reports to compare historical data. For example, you may want to see how your system performs today, compared to yesterday and the day before. Or, you may only be interested in analyzing data during relevant time periods, such as web traffic during business hours.

This topic discusses how to apply absolute and relative time ranges to your search using:

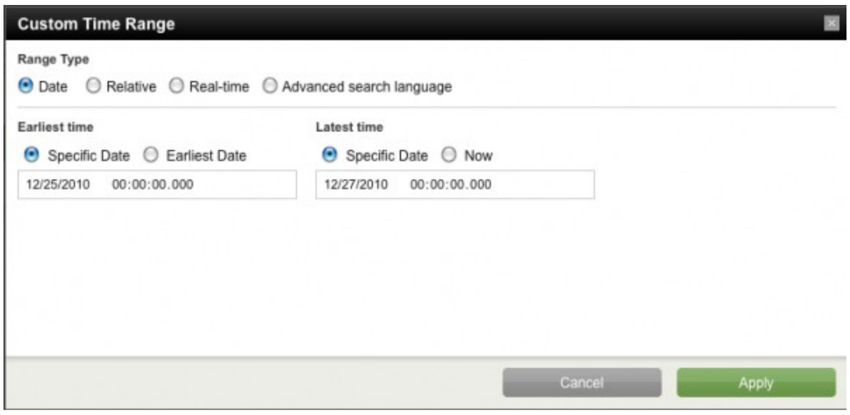
- the time range menu.
- search attributes **earliest** and **latest**.

Select time ranges to apply to your search

Use the **time range picker** dropdown to specify the time period you want to run your search. If you want to specify a custom date range, select **Custom time...** from the dropdown menu.

Then, select "Date" in the popup window. You can enter the date range manually or use the calendar pop-up.

For example, if you were interested in only events that occurred during the second business quarter, April through June, you might select the date range:

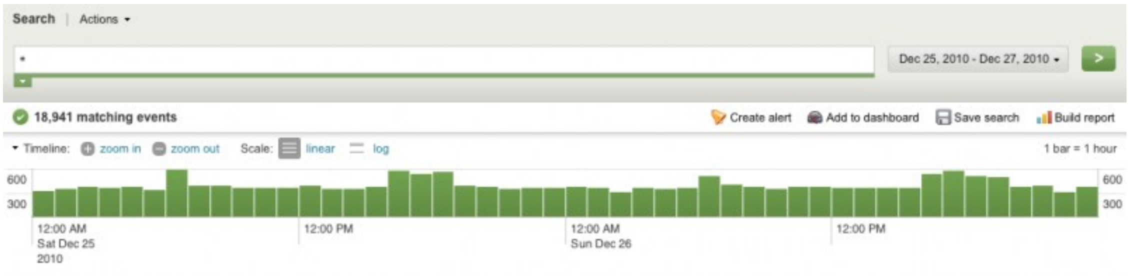

Custom Time Range

Range Type: [✓ Date] Relative Real-time Advanced search language

Earliest time: [✓ Specific Date] Earliest Date — 12/25/2010 00:00:00.000

Latest time: [✓ Specific Date] Now — 12/27/2010 00:00:00.000

The time range menu indicates the date range that you selected. Notice also that the timeline only shows the selected date range:



Note: If you are located in a different timezone from your server, time-based searches use the timestamp of the event from the server where it is indexed.

Specify absolute time ranges in your search

When searching or saving a search, you can specify time ranges using the following attributes:

```
earliest=<time_modifier>
latest=<time_modifier>
```

For exact time ranges, the syntax of `time_modifier` is: `%m/%d/%Y:%H:%M:%S`. For example, to specify a time range from 12AM October 19, 2009 to 12AM October 27, 2009:

```
earliest=10/19/2009:0:0:0 latest=10/27/2009:0:0:0
```

If you specify only the "earliest" attribute, "latest" is set to the current time (*now*) by default. In general, you won't specify "latest" without an "earliest" time.

When you specify a time range in your search or saved search, it overrides the time range that is selected in the dropdown menu. However, the time range specified directly in the search string will not apply to subsearches (but the range selected from the dropdown will apply).

Specify relative time ranges in your search

You can also use the `earliest` and `latest` attributes to specify relative time ranges using the syntax described as follows. Also, you can specify the time ranges in your search string or using the time range picker.

Syntax for relative time modifiers

You can define the relative time in your search with a string of characters that indicate time amount (integer and unit) and, optionally, a "snap to" time unit: `[+|-]<time_integer><time_unit>@<time_unit>`. Also, when specifying relative time, you can use *now* to refer to the current time.

1. Begin your string with a plus (+) or minus (-) to indicate the offset of the time amount.

2. Define your time amount with a number and a unit. When you specify single time amounts, the number is implied: 's' is the same as '1s', 'm' is the same as '1m', etc. The supported time units are:

- second: s, sec, secs, second, seconds
- minute: m, min, minute, minutes
- hour: h, hr, hrs, hour, hours
- day: d, day, days
- week: w, week, weeks
- month: mon, month, months
- quarter: q, qtr, qtrs, quarter, quarters
- year: y, yr, yrs, year, years

3. You can also specify a "snap to" time unit to indicate the nearest or latest time to which your time amount rounds down. To do this, separate the time amount from the "snap to" time unit with an "@" character.

You can define the relative time modifier as **only** a "snap to" time unit. For example, to "snap to" a specific day of the week, use @w0 for Sunday, @w1 for Monday, etc.

If you don't specify a "snap to" time unit, Splunk snaps automatically **to the second**.

Special time units

These abbreviations are reserved for special cases of time units and snap time offsets.

Time Unit	Description
`0 or all`	Specify a time range that is negative or positive 'infinity', depending on whether you use `earliest` or `latest`.
`now`	The current time.
`@q, @qtr, or @quarter`	Specify a snap to the beginning of the most recent quarter: Jan 1, Apr 1, July 1, or Oct 1.
`rt`	Specify a real-time search time window.
`w0, w1, w2, w3, w4, w5 and w6`	Specify "snap to" **days of the week**; where w0 is Sunday, w1 is Monday, etc. When you snap to a week, `@w` or `@week`, it is equivalent to snapping to Sunday or `@w0`.

More about snap time offsets

When snapping to the nearest or latest time, Splunk always **snaps backwards** or rounds down to the latest time not after the specified time. For example, if it is 11:59:00 and you "snap to" hours, you will snap to 11:00 not 12:00.

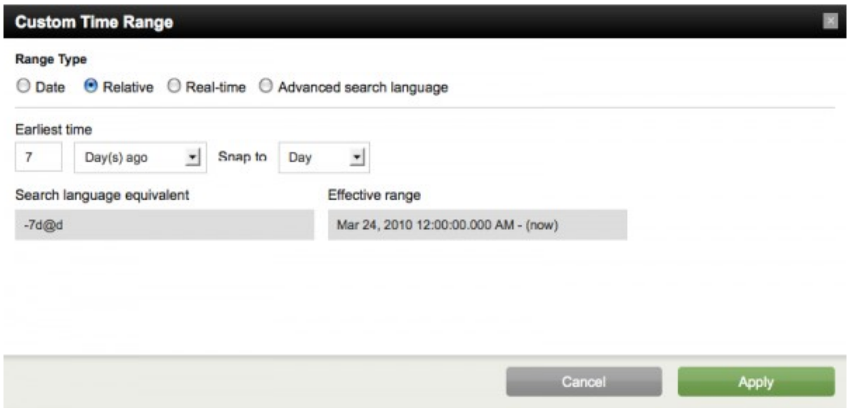
If you don't specify a time offset before the "snap to" amount, Splunk interprets the time as "current time snapped to" the specified amount. For example, if it is currently 11:59 PM on Friday and you use `@w6` to "snap to Saturday", the resulting time is the *previous* Saturday at 12:01 AM.

Define custom relative time ranges

1. From the time range picker, select **Custom time...**

2. Select **Relative** from the *Range Type* options.

3. Enter an **Earliest time** value.


Custom Time Range

Range Type

Date [✓ Relative] Real-time Advanced search language

Earliest time: 7 Day(s) ago Snap to Day

Search language equivalent: -7d@d

Effective range: Mar 24, 2010 12:00:00.000 AM - (now)

Note: You can also use this window to see the **Search language equivalent** of your earliest time value and the **Effective range** that it translates to in Splunk.

Examples of relative time modifiers

For these examples, the current time is Wednesday, 05 February 2009, 01:37:05 PM. Also note that 24h is usually but not always equivalent to 1d because of Daylight Savings Time boundaries.

Time modifier	Description	Resulting time	Equivalent modifiers
now	Now, the current time	Wednesday, 05 February 2009, 01:37:05 PM	+0, -0
-60m	60 minutes ago	Wednesday, 05 February 2009, 12:37:05 PM	-60m@s
-1h@h	1 hour ago, to the hour	Wednesday, 05 February 2009, 12:00:00 PM	
-1d@d	Yesterday	Tuesday, 04 February 2009, 12:00:00 AM	
-24h	24 hours ago (yesterday)	Tuesday, 04 February 2009, 01:37:05 PM	-24h@s
-7d@d	7 days ago, 1 week ago today	Wednesday, 28 January 2009, 12:00:00 AM	
-7d@m	7 days ago, snap to minute boundary	Wednesday, 28 January 2009, 01:37:00 PM	
@w0	Beginning of the current week	Sunday, 02 February 2009, 12:00:00 AM	

+1d@d	Tomorrow	Thursday, 06 February 2009, 12:00:00 AM	
+24h	24 hours from now, tomorrow	Thursday, 06 February 2009, 01:37:05 PM	+24h@s

Examples of chained relative time

You can also "chain" together the time modifiers for more specific relative time definitions.

Time modifier	Description	Resulting time
@d-2h	Snap to the beginning of today (12AM) and subtract 2 hours from that time.	10PM last night.
-mon@mon+7d	One month ago, snapped to the first of the month at midnight, and add 7 days.	The 8th of last month (at 12AM).

Examples of searches with relative time modifiers

Example 1: Web access errors from the beginning of the week to the current time of your search (now).

```
eventtype=webaccess error earliest=@w0
```

This search returns matching events starting from 12:00 AM of the Sunday of the current week to the current time. Of course, this means that if you run this search on Monday at noon, you will only see events for 36 hours of data.

Example 2: Web access errors from the current business week (Monday to Friday).

```
eventtype=webaccess error earliest=@w1 latest=+7d@w6
```

This search returns matching events starting from 12:00 AM of the Monday of the current week and ending at 11:59 PM of the Friday of the current week.

If you run this search on Monday at noon, you will only see events for 12 hours of data. Whereas, if you run this search on Friday, you will see events from the beginning of the week to the current time on Friday. The timeline however, will display for the full business week.

Example 3: Web access errors from the last full business week.

```
eventtype=webaccess error earliest=-7d@w1 latest=@w6
```

This search returns matching events starting from 12:00 AM of last Monday and ending at 11:59 PM of last Friday.

Customize the time ranges you can select

Splunk now ships with more built-in time ranges. Splunk administrators can also customize the set of time ranges that you view and select from the drop down menu when you search. For more information about configuring these new time ranges, see the times.conf reference in the *Admin Manual.*

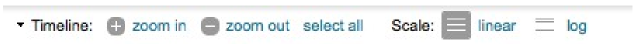
Use the timeline to investigate patterns of events

Use the timeline to investigate patterns of events

The timeline is a visual representation of the number of events that occur at each point in time. Thus, you can use the timeline to highlight patterns of events or investigate peaks and lows in event activity.

As the timeline updates with your search results, you might notice clusters or patterns of bars; the height of each bar indicates the count of events. Peaks or valleys in the timeline can indicate spikes in activity or server downtime.

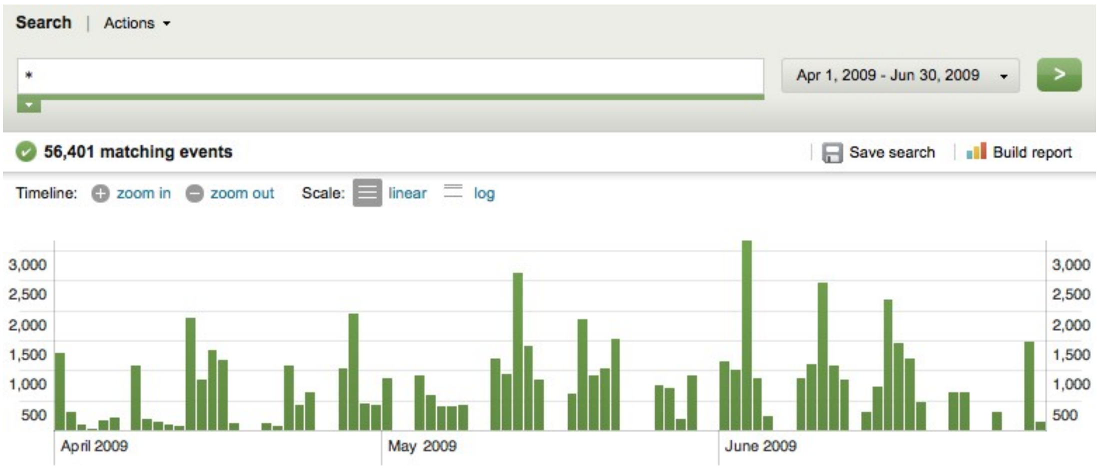
The timeline options are located above the timeline. You can zoom in and zoom out and change the scale of the chart.



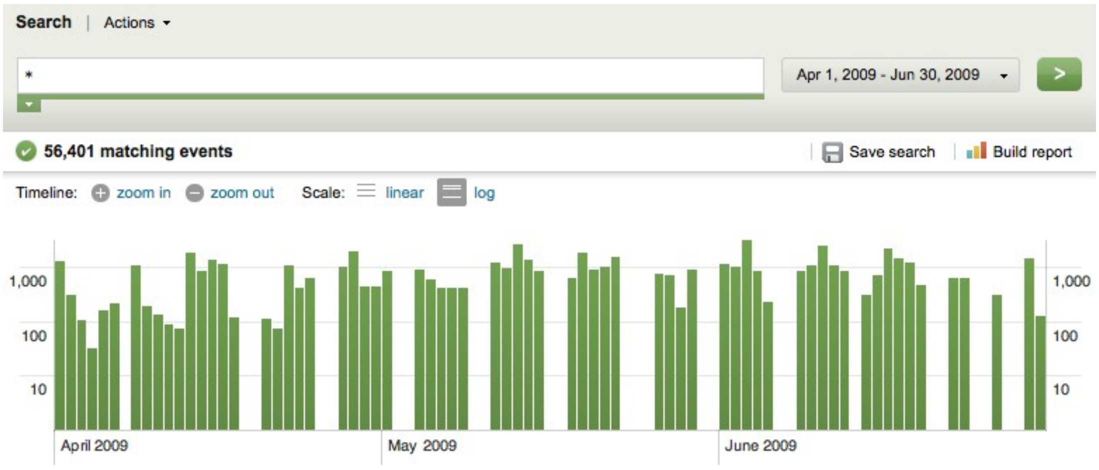
Change the scale of the timeline

You can view the timeline on two scales: linear or logarithmic (log).

The following image shows the search results for all events in the second quarter on a linear scale.



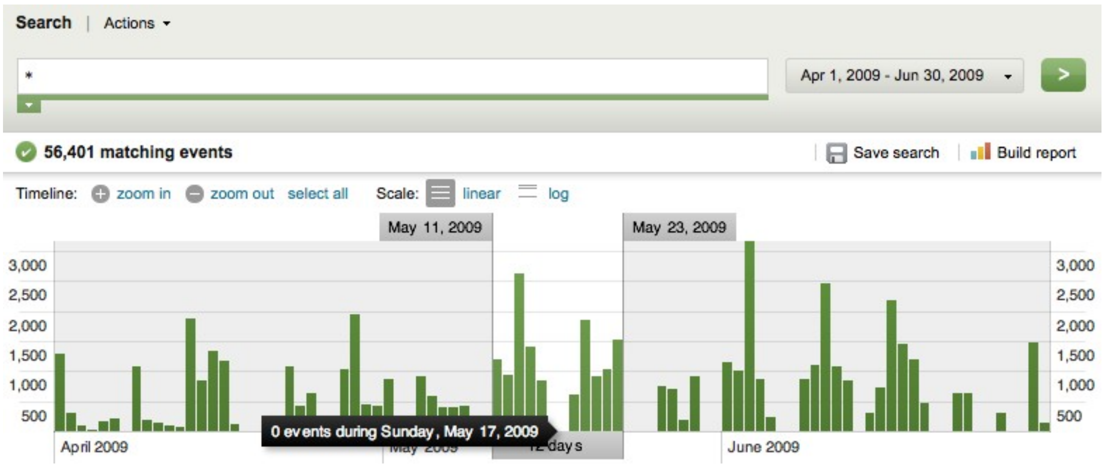
The following image shows the same search results for all events in the second quarter on a log scale.



Zoom in and zoom out to investigate events

Click and drag your mouse over a cluster of bars in the timeline.

- Your *search results* update to display only the events that occurred in that selected time range.
- If you click **zoom in**, the *timeline* updates to display only the span of events that you selected.



Click on one bar in the timeline.

- Your search results update to display only the events that occur at that selected point.
- Once again, if you click **zoom in**, the timeline updates to display only the events in that selected point.

If you want to select all the the bars in the timeline (undo your previous selection) click **select all**. This option is only available after you've selected one or more bars and before you selected either **zoom in** or **zoom out**.

Search and report in real time

Search and report in real time

With **real-time searches** and reports, you can search events before they are **indexed** and preview reports as the events stream in. This topic discusses:

- The mechanics behind real-time search and time range windows.
- How to invoke real-time search in **Splunk Web** and the **CLI**.
- Performance considerations for real-time searches and reports.
- How to disable real-time search in `indexes.conf`.
- How to set search and indexer limits on real-time searches.

You can design **alerts** based on real-time searches that run continuously in the background. Such **real-time alerts** can provide timelier notifications than alerts that are based on **scheduled searches**. For more information, see the "Create an alert" topic, in this manual.

You can also display real-time search results and reports in your custom **dashboards** using the **visual dashboard editor** and **simple XML**. For more information about the visual dashboard editor, see "Create simple dashboards with the visual dashboard editor" in this manual.

For more information about using real-time dashboards with advanced features that go beyond what the visual dashboard editor can provide, see Build a real-time dashboard in the *Developer manual*.

Note: When Splunk is used out-of-the-box, *only users with the Admin role* can run and save real-time searches. For more information on managing roles and assigning them to users, see "Add and edit roles" in the Admin Manual.

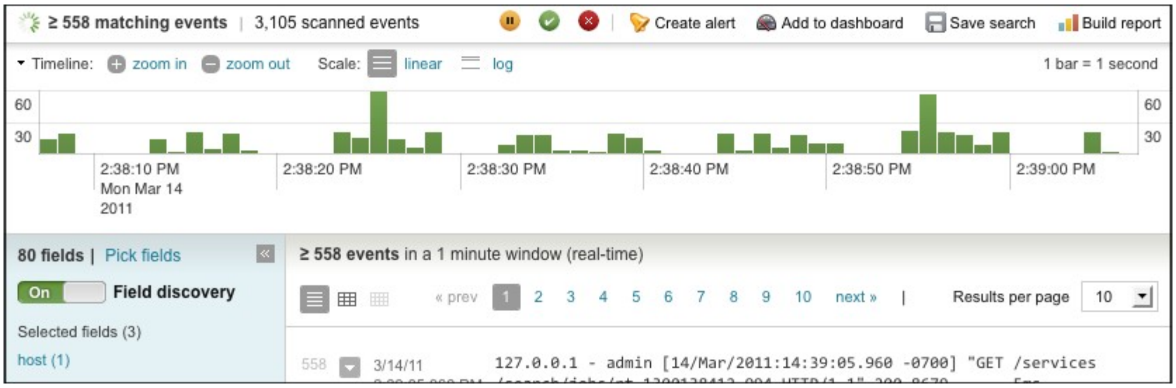
Real-time search mechanics

Real-time searches search through events as they stream into Splunk for indexing. When you kick off a real-time search, the Splunk scans incoming events that contain index-time fields that indicate they *could* be a match for your search. Splunk identifies these events in the UI as *scanned events*. This number is cumulative and represents the sum of all events scanned since the search was launched.

As the real-time search runs, Splunk periodically evaluates the scanned events against your search criteria to find actual matches. These events are identified in the UI as *matching events*. This number represents the number of matching events that currently exist within the sliding **time range window** that you have defined for the search. As such it can fluctuate up or down over time as Splunk discovers matching events at a faster or slower rate. If you are running the search in Splunk Web, the search timeline also displays the matching events that the search has returned within the chosen time range.

Here's an example of a real-time search with a one minute time range window. At the point that this screen capture was taken, the search had scanned a total of 3,105 events since it was launched. The matching event count of 558 represents the number of events matching the search criteria that had been identified in the past minute. This number fluctuated between 530 and 560 for the following minute; if it had spiked or dropped dramatically, that could have been an indication that something interesting was happening that required a closer look.

As you can see, the newest events are on the right-hand side of the timeline. As time passes, they move right until they move off the left-hand side, disappearing from the time range window entirely.



```
558  3/14/11  127.0.0.1 - admin [14/Mar/2011:14:39:05.960 -0700] "GET /services
```

A real-time search should continue running until you or another user stops it or deletes the search job; it should not "time out" for any other reason. If your events are stopping it could be a performance-related issue (see the subtopic "Expected performance and known limitations," below).

Real-time searches can take advantage of all Splunk search functionality, including advanced functionality like lookups, transactions, and so on. We've also designed search commands that are to be used specifically in conjunction with real-time searches, such as `streamstats` and `rtorder`.

Real-time searches in Splunk Web

You run a real-time search and build a real-time report in exactly the same way you run **standard searches**. However, because you are searching a live and continuous stream of data, the timeline updates as the events stream in and you can only view the report in preview mode. Also, some search commands are more applicable to real-time searches than standard searches. For example, streamstats and rtorder were designed for use in real-time searches.

To kick off a real-time search in Splunk Web, use the time range menu to select a preset **Real-time time range window**, such as *30 seconds* or *1 minute.* You can also specify a sliding time range window to apply to your real-time search.

Run this search to see pageview events as they stream in.

```
eventtype="pageview"
```

The raw events that are streamed from the input pipeline are not time-ordered. You can use the `rtorder` command to buffer the events from a real-time search and emit them in ascending time order.

The following example keeps a buffer of the last 5 minutes of pageview events, emitting events in ascending time order once they are more than 5 minutes old. Newly received events that are older than 5 minutes are discarded if an event after that time has already been emitted.

```
eventtype="pageview" | rtorder discard=t buffer_span=5m
```

Real-time search relies on a stream of events. Thus, you cannot run a real-time search with any other leading search command, such as `| metadata` which does not produce events or `| inputcsv` which just reads in a file. Also, if you try to send the search results to `| outputcsv`, the CSV file will not be written until the real-time search is Finalized.

Real-time reports in Splunk Web

Run a report to preview the IP addresses that access the most web pages. In this case, the `top` command returns a table with three columns: clientip, count, and percent. As the data streams in, the table updates with new values.

```
eventtype="pageview" | top clientip
```

For each pageview event, add a `count` field that represents the number of events seen so far (but do not include the current event in the count).

```
eventtype="pageview" | streamstats count current=f
```

You can also drilldown into real-time reports. However, real-time drilldown does not spawn another real-time search. Instead, it spawns a historic search, as you will drilldown into the events that have already been retrieved and indexed. For more information, see Understand table and chart drilldown actions in the *User manual*.

Real-time searches and reports in the CLI

To run a real-time search in the CLI, replace the command "search" with "rtsearch":

```
./splunk rtsearch 'eventtype=pageview'
```

Use the `highlight` command to emphasize terms in your search results. The following example highlights "GET" in your page view events:

```
./splunk rtsearch 'eventtype=pageview | highlight GET'
```

By default, search results have line wrapping enabled. Use the `-wrap` option to turn off line wrapping:

```
./splunk rtsearch 'eventtype=pageview' -wrap 0
```

Real-time reports in the CLI will also display in preview mode and update as the data streams in.

```
./splunk rtsearch 'error | top clientip'
```

Use the `-preview` option to suppress the results preview:

```
./splunk rtsearch 'error | top clientip' -preview false
```

If you turn off preview, you can still manage (Save, Pause, Finalize, or Delete) the search from the Jobs page in Splunk Web. After you finalize the search, the report table will display. For more information, see "Manage your search jobs" in this manual.

You can view all CLI commands by accessing the CLI help reference. For more information, see "Get help with the CLI" in this manual.

Specify real-time time range windows

Time bounds for historical searches are set at the time the search runs. With real-time searches, the time bounds are constantly updating and by default, the results accumulate from the start of the search. You can also specify a time range that represent a sliding window of data, for example, the last 30 seconds. When you specify a sliding window, Splunk takes that amount of time to accumulate data. For example, if your sliding window is 5 minutes, you will not start to see data until after the first 5 minutes have passed. You can override this behavior so that Splunk backfills the initial window with historical data before running in the normal real-time search mode (see "Real-time backfill," below).

You can specify real-time windows with pre-configured options listed in the time range picker, or by defining a custom real-time window in the time range picker.

Time ranges for real-time search follow the same syntax as for historical searches, except that you precede the relative time specifier with "rt", so that it's rt<time_modifier>.

The syntax for real-time time modifers is:
`rt[+|-]<time_integer><time_unit>@<time_unit>`. Read about the syntax for time modifiers in the topic, Change the time range of your search.

These values are not designed to be used from within the search language. They are configuration values that you can specify in the time range picker when you select *Custom > Real-time*. Also, you can use them in times.conf (to add options to the time range picker), or in the saved search dialog, or if you were directly using the REST API to access the Splunk back end search engine.

When you use time range windows with real-time searches, some of the events that occur within the latest second may not display in Splunk. This is expected behavior and is due to the latency between the timestamps within the events and the time when the event arrives. Because the time range window is with respect to the timestamps within the events and not the time when the event arrives, events that arrive after the time window won't display.

Real-time searches over "all time"

It's important to keep in mind that there is a small difference between real-time searches that take place within a set time window (30 seconds, 1 minute, 2 hours) and real-time searches that are set to "all time."

- **In "windowed" real time searches,** the events in the search can disappear as they fall outside of the window, and events that are newer than the time the search job was created can appear in the window when they occur.
- **In "all-time" real-time searches,** the window spans all of your events, so events do not disappear once they appear in the window, but events that are newer than the time the search job was created can appear in the window as they occur.
- In comparison, **in standard searches,** events never disappear from within the set range of time that you are searching and the latest event is always earlier than the job creation time (with the exception of searches that include events that have future-dated timestamps).

Real-time backfill

For real-time windowed searches, you can specify that Splunk backfill the initial window with historical data. This is run as a single search, just in two phases: first, a search on historical data to backfill events; then, a normal real-time search. Real-time backfill ensures that real-time dashboards seeded with data on actual visualizations and statistical metrics over time periods are accurate from the start.

You can enable real-time backfill in `limits.conf` in the [realtime] stanza:

```
[realtime]

default_backfill = <bool>
* Specifies if windowed real-time searches should backfill events
* Defaults to true
```

Expected performance and known limitations

Splunk's performance is expected to be acceptable as long as the indexers are not currently heavily loaded and do not have more than a few concurrent real-time searches. However, real-time searches will have a significant impact on performance in high volume environments and network load when you have many concurrent real-time searches.

You can run multiple real-time and historical searches concurrently, within the limits of your hardware. There are no restrictions on separate searches for the same or different users.

When planning your real-time searches, you should consider how it will affect the performance of both:

- The indexer that must forward the live events
- The searcher that must process the live events

The more work that is done on the indexer, the less that is required on the searcher, and vice versa. The indexer is important to the overall system function, so you do not want to burden it with too much filtering of live events. However, if the indexer does not filter at all, the bandwidth required to send all the live events to the searcher may prove costly, especially when multiple real-time searches running concurrently.

In the case where the searcher can't keep up with the indexer, the queue on the index processor will drop events. However, the events will have a sequence number, so we can tell when and how many events were dropped.

How to disable real-time search

Disable real-time search in indexes.conf

Searching in real time may be very expensive on the indexer. If you want to disable it on an indexer, you can edit a [default] setting in that indexer's `indexes.conf`.

```
[default]
enableRealtimeSearch = <bool>
```

Note: A *search head* that connects to multiple indexers will still be able to get real-time search results from the indexers that do have it enabled.

Disable real-time search for a user or role

Real-time search is a **capability** that you can map to specific users or roles in Splunk Web from **Manager > Access Controls**. By default, the **rtsearch** capability is assigned to the Admin and Power roles and not the User role. A role without the rtsearch capability will not be able to run a real-time search on that search head, regardless what indexers that search head is connected to.

Setting search limits on real-time searches

You can use the `[search]` stanza in `limits.conf` to change the maximum number of real-time searches that can run concurrently on your system.

```
[search]
max_rt_search_multiplier = <decimal number>
realtime_buffer = <int>
```

`max_rt_search_multiplier`

- A number by which the maximum number of historical searches is multiplied to determine the maximum number of concurrent real-time searches. Defaults to 3.
- **Note:** The maximum number of real-time searches is computed as: `max_rt_searches = max_rt_search_multiplier x max_hist_searches`

`realtime_buffer`

- The maximum number of accessible events to keep for real-time searches from the UI. Must be >= 1. Defaults to 10000.
- The real-time buffer acts as a circular buffer once this limit is reached.

Setting indexer limits for real-time search

You can use the `[realtime]` stanza in `limits.conf` to change the default settings for indexer support of real-time searches. **These options can be overridden for individual searches via REST API arguments.**

```
[realtime]
queue_size = <int>
blocking = [0|1]
max_blocking_secs = <int>
indexfilter = [0|1]
```

`queue_size = <int>`

- The size of queue for each real-time search. Must be > 0.
- Defaults to 10000.

`blocking =[0|1]`

- Specifies whether the indexer should block if a queue is full.
- Defaults to false (0).

```
max_blocking_secs = <int>
```

- The maximum time to block if the queue is full. This option is meaningless, if `blocking = false`.
- Means "no limit" if set to 0.
- Defaults to 60.

```
indexfilter = [0|1]
```

- Specifies whether the indexer should pre-filter events for efficiency.
- Defaults to true (1).

Specify one or multiple indexes to search

Specify one or multiple indexes to search

You have always been able to create new indexes and manage where you want to store your data. Now, when you have data split across different indexes, you're no longer limited to searching one index at a time — you can search across multiple indexes at once!

The Splunk administrator can set the default indexes that a user searches. Based on the user's roles and permissions, he may have access to one or many indexes; for example the user may only be able to search main or all public indexes. The user can then specify a subset of these indexes, either an individual index or multiple indexes, to search. For more information about setting up users and roles, see the "About users and roles" chapter in the Admin manual.

For more information about managing your indexes and setting up multiple indexes, see the "About managing indexes" chapter in the Admin manual.

Control index access via Splunk Web

Go into the Splunk Manager screen (click on 'Manager' in the top right corner), then click on 'Roles'. Select the role that the User has been assigned to and then on the bottom of the next screen you'll find the index controls. You can control the indexes that particular role has access to, as well as the default search indexes.

Syntax

You can specify different indexes to search in the same way that you specify field names and values. In this case, the field name is `index` and the field value is the name of a particular index:

```
index=<indexname>
```
You can use the * wildcard to specify groups of indexes; for example, if you wanted to search both "mail" and "main" indexes, you can search for:
```
index=mai*
```
You can also use parentheses to partition different searches to certain indexes. See Example 3 for details.

Note: When you type "index=" into the search bar, typeahead indicates all the indexes that you can search, based on your roles and permissions settings.

Examples

Example 1: Search across all public indexes.

```
index=*
```

Example 2: Search across all indexes, public and internal.

```
index=* OR index=_*
```

Example 3: Partition different searches to different indexes; in this example, you're searching three different indexes: main, _internal, and mail. You want to see events that match "error" in all three indexes; but also, errors that match "warn" in main or "failed" in mail.

```
(index=main (error OR warn)) OR (index=_internal error) OR (index=mail
(error OR failed))
```

Example 4: Search across multiple indexes on different distributed Splunk servers.

```
(splunk_server=local index=main 404 ip=10.0.0.0/16) OR
(splunk_server=remote index=mail user=admin)
```

Not finding the events you're looking for?

When you add an input to Splunk, that input gets added relative to the app you're in. Some apps, like the *nix and Windows apps that ship with Splunk, write input data to a specific index (in the case of *Nix and Windows, that is the 'os' index).

If you're not finding data that you're certain is in Splunk, be sure that you're looking at the right index. You may want to add the 'os' index to the list of default indexes for the role you're using. For more information about roles, refer to the topic about roles in this manual.

Search across one or more distributed search peers

Search across one or more distributed search peers

When performing a distributed search from a search head, you can restrict your searches to specific search peers (also known as "indexer nodes") by default and in your saved and scheduled searches. The names of your Splunk search peers are saved as values in the "splunk_server" field. For more information about distributed search, see "What is distributed search?" in the Distributed Deployment manual.

If no search peer is specified, your search accesses all search peers you have have permission to access. The default peers that you can access are controlled by the roles and permissions associated with your profile and set by your Splunk admin. For more information, see "About users and roles" in the Admin manual.

The ability to restrict your searches to specific peers can be useful when there is high latency to certain search peers and you do not want to search them by default. When you specify one or more peers, those are the only servers that are included in the search.

You can specify different peers to search in the same way that you specify other field names and values. In this case, the field name is "splunk_server" and the field value is the name of a particular distributed peer:

```
splunk_server=<peer_name>
```

Note: You can use the value "local" to refer to the Splunk instance that you are searching from; in other words, the search head itself.

```
splunk_server=local
```

Keep in mind that **field names are case sensitive**; Splunk will not recognize a field name if the case doesn't match.

Examples

Example 1: Return results from specified search peers.

```
error (splunk_server=NYsplunk OR splunk_server=CAsplunk) NOT
splunk_server=TXsplunk
```

Example 2: Search different indexes on distributed search peers "foo" or "bar".

```
(splunk_server=foo index=main 404 ip=10.0.0.0/16) OR (splunk_server=bar
index=mail user=admin)
```

How search commands work

How search commands work

A Splunk search consists of one or more data-generating commands and their arguments, which can include literal keywords, wildcards, Boolean expressions, field name and value expressions, and subsearches. The generated data (search results) can then be used as inputs into other search commands in a search pipeline.

For the most part, search commands fall into categories based on what they do, such as: filter unwanted information, extract more information, evaluate your data, transform your data into statistical results, and reorder your results. The specific commands themselves may fit more than one category depending on the arguments you use. Refer to the Search Reference manual for the complete list of search commands.

To better understand how search commands act on your data, it helps to visualize all your indexed data as a table. Each search command redefines the shape of your table. This topic illustrates how the different types of search commands act on your data.

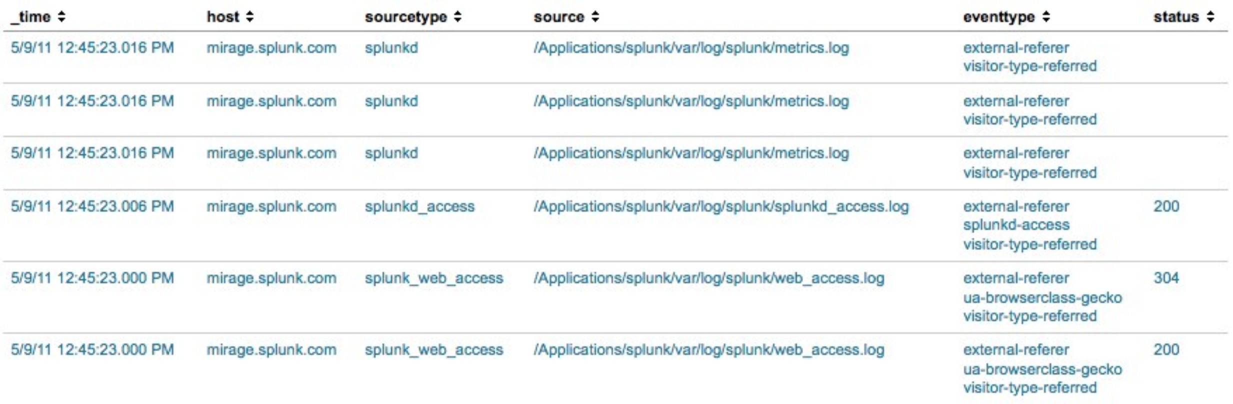
Also, if you want to just jump right in and start searching, the Search command cheatsheet is a quick reference complete with descriptions and examples.

Note: This topic uses Splunk's internal `metrics.log`, which are stored in `index=_internal`. You don't need to index any other data to follow these examples. If you want to index new data and follow along, read About data and indexes in the *User Manual*. You can also read about Working with metrics.log in the *Admin Manual*.

Start with a table of indexed data

Before you search, think of your indexed data as a table. In this table, **each indexed event is a row**. Each of these events contain indexed or extracted fields, which are name and value pairs of information. In this table, **the field names are columns**, and **the field values are the individual cells in each row**.

You can approximate this data table in Splunk if you search for everything in an index and select the Events Table view for your results. Restrict your time range to a short period of time, such as the Last 15 minutes or last hour and search for `index=_internal`. Then, use the field picker to add all the fields to the results view. Your table should look similar to this:



_time	host	sourcetype	source	eventtype	status
5/9/11 12:45:23.016 PM	mirage.splunk.com	splunkd	/Applications/splunk/var/log/splunk/metrics.log	external-referer visitor-type-referred	
5/9/11 12:45:23.016 PM	mirage.splunk.com	splunkd	/Applications/splunk/var/log/splunk/metrics.log	external-referer visitor-type-referred	
5/9/11 12:45:23.016 PM	mirage.splunk.com	splunkd	/Applications/splunk/var/log/splunk/metrics.log	external-referer visitor-type-referred	
5/9/11 12:45:23.006 PM	mirage.splunk.com	splunkd_access	/Applications/splunk/var/log/splunk/splunkd_access.log	external-referer splunkd-access visitor-type-referred	200
5/9/11 12:45:23.000 PM	mirage.splunk.com	splunk_web_access	/Applications/splunk/var/log/splunk/web_access.log	external-referer ua-browserclass-gecko visitor-type-referred	304
5/9/11 12:45:23.000 PM	mirage.splunk.com	splunk_web_access	/Applications/splunk/var/log/splunk/web_access.log	external-referer ua-browserclass-gecko visitor-type-referred	200

This table shows columns for a few of the internal and default fields Splunk automatically adds to the data. In your own results, you should see columns for many more default fields. These default columns are followed by columns for all other extracted fields.

Search at the beginning or elsewhere

You can search at any point in the search command pipeline. A search results in a smaller table that contains the exact same number of columns minus the rows of events that did not match the search conditions. Searches do not change any cell values.

Searching commands: crawl, savedsearch, search.

Example: Search for matching `host`.

Let's say you have this beginning table:

You want to find all the HTTP servers in your events:

```
host=http*
```

Filter unwanted information

Filtering commands produce the same results as a search: a smaller table. However, depending on the search command, the smaller table may have fewer rows or fewer columns. Filtering commands also do not change any cell values.

Filtering commands: dedup, fields, head, localize, regex, search, set, tail, where.

The following 3 examples use the same beginning table from the previous search example.

Example: Remove duplicates of cell values in a column with `dedup`.

You want to remove duplicate events based on the hostname:

```
* | dedup host
```

Example: Remove or keep columns with `fields`.

You want to see only the `host` and `sourcetype` information:

```
* | fields + host, sourcetype
```

Example: Remove all rows after the number specified with `head`.

You want to see only the first three results of your search:

```
* | head 3
```

Evaluate your data

Evaluating commands can change specific column names or cell values. Depending on the command, evaluating commands may or may not add columns.

Evaluating commands: abstract, addtotals, bucket, cluster, collect, convert, correlate, diff, eval, eventstats, format, fillnull, format, kmeans, makemv, mvcombine, mvexpand, nomv, outlier, overlap, replace, strcat, transaction, typelearner, xmlunescape.

The next example uses this beginning table; each succeeding example builds on it.

Example: Create a new column where the cells are the results of an `eval` expression.

You want to create a new field for the `sum` of `count1` and `count2` values.

```
* | eval sum=count1+count2
```

Example: Change one or more column names with `rename`. This does not create a new column.

Using the previous resulting table, you want to change the column name of `sum` to `total`

```
* | rename sum as total
```

Example: Overwrite cell values with `replace`. This does not create a new column.

Using the previous resulting table, you want to change all `host` values that are `host1` to `localhost`.

```
* | replace host1 with localhost in host
```

Example: Create new columns for the concatenated string value of other columns with `eval`.

Using the previous resulting table, you want to add a new column called `hosttype` that combines the `host` and `sourcetype` values, separated by a hyphen.

```
* | eval hostsourcetype=host."-".sourcetype
```

Reorder your results

Reording commands sort the rows of the entire table based on the values of the specified column name. These commands do not add or remove rows and do not change any cell values.

Reordering commands: reverse, sort.

Example: Reorder the table with `sort`.

Using the previous resulting table, reorder the rows in ascending order of `total`.

```
* | sort + total
```

Extract more information

Extracting commands create new rows or columns from information found in the `_raw` column for each row.

Extracting commands: addinfo, extract/kv, iplocation, multikv, rex, top, typer, xmlkv.

Example: Create new columns from key/value pairs in your events with `extract/kv`.

Example: Create new rows from information found in multi-line or tabular events with `multikv`.

Transform your data into statistical results

Transforming commands create an entirely new table of data. These commands change the specified cell values for each event into numerical values that Splunk can use for statistical purposes.

Transforming commands: chart, contingency, highlight, rare, stats, timechart, top.

Example: `chart`

How subsearches work

How subsearches work

A subsearch is a search with a search pipeline as an argument. Subsearches are contained in square brackets and evaluated first. The result of the subsearch is then used as an argument in the primary or outer search. You can use subsearches to match subsets of your data that you cannot describe directly in a search expression, but which can be generated from a search.

For example, if you're interested in finding all events from the **most active host in the last hour**, you can't search for a specific host because it might not be the same host every hour. First, you need to identify which host is most active.

```
sourcetype=syslog earliest=-1h | top limit=1 host | fields + host
```

Note that the previous search will only return one host value. Once you have this host, which is the most active host in the last hour, you can search for all events on that host. Let's say it's a server named, "crashy":

```
sourcetype=syslog host=crashy
```

But, instead of running two searches each time you want this information, you can use a subsearch to give you the hostname:

```
sourcetype=syslog [search sourcetype=syslog earliest=-1h | top limit=1
host | fields + host]
```

Modify subsearch limits in limits.conf

You can control the subsearch runtime and number of results by setting these limits in the `[subsearch]` stanza of a `limits.conf` file.

```
[subsearch]
maxout = <integer>
* Maximum number of results to return from a subsearch.
* This number cannot be greater than or equal to 10500.
* Defaults to 100.

maxtime = <integer>
* Maximum number of seconds to run a subsearch before finalizing
* Defaults to 60.

ttl = <integer>
* Time to cache a given subsearch's results.
* Defaults to 300.
```

Use subsearch to correlate data

You can use subsearches to correlate data, including data across different indexes or Splunk servers in a distributed environment.

For example, you may have two or more indexes for different application logs. The event data from these logs may share at least one common field. You can use the values of this field to search for events in one index based on a value that is not in another index:

```
sourcetype=some_sourcetype NOT [search sourcetype=another_sourcetype |
fields field_val]
```

Note: This is equivalent to the SQL "NOT IN" functionality:

```
SELECT * from some_table
WHERE field_value
NOT IN (SELECT field_value FROM another_table)
```

Change the format of subsearch results

When you use a subsearch, the `format` command is implicitly applied to your subsearch results. The format command changes your subsearch results into a single linear search string. This is used when you want to pass the returned values in the returned fields into the primary search.

If your subsearch returned a table, such as:

```
           | field1  | field2  |
            -------------------
event/row1 | val1_1  | val1_2  |
event/row2 | val2_1  | val2_2  |
```

The format command returns:

```
(field1=val1_1 AND field2=val1_2) OR (field1=val2_1 AND field2=val2_2)
```

For more information, see the format search command reference.

There are a couple of exceptions to this. First, all internal fields (fields that begin with a leading underscore "_*") are ignored and not reformatted in this way. Second, the "search" and "query" fields have their values rendered directly in the reformatted search string.

Using "search"

Generally,"search" can be useful when you need to append some static data or do some `eval` on the data in your subsearch and then pass it to the primary search. When you use "search", the first value of the field is used as the actual search term. For example, if field2 was "search" (in the table above), the format command returns:

```
(field1=val1_1 AND val1_2) OR (field1=val2_1 AND val2_2)
```

You can also use "search" to modify the actual search string that gets passed to the primary search.

Using "query"

"Query" is useful when you are looking for the values in the fields returned from the subsearch, but not in these exact fields. The "query" field behaves similarly to format. Instead of passing the field/value pairs, as you see with `format`, it passes the values:

```
(val1_1 AND val1_2) OR (val2_1 AND val2_2)
```

Examples

Let's say you have the following search, which searches for a `clID` associated with a specific `Name`. This value is then used to search for several sources.

```
index="myindex" [ index="myindex" host="myhost" <Name> | top limit=1 clID
| fields + clID ]
```

The subsearch returns something like: `( (clID="0050834ja") )`

If you want to return only the value, `0050834ja`, run this search:

```
index=myindex [ index=myindex host=myhost MyName | top limit=1 clID |
fields + clID | rename clID as search ]
```

If the field is named search (or query) the field name will be dropped and the subsearch (or technically, the implicit `| format` command at the end of the subsearch) will drop the field name and return ((0050834ja)). Multiple results will return, e.g., ((value1) OR (value2) OR (value3)).

This is a special case only when the field is named either "search" or "query". Renaming your fields to anything else will make the subsearch use the new field names.

Performance of subsearches

If your subsearch returns a large table of results, it will impact the performance of your search. **To change the maximum number of results that the format command operates over, append `| format maxresults = <integer>` to the end of the subsearch.** For more information, see the format search command reference.

After running a search you can click the **Actions** menu and select "Inspect Search". Scroll down to the `remoteSearch` component, and you can see what the actual query that resulted from your subsearch. Read more about the "Search Job Inspector" in the *Search reference manual*.

Answers

Have questions? Visit Splunk Answers and see what questions and answers the Splunk community has about using subsearches.

Create and use search macros

Create and use search macros

Search macros are chunks of a search that you can reuse in multiple places, including saved and ad hoc searches. Search macros can be any part of a search, such as an eval statement or search term, and do not need to be a complete command. You can also specify whether or not the macro field takes any arguments.

This topic discusses how to create and then use search macros via Splunk Web. For more information about how and why to use search macros, see the Design macro searches in the Knowledge Manager manual.

Create search macros in Splunk Web

In **Manager > Advanced Search > Search macros**, click "New" to create a new search macro.

Define the search macro and its arguments

Your search macro can be any chunk of your search string or search command pipeline that you want to re-use as part of another search.

- **Destination app** is the name of the app you want to restrict your search macro to; by default, your search macros are restricted to the Search app.
- **Name** is the name of your search macro, such as `mymacro`. If your search macro takes an argument, you need to indicate this by appending the number of arguments to the name; for example, if `mymacro` required two arguments, it should be named `mymacro(2)`. You can create multiple search macros that have the same name but require different numbers of arguments: `foo, foo(1), foo(2), etc`.

- **Definition** is the string that your search macro expands to when referenced in another search. If the search macro includes arguments, they are filled in and wrapped by dollar signs; for example, `$arg1$`.
- If **Eval Generated Definition?** is checked, then the 'Definition' is expected to be an eval expression that returns a string that represents the expansion of this macro.
- **Arguments** are a comma-delimited string of argument names. Argument names may only contain the characters: alphanumeric 'a-Z, A-Z, 0-9'; underscore '_'; and dash '-'. This list should not contain any repeated elements.

If a macro definition includes a leading pipe character ("|"), you may not use it as the first term in searches from the UI. Example: "| metadata type=sources". The UI does not do the macro expansion and cannot correctly identify the initial pipe to differentiate it from a regular search term. The UI constructs the search as if the macro name were a search term, which after expansion would cause the metadata command to be incorrectly formed and therefore invalid.

If a macro argument includes quotes, you need to escape the quotes when you call the macro in your search. For example, if you wanted to pass a quoted string as your macro's argument, you would use: ``` `my-macro("He said \"hello!\"")` ```.

Validate your argument values

You can verify that the argument values used to invoke the search macro are acceptable. How to invoke search macros are discussed in the following section, "Apply macros to saved and ad hoc searches".

- **Validation Expression** is a string that is an 'eval' expression that evaluates to a boolean or a string.
- If the validation expression is a boolean expression, validation succeeds when it returns true. If it returns false or is null, validation fails, and the **Validation Error Message** is returned.

If the validation expression is not a boolean expression, it is expected to return a string or NULL. If it returns null, validation is considered a success. Otherwise, the string returned is rendered as the error string.

Apply macros to saved and ad hoc searches

To include a search macro in your saved or ad hoc searches, use the left quote (also known as a grave accent) character; on most English-language keyboards, this character is located on the same key as the tilde (~). You can also reference a search macro within other search macros using this same syntax.

Note: Do NOT use the straight quote character that appears in the same key as the double quote (").

Example - Combine search macros and transactions

Transactions and macro searches are a powerful combination that you can use to simplify your transaction searches and reports. This example demonstrates how you can use search macros to build reports based on a defined transaction.

Here, a search macro, named "makesessions", defines a transaction session from events that share the same clientip value that occurred within 30 minutes of each other:

```
transaction clientip maxpause=30m
```

This search takes pageview events and breaks them into sessions, using the "makesessions" search macro:

```
eventtype=pageview | `makesessions`
```

This search returns a report of the number of pageviews per session for each day:

```
eventtype=pageview | `makesessions` | timechart span=1d sum(eventcount) as
pageviews count as sessions
```

If you wanted to build the same report, but with varying span lengths, just save it as a search macro with an argument for the span length. Let's call this search macro, "pageviews_per_second(1)":

```
eventtype=pageview | `makesessions` | timechart $spanarg$ sum(eventcount) as pageviews count as
```

Now, you can specify a span length when you run this search from the Search app or add it to a saved search:

```
`pageviews_per_second(span=1h)`
```

Capture Knowledge

About capturing knowledge

About capturing knowledge

Once you master the basics of freeform search as described in the "Search and Investigate" chapter, you'll want to take things to a higher level of precision, because the raw data you get from those searches won't always get you to the answers you need.

Leverage Splunk's ability to marry the flexibility of unstructured search with the power of working with structured data. Add knowledge about the events, fields, transactions, and patterns in your data. Discover similar events and group them together with a collective name (an "event type") so you can search on them like you do any other field. Identify transactions that are associated with clusters of events and track them. Group related fields together with tags and aliases. Interactively extract new fields based on event data or external information (such as lookup tables) and add them to your searches.

In this chapter you will:

- Learn how to identify similar or related events and group them together as event types.
- See a list of the default fields automatically extracted by Splunk during the indexing process and see examples of their use in searches.
- Find out how to group fields with related values together through tags and aliases.
- Learn how to interactively extract and add new fields.
- Discover how you can identify event clusters related to transactions and use them to your advantage in searches.
- See a detailed example of interactive field extraction.
- Learn how to create a saved search string and share the results of searches with others.
- Manage both in-process and completed search jobs and review their results.

Use default fields

Use default fields

Fields are searchable name/value pairs in event data. When you search, you're matching search terms against segments of your event data; you can search more precisely by using fields. When you index new data, Splunk automatically recognizes and adds fields from information in your data with name/value pairs, headers, or what is otherwise self-explanatory. Some of these default fields are information about where the data came from, such as host, source, and sourcetype. Fields that begin with an underscore are internal fields.

For more information on using default fields in search commands, see "How search commands work" in this manual. For information on configuring default fields, see "About default fields" in the Getting Data In manual.

Type of field	List of fields	Description

Internal fields	`_raw, _time, _indextime, _cd`	These are fields that contain general information about events in Splunk.
Default fields	`eventtype, host, index, linecount, punct, source, sourcetype, splunk_server, timestamp`	These are fields that contain information about where an event originated, in which index it's located, what type it is, how many lines it contains, and when it occurred. These fields are indexed and added to the Fields menu by default.
Default datetime fields	`date_hour, date_mday, date_minute, date_month, date_second, date_wday, date_year, date_zone`	These are fields that provide additional searchable granularity to event timestamps. **Note:** Only events that have timestamp information in them as generated by their respective systems will have date_* fields. If an event has a date_* 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.

A field may have more than one value; for more information about how to handle such fields and their values, see the "Manipulate and evaluate fields with multiple values" topic in this chapter.

You can extract additional fields using Splunk Web or by using extracting search commands. For more information, see the "Extract and add new fields" topic in this chapter.

You might also want to change the name of a field, or group it with other similar fields. This is easily done with tags or aliases for the fields and field values. For more information, see the "Tag and alias field values" topic in this chapter.

This topic discusses the internal and other default fields that Splunk automatically adds when you index data.

Internal fields

_raw

The `_raw` field contains the original raw data of an event. Splunk's `search` command uses the data in `_raw` when performing searches and data extraction.

You can't use `_raw` as an argument of the `search` command; use `_raw` in data-processing commands only.

Example: Return sendmail events that contain an IP address that starts with "10".

```
eventtype=sendmail | regex _raw=*10.\d\d\d\.\d\d\d\.\d\d\d\*
```

_time

The `_time` field contains an event's timestamp expressed in Unix time. Splunk uses this field to create the event timeline in Splunk Web.

You can only use `_time` in data-processing commands.

Example: Search all sources of type "mail" for mail addressed to the user "strawsky@bigcompany.com", then sorts the search results by timestamp.

```
sourcetype=mail to=strawsky@bigcompany.com | sort _time
```

Other default fields

eventtype

The `eventtype` field contains event types that you (or another user) have defined for an event. Use the `eventtype` field to filter searches; specify event types for your results to match in a `search` argument. Also use `eventtype` to construct data extraction rules, and run reports.

Classify events into an event type by finding similar patterns in your data, and then saving an event type based on the similarities between events.

Note: You can use wildcards to specify multiple event types with a single expression (Example: `eventtype=access*`).

Example 1: Search for events that match any event type that begins with "access".

```
eventtype=access*
```

Example 2: Display the top 10 most common event types of sourcetype "syslog" on `splunk3`.

```
sourcetype="syslog" host=splunk3 | top eventtype
```

host

The `host` field contains the originating hostname or IP address of the network device that generated the event. . Use the `host` field to narrow searches by specifying a `host` value that events must match. You can use wildcards to specify multiple hosts with a single expression (Example: `host=corp*`).

You can use `host` to filter results in data-generating commands, or as an argument in data-processing commands.

Example 1: Search for events on all "corp" servers for accesses by the user "strawsky". It then reports the 20 most recent events.

```
host=corp* eventtype=access user=strawsky | head 20
```

Example 2: Search for events containing the term "404", and are from any host that starts with "192".

```
404 | regex host=*192.\d\d\d\.\d\d\d\.\d\d\d\*
```

index

The_`index` field contains the name of the index in which a given event is indexed. Specify an index to use in your searches by using: `index="name_of_index"`. By default, all events are indexed in the `main` index (`index="main"`).

Example: Search the `myweb` index for events that have the ".php" extension.

```
index="myweb" *.php
```

linecount

The `linecount` field contains the number of lines an event contains. This is the number of lines an event contains before it is indexed. Use `linecount` to search for events that match a certain number of lines, or as an argument in data-processing commands. To specify a matching range, use a greater-than and less-than expression (Example: `linecount>10 linecount<20`).

Example: Search corp1 for events that contain "40" and have 40 lines, and omit events that contain 400.

```
40 linecount=40 host=corp1 NOT 400
```

punct

The `punct` field contains a punctuation pattern that is extracted from an event. The punctuation pattern is unique to types of events. Use `punct` to filter events during a search or as a field argument in data-processing commands.

You can use wildcards in the `punct` field to search for multiple punctuation patterns that share some common characters that you know you want to search for. You must use quotation marks when defining a punctuation pattern in the `punct` field.

Example 1: Search for all punctuation patterns that start and end with **:**

```
punct=":*:"
```

Example 2: Search the php_error.log for php error events that have the punctuation pattern:"[--_::]__:___:____/-..-///.___".

```
source="/var/www/log/php_error.log"
punct="[--_::]__:___:____''/-..-''///.___"
```

source

The `source` field contains the filename or pathname from which the event was indexed. Use `source` to filter events during a search, or as an argument in a data-processing command. You can use wildcards to specify multiple sources with a single expression (Example: `source=*php.log*`).

You can use `source` to filter results in data-generating commands, or as an argument in data-processing commands.

Example: Search for events from the source "/var/www/log/php_error.log".

```
source="/var/www/log/php_error.log"
```

sourcetype

The `sourcetype` field contains a classification, or type, of source. A Splunk administrator can predefine source types, or they can be generated automatically by Splunk at index time. Use `sourcetype` to filter events during a search, or as an argument in a data-processing command. You can use wildcards to specify multiple sources with a single expression (Example: `sourcetype=access*`).

Example: Search for all events that are of the source type "access log".

```
sourcetype=access_log
```

splunk-server

The `splunk-server` field contains the different server names in a distributed Splunk environment.

Example: Restrict a search to the main index on a remote server that is named, remote.

```
splunk-server=remote index=main 404
```

timestamp

The `timestamp` field contains an event's timestamp value (extracted at index time). Splunk extracts timestamps based on how you (or your Splunk admin) has timestamp extraction configured. You can use `timestamp` as a `search` command argument to filter your search.

For example, you can add `timestamp=none` to your search to filter your search results to include only events that have no recognizable timestamp value.

Example: Return the number of events in your data that have no recognizable timestamp.

```
timestamp=none | stats count(_raw) as count
```

Default datetime fields

You can use datetime fields to filter events during a search or as a field argument in data-processing commands.

If you are located in a different timezone from the Splunk server, time-based searches use the timestamp of the event that is applied from the server where it is indexed. The datetime values are the literal values parsed from the event when it is indexed, regardless of its timezone. So, a string such as "05:22:21" will be parsed into indexed fields: "date_hour::5 date_minute::22 date_second::21".

date_hour

The `date_hour` field contains the value of the hour in which an event occurred (range: 0-23). This value is extracted from the event's timestamp (the value in `_time`).

Example: Search for events with the term "apache" that occurred between 10pm and 12am on the current day.

```
apache (date_hour >= 22 AND date_hour <= 24)
```

date_mday

The `date_mday` field contains the value of the day of the month on which an event occurred (range: 1-31). This value is extracted from the event's timestamp (the value in `_time`).

Example: Search for events containing the term "apache" that occurred between the 1st and 15th day of the current month.

```
apache (date_mday >= 1 AND date_mday <= 15)
```

date_minute

The `date_minute` field contains the value of the minute in which an event occurred (range: 0-59). This value is extracted from the event's timestamp (the value in `_time`).

Example: Search for events containing the term "apache" that occurred between the 15th and 20th minute of the current hour.

```
apache (date_minute >= 15 AND date_minute <= 20)
```

date_month

The `date_month` field contains the value of the month in which an event occurred. This value is extracted from the event's timestamp (the value in `_time`).

Example: Search for events with the term "apache" that occurred in January.

```
apache date_month=1
```

date_second

The `date_second` field contains the value of the seconds portion of an event's timestamp (range: 1-59). This value is extracted from the event's timestamp (the value in `_time`).

Example: Search for events containing the term "apache" that occurred between the 1st and 15th second of the current minute.

```
apache (date_second >= 1 AND date_second <= 15)
```

date_wday

The `date_wday` field contains the day of the week on which an event occurred (Sunday, Monday, etc.). Splunk extracts the date from the event's timestamp (the value in `_time`) and determines what day of the week that date translates to. This day of the week value is then placed in the `date_wday` field.

Example: Search for events containing the term "apache" that occurred on Sunday.

```
apache date_wday="sunday"
```

date_year

The `date_year` field contains the value of the year in which an event occurred. This value is extracted from the event's timestamp (the value in `_time`).

Example: Search for events containing the term "apache" that occurred in 2008.

```
apache date_year=2008
```

date_zone

The `date_zone` field contains the value of time for the local timezone of an event, expressed as hours in Unix Time. This value is extracted from the event's timestamp (the value in `_time`). Use `date_zone` to offset an event's timezone by specifying an offset in minutes (range: -720 to 720).

Example: Search for events containing the term "apache" that occurred in the current timezone (local).

```
apache date_zone=local
```

Manipulate and evaluate fields with multiple values

Manipulate and evaluate fields with multiple values

Splunk parses multivalue fields at search time, and allows you to process the values in the search pipeline. Search commands that work with multivalue fields include makemv, mvcombine, mvexpand, and nomv. The eval and where commands support functions, such as `mvcount(), mvfilter(), mvindex(), and mvjoin()` that you can use with multi-valued fields. For more information on these functions see the Functions for eval and where in the *Search Reference manual* and the examples on this page.

You can configure multi-value fields in **fields.conf** to tell Splunk how to recognize more than one field value in a single extracted field value. Edit fields.conf in `$SPLUNK_HOME/etc/system/local/`, or your own custom application directory in `$SPLUNK_HOME/etc/apps/`. For more information on how to do this, see "Configure multivalue fields" in the *Knowledge Manager manual*.

Manipulate multivalued fields

Use nomv to convert a multivalue field into a single value

You can use the nomv command to convert values of the specified multivalued field into one single value. The nomv command overrides multivalue field configurations set in fields.conf.

In this example for sendmail events, you want to combine the values of the senders field into a single value.

```
eventtype="sendmail" | nomv senders
```

Use makemv to separate a multivalue field

You can use the **makemv** command to separate multivalue fields into multiple single value fields. In this example for sendmail search results, you want to separate the values of the "senders" field into multiple field values.

```
eventtype="sendmail" | makemv delim="," senders
```

After you separate the field values, you can pipe it through other commands. For example, you can display the top senders.

```
eventtype="sendmail" | makemv delim="," senders | top senders
```

Use mvexpand to create multiple events based on a multivalue field

You can use the mvexpand command to expand the values of a multivalue field into separate events for each value of the multivalue field. In this example, Splunk creates new events for each value of multivalue field, "foo".

```
... | mvexpand foo
```

Use mvcombine to create a multivalue field from similar events

Combine the values of "foo" with ":" delimiter.

```
... | mvcombine delim=":" foo
```

Evaluate multivalued fields

One of the more common examples of multivalue fields is that of email address fields, which typically appears two to three times in a single sendmail event--once for the sender, another time for the list of recipients, and possibly a third time for the list of Cc addresses, if one exists.

Count the number of values in a field

Use the `mvcount()` function to count the number of values in a single-valued or multivalued field.

In this example, mvcount() returns the number of email addresses in the To, From, and Cc fields and saves them in the specified "_count" fields.

```
eventtype="sendmail" | eval To_count=mvcount(to) | eval
From_count=mvcount(from) | eval Cc_count=mvcount(cc)
```

Note: If only a single email address to exists in the sender field (as you would expect), mvcount(from) returns 1. Also, if there is no Cc address included, the Cc field might not exist for the event and mvcount(cc) returns NULL.

Filter values from a multivalued field

Use the `mvfilter()` function to filter a multivalued field using an arbitrary Boolean expression.

In this example, mvfilter() keeps all values of the field `email` that end in .net or .org:

```
eventtype="sendmail" | eval email=mvfilter(match(email, "\.net$") OR
match(email, "\.org$"))
```

Important: This function works with ONLY ONE field at a time.

Note: This example also uses the `match()` function to compare the pattern defined in quotes to the value of `email`. For more information, see Functions for eval and where in the *Search Reference manual*.

Return a subset of values from a multivalued field

Use the `mvindex()` function to reference a specific value or a subset of values in a multivalued field. Since the index numbering starts at 0, if you want to reference the 3rd value of a field, you would specify it as 2.

In this example, mvindex() returns the first email address in the "To" field for every email sent by Sender:

```
eventtype="sendmail" from=Sender@* | eval to_first=mvindex(to,0)
```

If you wanted to see the top 3 email addresses that Sender writes to each time:

```
eventtype="sendmail" from=Sender@* | eval top_three=mvindex(to,0,2)
```

Note: In this case, `top_three` is, itself, a multivalued field.

Classify and group similar events

Classify and group similar events

An event is not the same thing as an event type. An event is a single instance of data — a single log entry, for example. An event type is a classification used to label and group events.

The names of the matching event types for an event are set on the event, in a multi-valued field called `eventtype`. You can search for these groups of events (for example, SSH logins) the same way you search for any field value.

This topic discusses how to save event types and use them in searches. For more information about events, how Splunk recognizes them, and what it does when it processes them for indexing, see the "Overview of event processing" topic in the Getting Data In manual.

Important: You cannot save a search pipeline as an event type; that is, when saving a search as an event type, it cannot include a search command.

Save a search as a new event type

When you search your event data, you're essentially weeding out all unwanted events. Therefore, the results of your search are events that share common characteristics, and you can give them a collective name.

For example, if you often search for failed logins on different host machines, you can save an eventtype for the events and call it `failed_login`:

```
"failed login" OR "FAILED LOGIN" OR "Authentication failure" OR "Failed to
authenticate user"
```

To save this search as an eventtype:

1. Select "Save as event type..." from the search **Actions** dropdown menu.

If necessary, you can modify the search string.

2. In the **Save Eventtype** window, give your search a name in the "Name" text area.

For our search example, we'll name it "failed_login".

3. Click "Save" to save your event type name.

Note: You can also define tags for the event type; this is discussed in more detail later.

Now, you can quickly search for all the events that match this `eventtype` the same way you can search for any field.

For example, you may be interested it in finding failed logins on specific host machines:

```
host=target eventtype=failed_login
```

Or you may want to investigate a suspicious user's activities:

```
user=suspicious eventtype=failed_login
```

For more information about searching for fields, see the "Start searching" topic in the Search and Investigate chapter of this manual.

Identify similar events with punct

Because the punctuation of an event is often unique to a specific type of event, Splunk indexes the punctuation characters of event in the `punct` field. The values of this field may look cryptic, but they can be an effective way of characterizing similar events.

To apply the `punct` field to your search results, use the Fields popup discussed in the "Search interactively with Splunk Web" topic in the Search and Investigate chapter of this manual. Select the `punct` value for an SSH login event. This updates your search to include this `punct` combination in the search bar. You may want to consider wildcarding the punctuation to match insignificant variations (for example, "punct=::[]*/*").

Use typelearner to discover new event types

Pass any of your searches into the `typelearner` command to see Splunk's suggestions for event types. By default, `typelearner` compares the punctuation of the events resulting from the search, grouping those that have similar punctuation and terms together.

You can specify a different field for Splunk to group the events; `typelearner` works the same way with any field. The result is a set of events (from your search results) that have this field and phrases in common.

For more information and examples, see "typelearner" in the search command reference.

Use tags to group and find similar events

Event types can have one or more tags associated with them. You can add these tags while you save a search as an event type and from the event type manager, located in **Manager > Event types**. From the list of event types in this window, select the one you want to edit.

After you add tags to your event types, you can search for them in the same way you search for any tag. Let's say you saved a search for firewall events as the event type `firewall_allowed`, and then saved a search for login events as the event type `login_successful`. If you tagged both of these event types with *allow*, all events of either of those event types can be retrieved by using the search:

```
tag::eventtype="allow"
```

For more information about using tags, see the "Tag and alias field values" topic in this chapter.

Tag and alias field values

Tag and alias field values

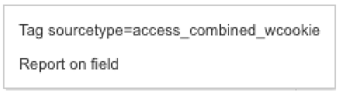
In your data, you might have groups of events with related field values. To help you search more efficiently for these groups of fields, you can assign tags to their field values. You can assign one or more tags to any extracted field (including event type, host, source, or source type).

For more information, read "About tags and aliases" in the Knowledge Manager manual.

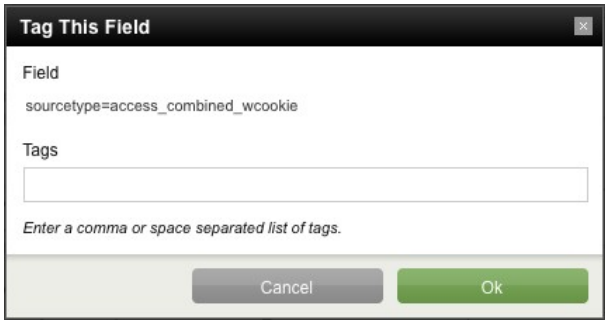
How to tag and alias field values

Tag field value pairs

You can use Splunk Web to tag any field value pair directly from the search results. In any resulting event that has the field value pair that you want to tag, click on the arrow next to that field value. A dropdown menu opens with an option to tag that value. For example, if you selected a syslog source type, you will see:



After you select the **Tag** action for your field value pair, you can add the tag or tags in the "Tag this field" popup window:


Tag This Field
Field
sourcetype=access_combined_wcookie
Tags
Enter a comma or space separated list of tags.
Cancel
Ok

Alias field names

You can add multiple aliases to a field name or use these field aliases to normalize different field names. This does not rename or remove the original field name. After you alias a field, you can search for it using any of its name aliases. To alias a field name, you need to have access to props.conf. For information on how to do this, see "Create aliases for fields" in the Knowledge Manager manual.

Search for tagged field values

There are two ways to search for tags. If you are searching for a tag associated with a value on any field, you can use the following syntax:

```
tag=<tagname>
```

Or, if you are looking for a tag associated with a value on a specific field, you can use the following syntax:

```
tag::<field>=<tagname>
```

Use wildcards to search for tags

You can use the asterisk (`*`) wildcard when searching keywords and field values, including for eventtypes and tags.

For example, if you have multiple event-type tags for varous types of IP addresses, such as `IP-src` and `IP-dst`, you can search for all of them with:

```
tag::eventtype=IP-*
```

If you wanted to find all hosts whose tags contain "local", you can search for the tag:

```
tag::host=*local*
```

Also, if you wanted to search for the events with eventtypes that have no tags, you can search for the Boolean expression:

```
NOT tag::eventtype=*
```

Disabling and deleting tags

If you have a tag that you no longer want to use, or want to have associated with a particular field, you have the option of either disabling it or removing it. You can:

- Remove a tag association for a specific field value through the Search app.
- Disable or delete tags, even if they are associated with multiple field values, via Splunk Manager.

For more information about using Splunk Manager to manage tags, see "Define and manage tags" in the Knowledge Manager manual.

Remove a tag association for a specific field value in search results

If you no longer want to have a tag associated with a specific field value in your search results, click the arrow next to that field/value combination to open up the dropdown menu. Select *Tag [fieldname]=[value]* to bring up the **Tag This Field** popup window.

Erase the tag or tags that you want to disable from the **Tags** field and click **Save**. This removes this particular tag and field value association from the system. If this is the only field value with which a tag is associated, then the tag is removed from the system.

Rename source types

When you configure a source type in props.conf, you can rename the source type. Multiple source types can share the same name; this can be useful if you want to group a set of source types together for searching purposes. For example, you can normalize source type names that include "-too_small" to remove the classifier. For information on how to do this, see "Rename source types" in the Admin manual.

Extract and add new fields

Extract and add new fields

As you learn more about your data, you may find more information that you want to use. There are a variety of ways to extract this information from your events, save it as a field, and use it to search and build reports. You can also look up information from external sources (such as a CSV file or the output of a script) and add it to your event data.

In this topic, you will:

- Learn how to extract and save fields interactively in Splunk Web.
- Learn about the search commands that extract fields from your events.
- Learn about using configuration files to define field extraction at index-time.
- Learn about matching fields with lookup tables to add new fields to your events.

Extract fields interactively in Splunk Web

You can create custom fields dynamically using the interactive field extraction (IFX) feature of Splunk Web. To access the IFX, run a search and then select "Extract fields" from the dropdown that appears left of the timestamps in the search results. The IFX enables you to extract one field at a time, based on a host, source, or source-type value. For more information, see the Interactive field extraction example in this manual.

Extract fields with search commands

You can use a variety of search commands to extract fields in different ways. Here is a list of those commands; for examples of how to use each of these commands, see "Extract fields with search commands" in this manual.

- rex performs field extractions using Perl regular expressions named groups.
- extract (or `kv`, for "key/value") explicitly extracts field/values using default patterns.
- multikv extracts field/values on multi-line, tabular-formatted events.
- xmlkv extracts field/values on xml-formatted event data.
- kvform extracts field/values based on predefined form templates.

Define field extraction in conf files

All field extraction rules that you add using IFX get written to the configuration files. You can also edit these files directly, if you have the permissions to access them. For more information see "Add fields at search time through configuration file edits" in the Knowledge Manager manual.

Look up fields from external data sources

You can match fields in your events to fields in external sources, such as lookup tables, and use these matches to add more information inline to your events.

A lookup table can be a static CSV file or the output of a Python script. You can also use the results of a search to populate the CSV file and then set that up as a lookup table. For more information about field lookups, see "Add fields from external data sources" in the Knowledge Manager manual.

After you configure a fields lookup, you can invoke it from the Search app with the `lookup` command.

Example: Given a field lookup named `dnslookup`, referencing a Python script that performs a DNS and reverse DNS lookup and accepts either a host name or IP address as arguments -- you can use the lookup command to match the host name values in your events to the host name values in the table, and then add the corresponding IP address values to your events.

```
... | lookup dnslookup host OUTPUT ip
```

For a more extensive example using the Splunk script `external_lookup.py`, see "Reverse DNS Lookups for Host Entries" in the Splunk blogs.

Extract fields with search commands

Extract fields with search commands

As mentioned in "Extract and add new fields", you can use a variety of search commands to extract fields in different ways. Continuing reading for examples of usage for the rex, extract, multikv, xmlkv, and kvform commands.

Extract fields using regular expressions

The rex search command performs field extractions using Perl regular expression named groups that you include in the search string. It matches segments of your raw events with the regular expression and saves these values into a field.

In this example, Splunk matches terms that occur after the strings "From:" and "To:" and saves these values into the "from" and "to" fields, respectively.

```
... | rex field=_raw "From: (?<from>.*) To: (?<to>.*)"
```

If a raw event contains "From: Susan To: Bob", then Splunk would extract the field name/value pairs: "from=Susan" and "to=Bob".

For a primer on regular expression syntax and usage, see Regular-Expressions.info. Splunk also maintains a list of useful third-party tools for writing and testing regular expressions.

Force field value extractions on search results

Force field extractions defined in conf files

The extract (or `kv`, for "key/value") search command forces field/value extraction on the result set. If you use `extract` without specifying any arguments, Splunk extracts fields using field extraction stanzas that have been added to props.conf. You can use `extract` to test any field extractions that you add manually through conf files.

Extract fields from events formatted as tables

Use multikv to force field/value extraction on multi-line, tabular-formatted events. It creates a new event for each table row and derives field names from the table title.

Extract fields from events formatted in xml

The xmlkv command enables you to force field/value extraction on xml-formatted tags in event data, such as transactions from web pages.

Extract fields from events based on form templates

The kvform command extracts field/value pairs from events based on form templates that are predefined and stored in `$SPLUNK_HOME/etc/system/local/`, or your own custom application directory in `$SPLUNK_HOME/etc/apps/`. For example, if `form=sales_order`, Splunk would look for a `sales_order.form`, and Splunk would match all processed events against that form, trying to extract values.

Extract fields interactively in Splunk Web

Extract fields interactively in Splunk Web

Use the interactive field extraction (IFX) feature of Splunk Web to create custom fields dynamically on your local Splunk instance. IFX enables you to define any pattern for recognizing one or more fields. IFX is especially useful if you are not familiar with regular expression syntax and usage, because it will generate field extraction regexes for you (and enable you to test them). Also, unless you hand edit the regular expression, IFX learns to extract only one field at a time.

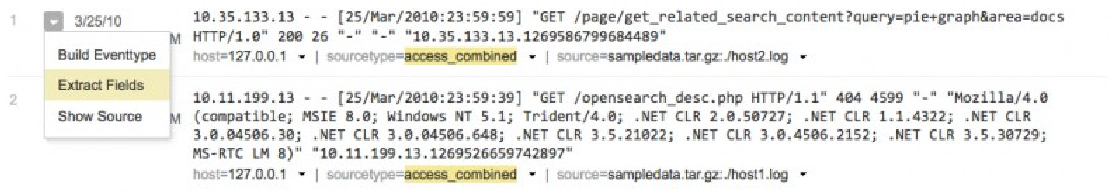
Splunk is very good at handling Web access logs, so there is really no information that it hasn't already extracted for you. For the examples here, let's just walk how to extract the IP addresses from events.

1. To access the IFX, first run a search that generates events containing the field values that you want to extract.

Because we're interested in extracting IP addresses, we can search for Apache access logs with:

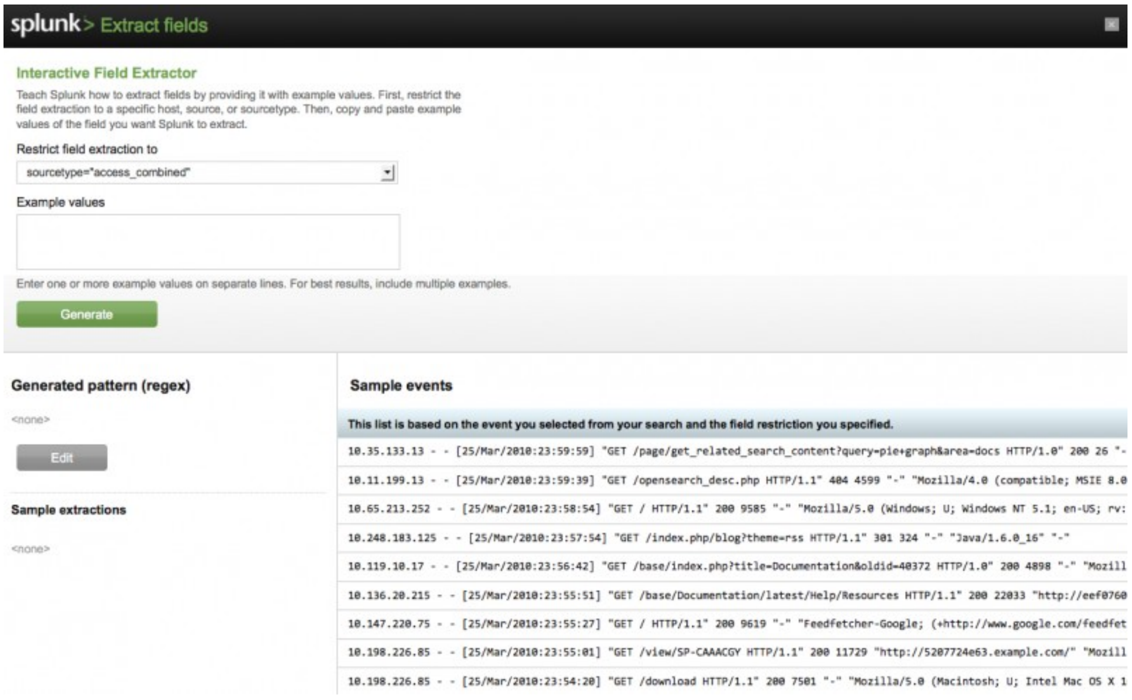
```
sourcetype=access_combined
```

2. From the results of the search, find an event that contains the field value; in this case, an IP address. Click on the arrow to the left of the timestamp of an event and select *Extract fields*.



```
10.35.133.13 - - [25/Mar/2010:23:59:59] "GET /page/get_related_search_content?query=pie+graph&area=docs
HTTP/1.0" 200 26 "-" "-" "10.35.133.13.1269586799684489"
host=127.0.0.1 | sourcetype=access_combined | source=sampledata.tar.gz:./host2.log

10.11.199.13 - - [25/Mar/2010:23:59:39] "GET /opensearch_desc.php HTTP/1.1" 404 4599 "-" "Mozilla/4.0
(compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; .NET CLR 2.0.50727; .NET CLR 1.1.4322; .NET CLR
3.0.04506.30; .NET CLR 3.0.04506.648; .NET CLR 3.5.21022; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729;
MS-RTC LM 8)" "10.11.199.13.1269526659742897"
host=127.0.0.1 | sourcetype=access_combined | source=sampledata.tar.gz:./host1.log
```

The IFX opens in a new window, **Extract fields**.


splunk> Extract fields

Interactive Field Extractor

Teach Splunk how to extract fields by providing it with example values. First, restrict the field extraction to a specific host, source, or sourcetype. Then, copy and paste example values of the field you want Splunk to extract.

Restrict field extraction to: sourcetype="access_combined"

Example values

Enter one or more example values on separate lines. For best results, include multiple examples.

Generated pattern (regex)

<none>

Sample extractions

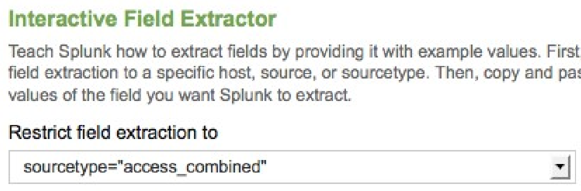
<none>

Sample events

This list is based on the event you selected from your search and the field restriction you specified.

```
10.35.133.13 - - [25/Mar/2010:23:59:59] "GET /page/get_related_search_content?query=pie+graph&area=docs HTTP/1.0" 200 26 "-
10.11.199.13 - - [25/Mar/2010:23:59:39] "GET /opensearch_desc.php HTTP/1.1" 404 4599 "-" "Mozilla/4.0 (compatible; MSIE 8.0
10.65.213.252 - - [25/Mar/2010:23:58:54] "GET / HTTP/1.1" 200 9585 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:
10.248.183.125 - - [25/Mar/2010:23:57:54] "GET /index.php/blog?theme=rss HTTP/1.1" 301 324 "-" "Java/1.6.0_16" "-"
10.119.10.17 - - [25/Mar/2010:23:56:42] "GET /base/index.php?title=Documentation&oldid=40372 HTTP/1.0" 200 4898 "-" "Mozill
10.136.20.215 - - [25/Mar/2010:23:55:51] "GET /base/Documentation/latest/Help/Resources HTTP/1.1" 200 22033 "http://eef0760
10.147.220.75 - - [25/Mar/2010:23:55:27] "GET / HTTP/1.1" 200 9619 "-" "Feedfetcher-Google; (+http://www.google.com/feedfet
10.198.226.85 - - [25/Mar/2010:23:55:01] "GET /view/SP-CAAACGY HTTP/1.1" 200 11729 "http://5207724e63.example.com/" "Mozill
10.198.226.85 - - [25/Mar/2010:23:54:20] "GET /download HTTP/1.1" 200 7501 "-" "Mozilla/5.0 (Macintosh; U; Intel Mac OS X 1
```

3. Choose *sourcetype="access_combined"* from the **Restrict field extraction to** menu.

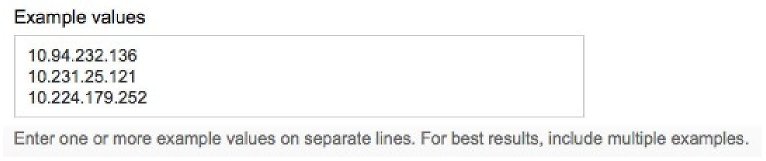

Interactive Field Extractor

Teach Splunk how to extract fields by providing it with example values. First, restrict the field extraction to a specific host, source, or sourcetype. Then, copy and paste example values of the field you want Splunk to extract.

Restrict field extraction to: sourcetype="access_combined"

This dropdown is populated with the field values from the event that you selected in step 2. If you want to choose a different host, source, or sourcetype value, close the window and select a different event in your search results or run a new search.

4. Type, or copy and paste, some values of IP addresses from your results into the **Example values** text box. For best results, give multiple examples.

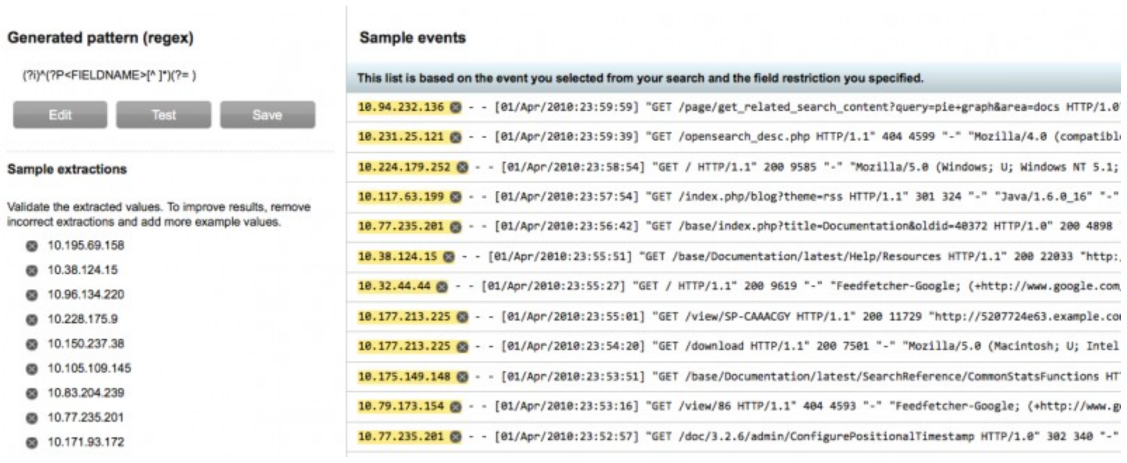

Example values

```
10.94.232.136
10.231.25.121
10.224.179.252
```

Enter one or more example values on separate lines. For best results, include multiple examples.

The list of Sample events is based on the event you selected from your search results and the field restriction you specified. If you change the field restriction, this list also changes; but it will still be based on the original event you selected.

5. When you think you have enough sample values, click **Generate**. Splunk generates a regex pattern for your field, based on the information you gave it and the general format of your events.



Generated pattern (regex)

```
(?i)^(?P<FIELDNAME>[^ ]*)(?= )
```

Sample extractions

Validate the extracted values. To improve results, remove incorrect extractions and add more example values.

```
10.195.69.158
10.38.124.15
10.96.134.220
10.228.175.9
10.150.237.38
10.105.109.145
10.83.204.239
10.77.235.201
10.171.93.172
```

Sample events

This list is based on the event you selected from your search and the field restriction you specified.

```
10.94.232.136 - - [01/Apr/2010:23:59:59] "GET /page/get_related_search_content?query=pie+graph&area=docs HTTP/1.0
10.231.25.121 - - [01/Apr/2010:23:59:39] "GET /opensearch_desc.php HTTP/1.1" 404 4599 "-" "Mozilla/4.0 (compatibl
10.224.179.252 - - [01/Apr/2010:23:58:54] "GET / HTTP/1.1" 200 9585 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1;
10.117.63.199 - - [01/Apr/2010:23:57:54] "GET /index.php/blog?theme=rss HTTP/1.1" 301 324 "-" "Java/1.6.0_16" "-"
10.77.235.201 - - [01/Apr/2010:23:56:42] "GET /base/index.php?title=Documentation&oldid=40372 HTTP/1.0" 200 4898
10.38.124.15 - - [01/Apr/2010:23:55:51] "GET /base/Documentation/latest/Help/Resources HTTP/1.1" 200 22033 "http:
10.32.44.44 - - [01/Apr/2010:23:55:27] "GET / HTTP/1.1" 200 9619 "-" "Feedfetcher-Google; (+http://www.google.com
10.177.213.225 - - [01/Apr/2010:23:55:01] "GET /view/SP-CAAACGY HTTP/1.1" 200 11729 "http://5207724e63.example.co
10.177.213.225 - - [01/Apr/2010:23:54:20] "GET /download HTTP/1.1" 200 7501 "-" "Mozilla/5.0 (Macintosh; U; Intel
10.175.149.148 - - [01/Apr/2010:23:53:51] "GET /base/Documentation/latest/SearchReference/CommonStatsFunctions HT
10.79.173.154 - - [01/Apr/2010:23:53:16] "GET /view/86 HTTP/1.1" 404 4593 "-" "Feedfetcher-Google; (+http://www.g
10.77.235.201 - - [01/Apr/2010:23:52:57] "GET /doc/3.2.6/admin/ConfigurePositionalTimestamp HTTP/1.0" 302 340 "-"
```

6. Review the **Sample extractions** and **Sample events** to see if there are any values that you don't want or any values that you do what that aren't displayed in the list.

If you see unwanted values, click the **X** next to the value to remove it. If you notice a value that is left out, add it to the list of **Sample events**. The IFX will update the generated pattern to reflect this change. To re-add any value and reset the regex, just click the **+** icon next to it.

7. Before you save the new field, you can test the regex against a larger data set or edit the pattern manually.

When you click **Test** from the field extractions page, Splunk takes you to the Search view and runs a search:

- against your host, source, or sourcetype restriction, limited to first 10,000 results.
- using the rex command with the regex Splunk generated for your FIELDNAME, removing any duplicate occurrences of the field value.

If you edit the search expression manually in the test window, you must copy-paste your changes back to the IFX page.

If you're familiar with writing regexes, you can edit the pattern manually after Splunk generates it; just click **Edit** in the IFX window.

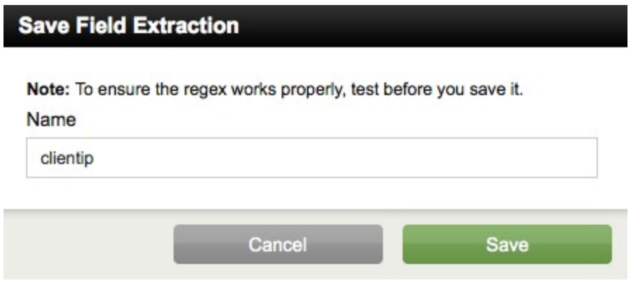
Also, you'll notice that the name of the extracted field in the search or edit window is "FIELDNAME". You do not need to rename this value because it will be set with the name you enter when you save the extraction.

After testing or editing your regex, return to the IFX window.

8. If the expression looks like it's working, click **Save**.

This opens the **Save Field Extraction** window.

9. Name your extraction *clientip* and click **Save**.



Note: To ensure the regex works properly, test before you save it.

clientip

Splunk only accepts field names that contain alpha-numeric characters or an underscore:

- Valid characters for field names are **a-z, A-Z, 0-9,** or _ .
- Field names cannot begin with **0-9** or _ . (Leading underscores are reserved for Splunk's internal variables).

Important: Extractions created by a user will be located in `$SPLUNK_HOME/etc/users` and will be a function of the role a user has, with relationship to the app.

Use field lookups to add information to your events

Use field lookups to add information to your events

Splunk's lookup feature lets you reference fields in an external CSV file that match fields in your event data. Using this match, you can enrich your event data by adding more searchable fields to them. You can base your field lookups on any field, including a temporal field, or on the output of a Python script.

This topic discusses how to use the Lookups manager page, located in Splunk Web, at **Manager > Lookups**, to:

- List existing lookup tables or upload a new file.
- Edit existing lookup definitions or define a new file-based or external lookup.
- Edit existing automatic lookups or configure a new lookup to run automatically.

For more details about lookups, see "Add fields from external data sources" in the Knowledge Manager manual.

List existing lookup tables or upload a new file

View existing lookup table files in **Manager > Lookups > Lookup table files**, or click "New" to upload more CSV files to use in your definitions for file-based lookups.

To upload new files:

1. Select a **Destination app**.

This tells Splunk to save your lookup table file in the app's directory:
`$SPLUNK_HOME/etc/users/<username>/<app_name>/lookups/`.

2. Give your lookup table file a **Name**.

This will be the name you use to refer to the file in a lookup definition.

3. Browse for the CSV file to upload.

4. Click **Save**.

Edit existing lookup definitions or define a new file-based or external lookup

Use the **Manager > Lookups > Lookup definitions** page to define the lookup table or edit existing lookup definitions. You can specify the type of lookup (file-based or external) and whether or not it is time-based. Once you've defined the lookup table, you can invoke the lookup in a search (using the lookup command) or you can configure the lookup to occur automatically.

Note: This is equivalent to defining your lookup in transforms.conf.

Configure a time-based lookup

File-based and external lookups can also be time-based (or temporal), if the field matching depends on time information (a field in the lookup table that represents the timestamp).

To **Configure a time-based lookup**, specify the *Name of the time field*. You can also specify a strptime format for this time information and offsets for the time matching.

Include advanced options

Under **Advanced options**, you can specify:

- The minimum number of matches for each input lookup value.
- The maximum number of matches for each input lookup value.
- A default value to output if fewer than the minimum number of matches are present for a given input.

Edit existing automatic lookups or configure a new lookup to run automatically

Instead of invoking the lookup command when you want to apply a fields lookup to your events, you can set the lookup to run automatically. Use the **Manager > Lookups > Automatic lookups** page to edit or configure automatic lookups:

1. Select the lookup table file that you want use in your fields lookup.

2. Select a host, source, or sourcetype value to apply the lookup.

3. Under lookup input fields, list one or more pairs of lookup field name and local field name.

4. Under lookup output fields, list one or more pairs of lookup field name and local field name.

5. You can also choose to overwrite the field values each time the lookup runs.

Note: This is equivalent to configuring your fields lookup in props.conf.

Example of HTTP status lookup

This examples walks through defining a static lookup that adds two informational fields, `status_description` and `status_type`, into your Web access events. This lets you search for the events you want when you might not know the specific error code. For example, instead of searching for all the server error codes, you can use `status="Server Error"`.

Upload the lookup table to Splunk

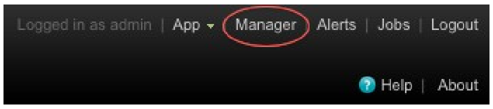
1. Download the `http_status.csv` file:

http_status.csv

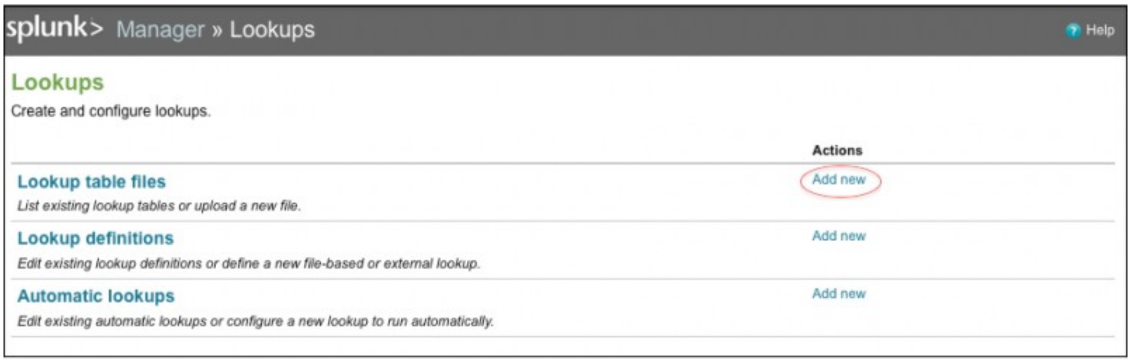
Here's a sampling of the file:

```
status,status_description,status_type
100,Continue,Informational
101,Switching Protocols,Informational
200,OK,Successful
201,Created,Successful
202,Accepted,Successful
203,Non-Authoritative Information,Successful
...
```

2. Go back to the Search app, and select Manager from the navigation menu on the upper right.



3. In the *Manager > Lookups* view, select *Add new* for **Lookup table files**.


splunk> Manager » Lookups

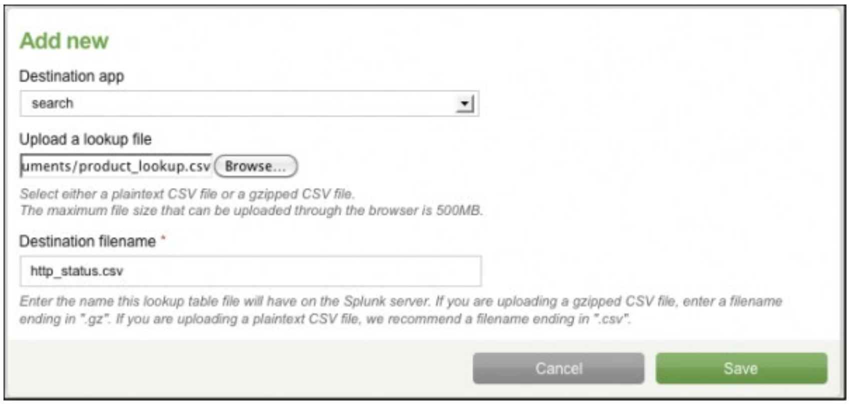
Lookups

Create and configure lookups.

	Actions
Lookup table files *List existing lookup tables or upload a new file.*	Add new
Lookup definitions *Edit existing lookup definitions or define a new file-based or external lookup.*	Add new
Automatic lookups *Edit existing automatic lookups or configure a new lookup to run automatically.*	Add new

4. In **Manager > Lookups > Lookup table files > Add new**,

- Select *search* for the destination app.
- Browse for the CSV file that you downloaded earlier.
- Name the lookup table *http_status*.
- Click *Save*.


Add new

Destination app: search

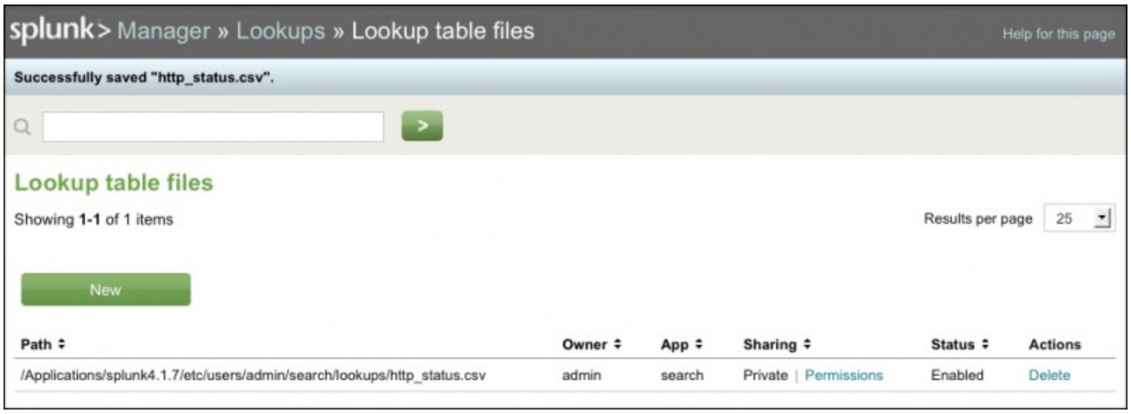
Upload a lookup file: uments/product_lookup.csv

Select either a plaintext CSV file or a gzipped CSV file.
The maximum file size that can be uploaded through the browser is 500MB.

Destination filename *: http_status.csv

Enter the name this lookup table file will have on the Splunk server. If you are uploading a gzipped CSV file, enter a filename ending in ".gz". If you are uploading a plaintext CSV file, we recommend a filename ending in ".csv".

After Splunk saves the file, it takes you to the following view:


splunk> Manager » Lookups » Lookup table files

Successfully saved "http_status.csv".

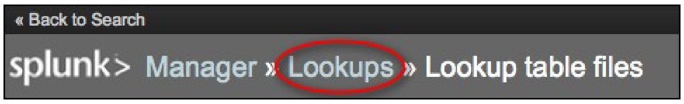
Lookup table files

Showing **1-1** of 1 items

Results per page 25

Path	Owner	App	Sharing	Status	Actions
/Applications/splunk4.1.7/etc/users/admin/search/lookups/http_status.csv	admin	search	Private | Permissions	Enabled	Delete

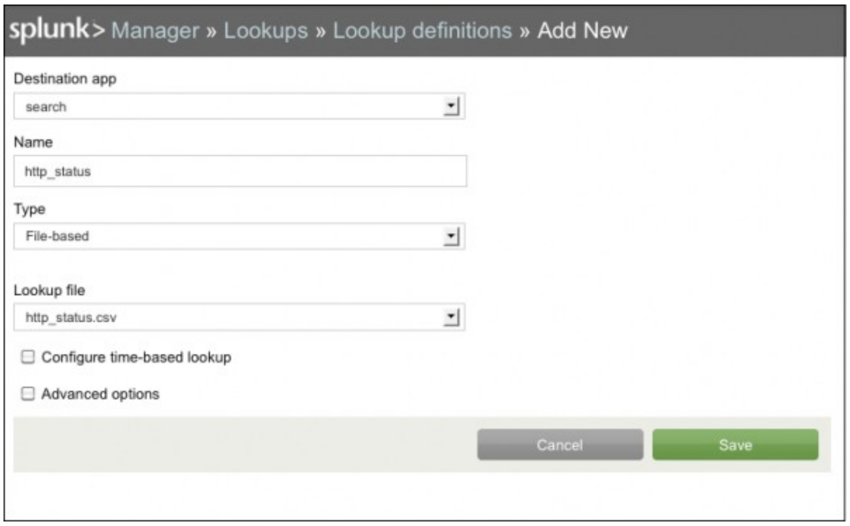
Now, let's go back to the **Manager > Lookups** view. To do this, click on the **Lookups** link in the page's breadcrumb. You can always use this to navigate back to a previous view.



Define the lookup

1. From *Manager > Lookups*, select *Add new* for **Lookup definitions**.

In the *Manager > Lookups > Lookup definitions > Add new* view:


splunk> Manager » Lookups » Lookup definitions » Add New

Destination app: search

Name: http_status

Type: File-based

Lookup file: http_status.csv

☐ Configure time-based lookup

☐ Advanced options

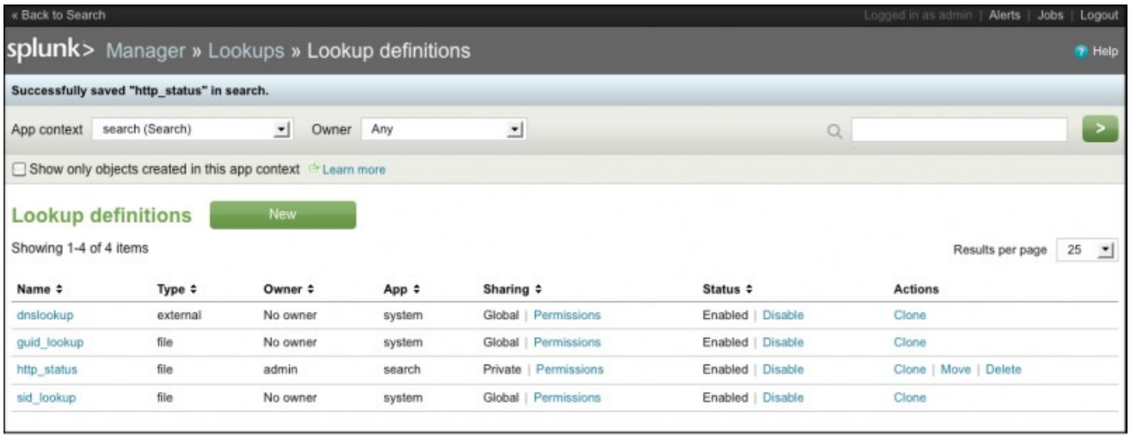
2. Select *search* for the **Destination app**.

3. Name your lookup definition *http_status*.

4. Select *File-based* under **Type**.

5. Click *Save*.

After Splunk saves your lookup definition, it takes you to the following view:


« Back to Search — Logged in as admin | Alerts | Jobs | Logout

splunk> Manager » Lookups » Lookup definitions — Help

Successfully saved "http_status" in search.

App context: search (Search) — Owner: Any

☐ Show only objects created in this app context — Learn more

Lookup definitions — New

Showing 1-4 of 4 items — Results per page: 25

Name	Type	Owner	App	Sharing	Status	Actions
dnslookup	external	No owner	system	Global | Permissions	Enabled | Disable	Clone
guid_lookup	file	No owner	system	Global | Permissions	Enabled | Disable	Clone
http_status	file	admin	search	Private | Permissions	Enabled | Disable	Clone | Move | Delete
sid_lookup	file	No owner	system	Global | Permissions	Enabled | Disable	Clone

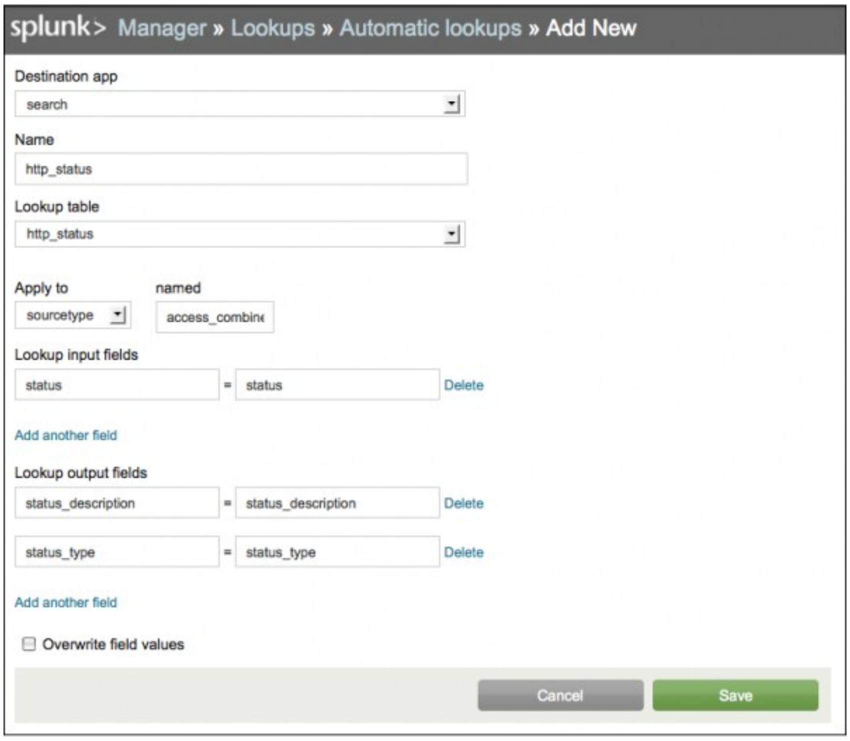
Notice there are some actions you can take on your lookup definition. **Permissions** lets you change the accessibility of the lookup table. You can **Disable**, **Clone**, and **Move** the lookup definition to a different app. Or, you can **Delete** the definition.

Once you define the lookup, you can use the `lookup` command to invoke it in a search or you can configure the lookup to run automatically.

Set the lookup to run automatically

1. Return to the *Manager > Lookups* view and select *Add new* for **Automatic lookups**.

In the *Manager > Lookups > Automatic lookups* view:


splunk> Manager » Lookups » Automatic lookups » Add New

Destination app: search

Name: http_status

Lookup table: http_status

Apply to: sourcetype — named: access_combine

Lookup input fields: status = status — Delete

Add another field

Lookup output fields:
status_description = status_description — Delete
status_type = status_type — Delete

Add another field

☐ Overwrite field values

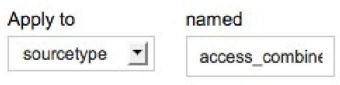
Cancel — Save

2. Select *search* for the **Destination app**.

3. Name the lookup *http_status*.

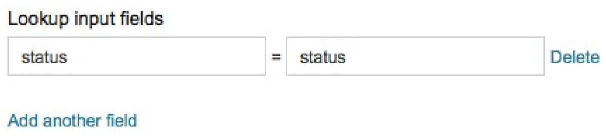
4. Select *http_status* from the **Lookup table** drop down.

5. Apply the lookup to the `sourcetype` named `access_combined`.



Apply to	named
sourcetype	access_combine

6. *Lookup input fields* are the fields in our events that you want to match with the lookup table. Here, both are named `status` (the CSV column name goes on the left and the field that you want to match goes on the right):

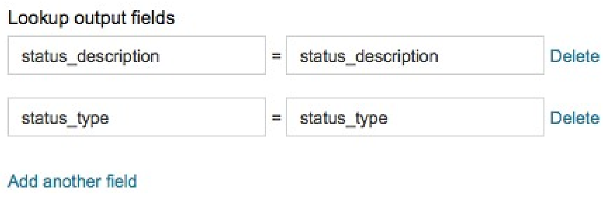


Lookup input fields

status	=	status	Delete

Add another field

7. *Lookup output fields* are the fields from the lookup table that you want to add to your events: status_description and status_type. The CSV column name goes on the left and the field that you want to match goes on the right.

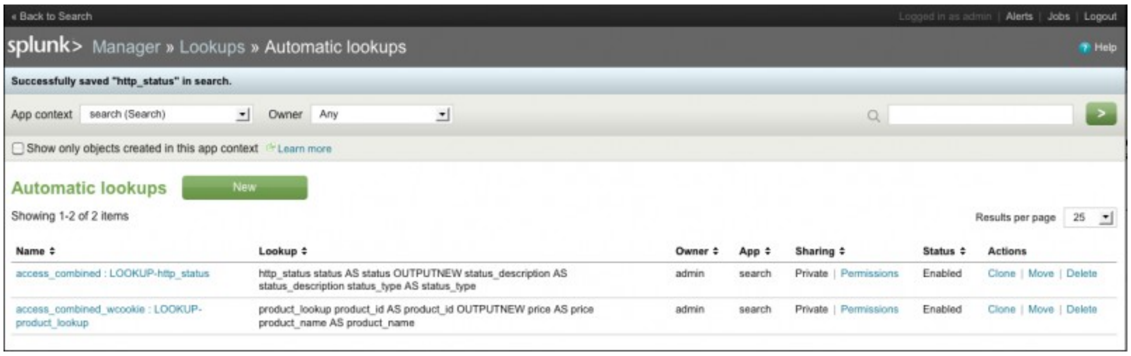


Lookup output fields

status_description	=	status_description	Delete
status_type	=	status_type	Delete

Add another field

8. Click *Save*.



« Back to Search   Logged in as admin | Alerts | Jobs | Logout

splunk> Manager » Lookups » Automatic lookups   Help

Successfully saved "http_status" in search.

App context: search (Search)   Owner: Any

☐ Show only objects created in this app context   Learn more

Automatic lookups   New

Showing 1-2 of 2 items   Results per page 25

Name	Lookup	Owner	App	Sharing	Status	Actions
access_combined : LOOKUP-http_status	http_status status AS status OUTPUTNEW status_description AS status_description status_type AS status_type	admin	search	Private | Permissions	Enabled	Clone | Move | Delete
access_combined_wcookie : LOOKUP-product_lookup	product_lookup product_id AS product_id OUTPUTNEW price AS price product_name AS product_name	admin	search	Private | Permissions	Enabled	Clone | Move | Delete

Identify transactions

Identify transactions

A **transaction** is a meta-event, a collection of events that you want to group together. Transactions can span multiple sources.

A **transaction type** is a configured type of transaction that is saved as a field in Splunk.

A common **transaction search** use is to group multiple events into a single meta-event that represents a single physical event. For example, an **out of memory problem** could trigger several database events to be logged, and they can all be grouped together into a transaction. Use the transaction command to define a transaction or override transaction options specified in transactiontypes.conf.

Example: Run a search that groups together all of the web pages a single user (or client IP address) looked at, over a time range.

This search takes events from the access logs, and creates a transaction from events that share the same `clientip` value that occurred within 5 minutes of each other (within a 3 hour time span).

```
sourcetype=access_combined | transaction fields=clientip maxpause=5m
maxspan=3h
```

For more information, including use cases and examples, see the "Group events into transactions" chapter of the Knowledge Manager manual.

Save searches and share search results

Save searches and share search results

After you run a search that returns interesting or useful results, you may want to **save the search,** so the search can easily be run again without having to retype the search string, among other things. Or you may want to **save the results of this particular search job,** so you and others can review the results of this search run at a later time. This topic covers both post-search actions.

Why save your searches?

It's important to understand why you might save your Splunk searches, because there are reasons for doing so that go beyond the basic use case of enabling at-will rerun of especially useful searches (when you select the saved search from the Searches & Reports dropdown).

You also save searches when you want to:

- **Make the search available for others to run.** When you save a search you can adjust its **permissions** so that it can be accessed and run by other users.
- **Set up an alert.** An **alert** is a saved search that runs automatically on a regular **schedule**. When the scheduled search runs and a specific condition is met by the search results, an alert action is triggered.

- **Add the search to a dashboard.** Saved searches form the basis for charts, tables, and event listings in **dashboard** panels.
- **Set up a summary index. Summary indexes**, which are used to enable fast searches through especially large data sets, are based on saved searches.

For more information about these uses for saved searches see the subtopic "Using saved searches," below.

How to save a search

If you've just designed a search that returns useful results and you want to save it for any of the reasons listed above, it's pretty easy to do so. There are a number of ways you can save the search through the Splunk Web UI once a search is running, finalized, or completed. You can also use Manager or `savedsearches.conf` to add a new saved search to Splunk. And you can define a saved search manually in `savedsearches.conf`. See the following subsections for details on these methods.

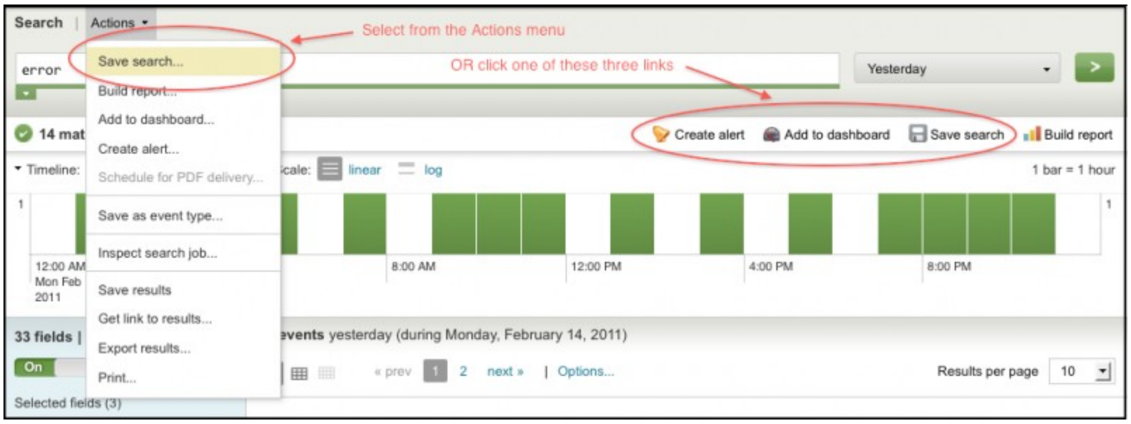
At minimum, a saved search definition includes the search string and the time range associated with the search (expressed in terms of relative time modifiers). It should also include a search name--this is what appears in the **Searches & Reports** dropdown after the search is saved.

Note: You can change the navigation rules for your app so that searches are saved to a location in the top-level navigation other than **Searches & Reports**. For more information, see "Managing saved search navigation", below.

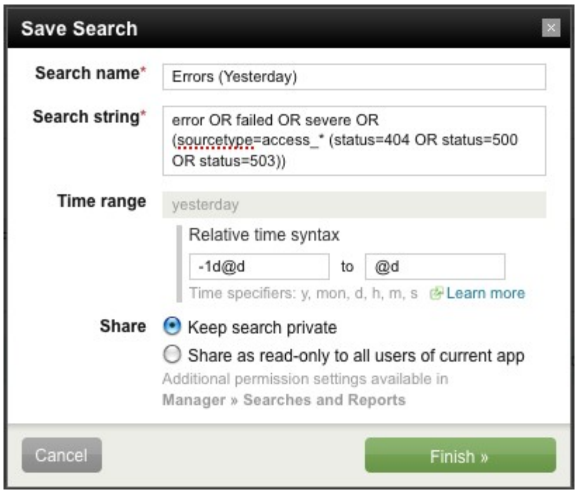
Save a running, completed, or finalized search

When you run a search in the Search view, you can:

- Click the **Save search** link that appears above the search job status bar to open the **Save search** dialog box.
- Click **Add to dashboard** or **Create alert**. These open dialog boxes that enable you to save the search before proceeding to either add the search to a dashboard (in the form of a dashboard **panel**) or creating an alert based on the search.
- Select *Save search* from the **Actions** menu. This also opens the **Save search** dialog box.



In all of these cases, the dialog boxes that open populate with the search string and time range (expressed with **relative time modifiers**) filled out. You can modify that information before you save it. You can also give the search a name. This name will appear in the "Searches & Reports" list after you save the search.


Save Search

Search name* Errors (Yesterday)

Search string* error OR failed OR severe OR (sourcetype=access_* (status=404 OR status=500 OR status=503))

Time range yesterday

Relative time syntax

-1d@d to @d

Time specifiers: y, mon, d, h, m, s Learn more

Share [✓ Keep search private] Share as read-only to all users of current app

Additional permission settings available in Manager » Searches and Reports

By default the saved search will be private and only available to you, but if your permissions allow it, the "Save search" dialog box enables you to reset the search permissions so that every user in the app you're in has "read" access to it, which means that they can run the search from the "Searches and Reports" list but can't edit it. For more information, see the "Share saved searches with other users" subsection, below.

Add a new saved search in Manager

When you are saving a new search, it's easiest to just run the search and then use the "Save search" dialog box to save it. This method enables you to test the search before you save it.

But if you're in Manager, you can also open the "Searches and reports" page there and click **New** to define and add a new saved search. This method requires that you manually add the search string and time range (using **relative time modifiers**) as well as the search name.

When you define a new saved search in Manager you can optionally define a schedule for it, set it up as an alert, and identify alert actions. Manager is also the only place where you set up a saved search to populate a **summary index**.

You can edit and update searches listed on the **Searches and reports** page if you have "write" permissions for them. For more information about permissions, see "Curate Splunk knowledge with Manager" in the Knowledge Manager manual.

Manually define a saved search in savedsearches.conf

Finally, you can manually define a saved search by creating a new saved search stanza in `savedsearches.conf`. It's easier to create saved searches using the **Splunk Web**-based approaches described above--the UI validates your changes, and you don't have to reboot the system to apply searches created through the web interface--but if you prefer to work with saved searches through configuration files, you certainly can.

For more information about configuring saved searches and alerts in `savedsearches.conf`, see the spec file for `savedsearches.conf` and the "Set up alerts in savedsearches.conf" topic in the

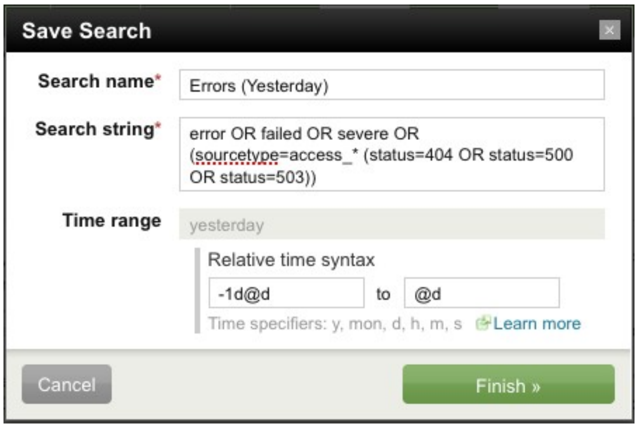
Admin Manual.

Use your saved searches

The most common reason to save a search is so you (and others, if you wish) can rerun it without having to recall and type in the search string each time. When you run a saved search, Splunk creates a new search job using the search string and time range in its definition. But, as mentioned at the top of this topic, there are other reasons for saving searches. The following subsections provide more detail about those saved search uses, with links to relevant topics for more information.

Share saved searches with other users

The search is private by default, meaning that it can only be seen and used by you. However, the "Save search" dialog box also gives you the ability to adjust the search permissions so that every user of the app you're currently in has read-only access to it. "Read-only access" means that users of the app can run the search from the **Searches & Reports** list, but they can't edit it or change its permissions.


Save Search

Search name*: Errors (Yesterday)

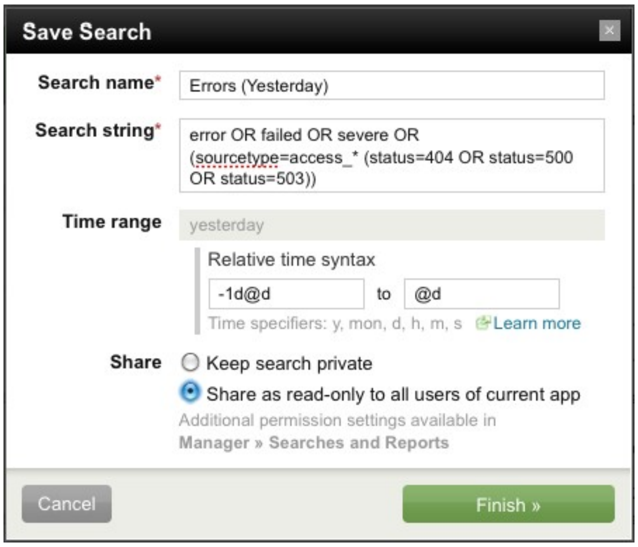
Search string*: error OR failed OR severe OR (sourcetype=access_* (status=404 OR status=500 OR status=503))

Time range: yesterday

Relative time syntax: -1d@d to @d

Time specifiers: y, mon, d, h, m, s Learn more

As mentioned above in "How to save a search", when you save a search with the "Save search" dialog box, the search is initially only available to you. If your permissions allow it, the "Save search" dialog box provides the option of giving all of the users of the app that you're currently in "read" access to the search. This means that they can run the search from the **Searches & Reports** list, but they can't edit it or change its permissions.


Save Search

Search name* Errors (Yesterday)

Search string* error OR failed OR severe OR (sourcetype=access_* (status=404 OR status=500 OR status=503))

Time range yesterday

Relative time syntax -1d@d to @d

Time specifiers: y, mon, d, h, m, s Learn more

Share ( ) Keep search private [✓ Share as read-only to all users of current app]

Additional permission settings available in Manager » Searches and Reports

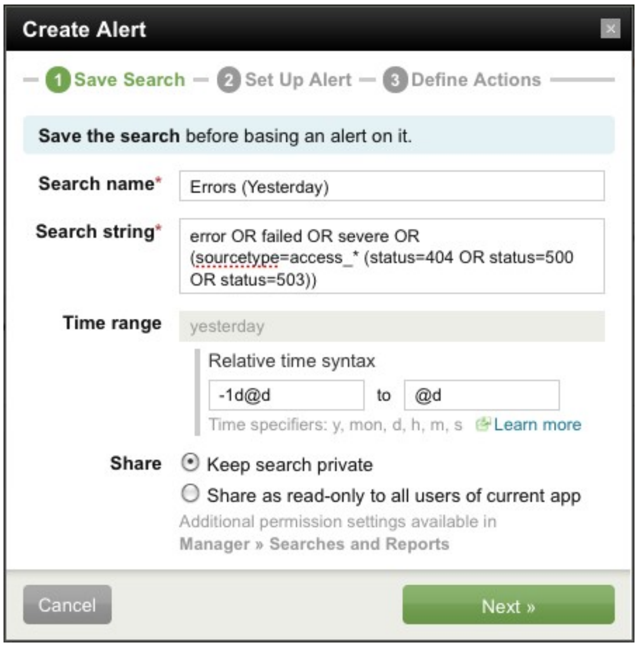
However, if you have an Admin-level role, you can go into **Manager > Searches and reports** and narrow or widen the potential usage of the saved search by further redefining its permissions. For example, you could make it "globally" available to everyone that uses your Splunk implementation. Or you could narrow the saved search permissions so that only specific roles within the current app can use it. You can also arrange for particular roles or users to have "write" access to the saved search, enabling them to edit its definition.

To learn more about managing saved search permissions, see "Curate Splunk knowledge with Manager" in the Knowledge Manager manual.

Schedule alerts and create alert actions

When you run a search, you can click **Create alert** to quickly and efficiently create an alert that is based on the search, whether the search is running in real time, or is collecting information for a specified time range.

After you run a search, click **Create alert** to open the **Create Alert** dialog box. **Create Alert** is broken into three steps. In the first step you can verify or update the basic saved search information (search name, string, time range, permissions). The next two steps of **Create Alert** enable you to define all of the characteristics of a Splunk alert, including the alert condition, schedule, throttling, expiration period (how long the alerts stay in the system before they are deleted), severity and the alert actions that are triggered when the alert condition is met. When you complete the **Create Alert** dialog box, the search should be saved, scheduled, and set up to trigger an alert action when its scheduled runs meet a specified alerting condition.


Create Alert

1 Save Search — 2 Set Up Alert — 3 Define Actions

Save the search before basing an alert on it.

Search name* Errors (Yesterday)

Search string* error OR failed OR severe OR (sourcetype=access_* (status=404 OR status=500 OR status=503))

Time range yesterday

Relative time syntax

-1d@d to @d

Time specifiers: y, mon, d, h, m, s Learn more

Share ● Keep search private

○ Share as read-only to all users of current app

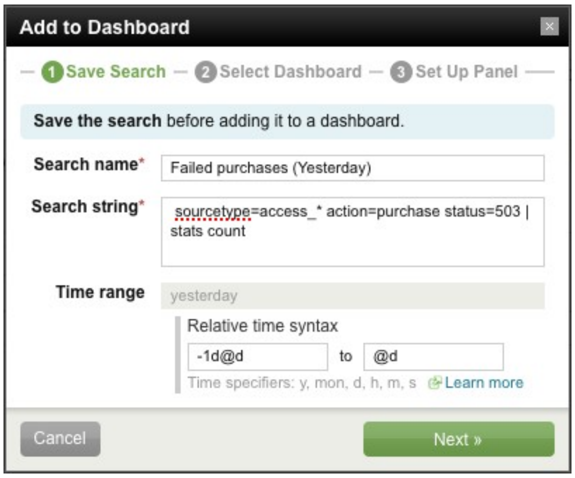
Additional permission settings available in Manager » Searches and Reports

For detailed information about using the **Create Alert** dialog box to create alerts that are based on your searches, see "Create an alert" in this manual.

Add a search to a dashboard

When you run a search, you can click **Add to dashboard** to quickly and efficiently create a dashboard panel that is based on the search and add that panel to a new or existing dashboard.

After you run a search, click **Add to dashboard** to open the **Add to dashboard** dialog box. **Add to dashboard** is broken into three steps. The first step enables you to verify, enter, or update your basic search information. The second and third steps enable you to select the dashboard where you want the saved search to be used (it can be a preexisting dashboard, or a brand new one) and identify the type of panel that will be based on it (for example, you would select a *list* panel type if you want the search to generate a list of events on the dashboard). You can also provide a name for the panel.


Add to Dashboard

1 Save Search — 2 Select Dashboard — 3 Set Up Panel

Save the search before adding it to a dashboard.

Search name* Failed purchases (Yesterday)

Search string* sourcetype=access_* action=purchase status=503 | stats count

Time range yesterday

Relative time syntax

-1d@d to @d

Time specifiers: y, mon, d, h, m, s Learn more

Note: In the **Add to dashboard** dialog box, saved search permissions are managed at the dashboard level (step 2 of the dialog box).

- If the dashboard panel you are creating is going on an existing dashboard, the search you are associating with it takes on the same permissions as that dashboard.
- If the dashboard panel that you are creating is going on a brand new dashboard, and you have admin-level permissions, you can choose to keep the dashboard private, or share the dashboard as read-only with all users of the current app. (If you do not have admin-level permissions the new dashboard will be private--viewable only by you--by default. The search you are associating with the dashboard panel will take on the permissions of the new dashboard.

For detailed information about using the **Add to dashboard** dialog box to create new dashboard panels that are based on your searches, see "Create simple dashboards" in this manual.

Set up a summary index

You use summary indexes to efficiently run reports on especially large data volumes. To do this you arrange for a saved search to run on a regular, frequent interval, with a time range that precisely fits that interval. Each time it runs it extracts precisely the information you want from the overall dataset, and then saves the results into a summary index that you designate. You can then run searches and reports on this significantly smaller (and thus seemingly "faster") summary index.

You arrange for a saved search to be used for summary indexing in the **Searches and reports** page in Manager, when you add a saved search (with the **Add** button) or when you edit an existing saved search. Summary indexing is set up as an alert action that happens each time the search runs on its schedule.

Note: Summary index searches use specific summary indexing reporting commands. Please read "Use summary indexing for increased reporting efficiency" in the Knowledge Manager manual before attempting to set up your own summary index.

Save charts and reports

It's important to note that saved searches do ***not*** include chart formatting parameters. If your search includes reporting commands, and you want the chart that the search produces to include custom formatting (so that it displays a pie chart rather than the default bar chart and has specific text for the title, x-axis, and y-axis, for example) **be sure to save it as a report from the Report Builder.** If you save it as a search, any formatting you set up for the chart in the report builder will be lost. This is especially important if you intend to display the chart in a specific way on a dashboard.

For more information, see "Save reports and share them with others" and "Create simple dashboards" in this manual.

Save the results of a search

Saving the results of a search is different from saving the search itself. When you save a search, you're saving the search string and time range, so it can easily be run again in the future. When you save the results of a search, you are saving the outcome of a specific **search job**.

To save the results of a search, select *Save results* from the **Actions** dropdown menu. This causes Splunk to save the search job, and it works even if you are saving the results of a real-time search that is currently running. You can examine the results later by finding the job on the Jobs page. You get to the Jobs page by clicking the **Jobs** link in the upper-right hand corner of the Splunk interface.

For more information on managing search jobs through the Job Manager, see "Supervise your search jobs" in this manual.

Share search results

Sharing search results is different from sharing a saved search. When you share search results you are making the results of a particular search job available to other people.

If you would like to share search results with others, you have a couple of options.

- **Export the event data to a file.** Select *Export results...* from the **Actions** dropdown menu to open the **Export results** window. This window enables you to export the event data from your search job to a csv, raw, xml, or json file. You can then archive this file or use it with a third party charting application. You can export up to 10,000 results from a search job via **Splunk Web**.

  If you want to export more than 10,000 results, run the same search in **CLI** and append the `-maxout 0` option to it. For example, if you want to write all of your events to a file called `events.out`, then you would run a CLI search like this:

  ```
  splunk search '*' -maxout 0 > events.out
  ```

  For details on how the CLI works, refer to "About the CLI" in the Admin Manual.

- **Get (and share) a link to the results.** Select *Get link to results...* from the **Actions** dropdown menu to get a URL link to the report results. You can share this link with other interested parties, who can view the job it links to as long as they have access to your instance of Splunk and the job exists in the system.

Note: Selecting *Get link to results...* automatically saves your search job, which you can access thereafter through the Jobs page. The Get Link to Results popup window enables you to undo this save action.

Managing saved search navigation

When you save a search, it should appear in one of the drop-down lists in the top-level navigation menu. In the Search app, for example, new searches appear in the **Searches & Reports** list by default.

If you have write permissions for an app, you can change this default location, and even set things up so that searches with particular keywords in their names are automatically placed in specific categories in the navigation menu. For example, you can set things up so that Splunk automatically places saved searches with the word "website" in their name into a list of website-related searches in the navigation menu. You can also move searches from the default list to different locations in the top-level navigation menu.

For an overview of the navigation setup options that are available for saved searches and reports, see "Define navigation for saved searches and reports" in the Knowledge Manager manual. For the app navigation setup details, see "Build navigation for your app" in the Developer manual.

Answers

Have questions? Visit Splunk Answers and see what questions and answers the Splunk community has around saved searches.

Supervise your search jobs

Supervise your search jobs

Each time you run a search or generate a report, Splunk creates a job that contains the event data returned by that search or report. The Job Manager enables you to review and oversee your recently dispatched jobs, as well as those you may have saved earlier.

Just to be clear, *jobs* are not the same as *saved searches* and *saved reports*. Saved searches and saved reports contain data used to run those searches and reports, such as the search string and the time arguments used to dispatch searches. Jobs are artifacts of previously run searches and reports. They contain the results of a particular run of a search or report. Jobs are dispatched by scheduled searches as well as manual runs of searches and reports in the user interface.

For more information about saving searches see "Save searches" in this manual. For more information about saving reports, see "Save reports" in this manual.

You access the Job Manager by clicking the **Jobs** link in the upper right hand corner of the screen.

Note: When backgrounded searches are in progress, the **Jobs** link appears with the number of running jobs in parentheses.

Use the Job Manager to:

- See a list of the jobs you've recently dispatched or saved for later review, and use it to compare job statistics (run time, total count of events found, and so on).
- View the results of any job listed in the Job manager.
    - If the job is related to a search, you'll see the results in the Search view.
    - If the job is related to a report, you'll see the results in the Format Report page of the Report Builder.
    - **Note:** If a job is canceled while you have the Job Manager window open it can still appear in the Job Manager list, but you won't be able to view it. If you close and reopen the Job Manager, the canceled job should disappear.
- Check on the progress of ongoing backgrounded jobs or jobs dispatched by scheduled searches and pause or stop them if necessary.

- Save or delete any search or report job displayed on the Job Manager (if you have permissions to see and manage them).

Note: The Job Manager only displays search jobs that you have permission to see and manage.

Keep in mind that unsaved search jobs expire within a set period of time after they complete. The default lifespan for manually run search jobs is 10 minutes (search jobs resulting from scheduled searches typically have much shorter lifespans). The **Expires** column tells you how much time each list job has before it is deleted from the system. If you want to be able to review a search job after that expiration point, or share it with others, save it.

In most views you can save the last search or report job you ran without accessing the Job Manager page, as long as the job hasn't already expired.

- If you want to save a search job, select *Save results* from the **Actions** dropdown menu in the Search view.
- If you want to save a report job, select *Save results only* from the **Save** menu on the Report formatting page of the Report Builder.

For more information, see "About jobs and jobs management" in the Admin manual.

Automate Monitoring

Monitor recurring situations

Monitor recurring situations

If you've read the preceding chapters you have a pretty good idea of how to use Splunk's powerful search capabilities to learn all kinds of things about the event data in your system. But this doesn't help you with the myriad of recurring situations that everyone in IT is faced with on a regular basis. You can't be running searches yourself all of the time.

This is why we've designed Splunk to be the most flexible monitoring tool in your arsenal. Every search you design can be set up to run automatically on a regular schedule. And any scheduled or real-time search can be configured to send alert messages to you and other interested parties when specific circumstances are met. You can base these alerts on a wide range of threshold and trend-based scenarios, including empty shopping carts, brute force firewall attacks, and server system errors.

In this chapter you'll find:

- A nuts-to-bolts explanation of alert creation, for both scheduled and real-time searches.
- A variety of alert use case examples.
- A guide to the Alert Manager, which enables you to manage recently triggered alerts.

Create an alert

Create an alert

Splunk **alerts** are based on **saved searches** that run either on a regular interval (if the saved search is a standard **scheduled search**) or in real time (if the saved search is a **real-time search**). The alert is triggered when the results of the scheduled or real-time search meet a particular condition that you add to the alert definition.

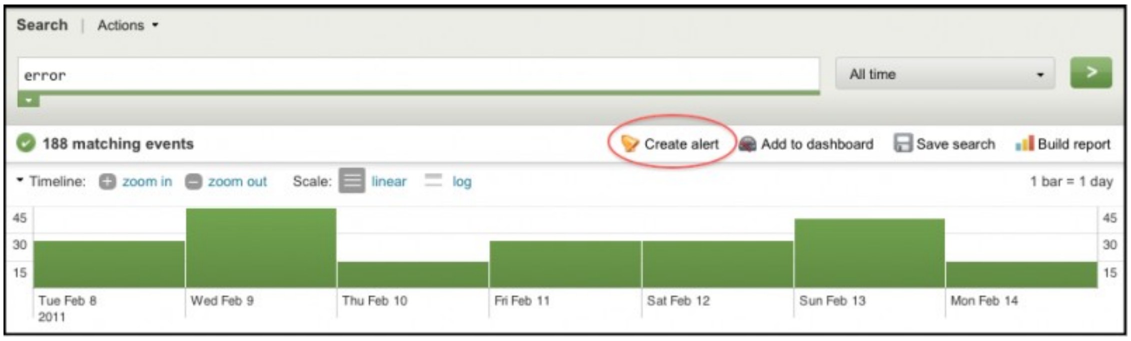
For example, you could base an alert on a scheduled search that runs every ten minutes. When it runs, it looks back at the syslog events that have been received from a particular host over the past 10 minutes. The alert is only triggered when the search returns a syslog event from this host that has a "disk full" error message. When the alert is triggered, an email goes out to a set of system administrators to inform them that the error has come up.

However, if your system administrators need *even timelier* alerts, you can base the disk-full alert on a real-time search that runs continuously in the background, looking at incoming syslog events from the same host. The moment the real-time search returns an event with the "disk full" error message, the alert is triggered, and Splunk sends an alert to the same set of system administrators.

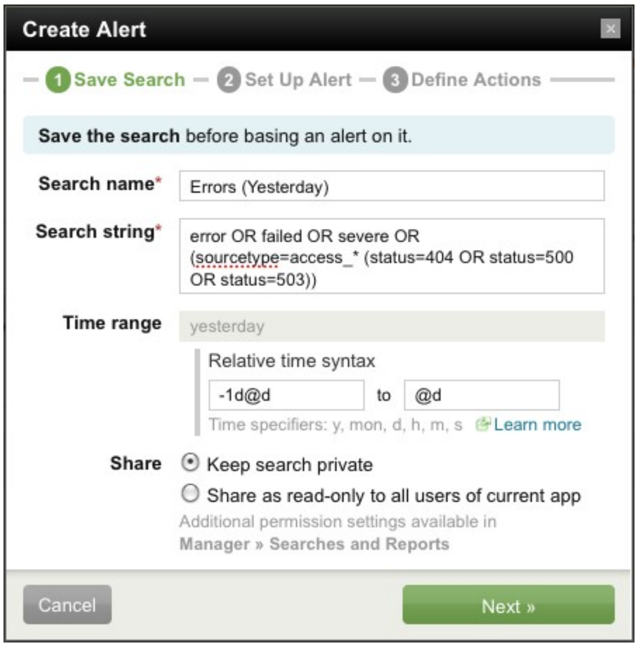
Note: When Splunk is used out-of-the-box, *only users with the Admin role* can run and save real-time searches, schedule searches, or create alerts. In addition you cannot create searches unless your role permissions enable you to do so. For more information on managing roles, see "Add and edit roles" in the Admin Manual.

Start by defining and saving a search

If you run a search, like the results it's giving you, and decide that you want to create an alert based on it, then click the **Create alert** link that appears above the search timeline after the search starts running. This opens the **Create Alert** window. The **Create Alert** window is broken up into three steps: Save Search, Set Up Alert, and Define Actions.



The first step, Save Search, enables you to define and save the search. If you need more information about this step, see "Save searches and share search results" in this manual.



Save the search before basing an alert on it.

Search name*: Errors (Yesterday)

Search string*: error OR failed OR severe OR (sourcetype=access_* (status=404 OR status=500 OR status=503))

Time range: yesterday

Relative time syntax: -1d@d to @d

Time specifiers: y, mon, d, h, m, s Learn more

Share: [✓ Keep search private] Share as read-only to all users of current app

Additional permission settings available in Manager » Searches and Reports

This topic focuses on the last two steps of the **Create Alert** window, which are specific to alert creation:

- **Set Up Alert**: This is where you define the condition that must be met for the search to be triggered, the schedule of the search upon which the alert is based (if the search is not a real-time search), and the alert throttling period, expiration time, and severity level.
- **Define Actions**: This is where you determine what happens when the alert conditions are met by the results of a scheduled run of the search and an alert is triggered. Alert actions include sending alert emails to a designated list of recipients, posting alert information to an RSS feed, and running a user-designated script that performs another action. The **Define actions** step is

also where you determine whether this alert is tracked by the Alert manager when it is triggered.

Note: If you need to add or update alert settings for a *preexisting* saved search, go to **Manager > Searches and Reports** and locate the saved search. Click the search name to open the search detail page. This page contains all of the settings that you would otherwise see in the **Create Alert** window. You may need to select **Schedule this search** to expose the scheduling and alert setup controls.

When you are in Manager, keep in mind that you can only edit existing searches that you have both read *and* write permissions for. Searches can also be associated with specific apps, which means that you have to be using that app in order to see and edit the search. For more information about sharing and promoting saved searches (as well as other Splunk **knowledge objects**), see "Curate Splunk knowledge with Manager" in the Knowledge Manager manual.

For a series of alert examples showing how you might design alerts for specific situations using both scheduled and real-time searches, see "Alert use cases"

Set up the alerting condition

Alerts are triggered when a triggering condition is met. You can set up basic and advanced conditional alerts. You do this in Set Up Alert, the second step of the **Create Alert** dialog box.

A **basic conditional alert** is triggered when the number of events, sources, or hosts in the results of a scheduled or real-time search meet a simple alerting condition (for example, a basic conditional alert can be triggered when the number of events returned by its search are greater than, less than, or equal to a number that you provide).

An **advanced conditional alert** uses a secondary "conditional" search to evaluate the results of the scheduled or real-time search. With this setup, the alert is triggered when the secondary search returns *any* results.

You can also set up a scheduled search to alert every time it runs by setting **Condition** to *always*. This setting is typically used for **summary indexing** or to deliver a PDF report on some recurring interval.

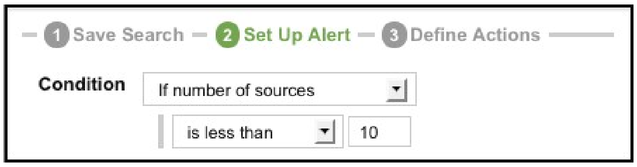
You can find these controls in the Set Up Alert step of the **Create Alert** window. You can also find them in **Manager > Searches and Reports**, when you add a new saved search or click through to the detail page for an existing saved search.

Define a basic conditional alert

On Set Up Alert, the second step of the **Create Alert** dialog box, follow this procedure to define a basic conditional alert that notifies you when a simple alerting condition is met by the number of events, hosts, or sources returned by the search.

1. In the **Condition** section, determine what basic condition triggers the alert. Choose either *If the number of events*, *If the number of hosts*, or *If the number of sources* from the **Condition** list, depending on what you want to track. Choosing one of these three values causes the comparison operation field to appear underneath the list.

Note: Alternatively, you can select a **Condition** of *always* to have Splunk trigger the alert each time the search is run. This can be handy if the search runs on an infrequent basis and you just want to be notified of the results, no matter what they are.



Condition: If number of sources
is less than 10

2. Choose a comparison operation from the list that appears after one of the aforementioned *If the number of* values is selected. Click *is greater than*, *is less than*, *is equal to*, *drops by*, or *rises by* depending on how you want the alerting condition to work. (The *rises by* and *drops by* operations are not available for real-time searches.)

3. In the field adjacent to the comparison operation list, enter an integer to complete the basic conditional alert. This integer represents a number of events.

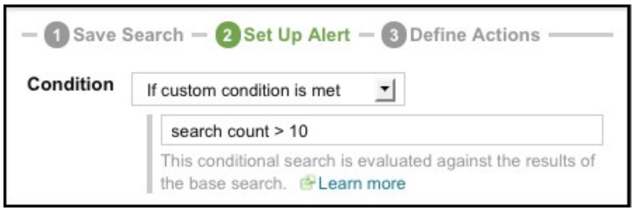
Note: Basic conditional alerts work differently for real-time searches and standard searches that are set up to run on a schedule:

- **For a scheduled search,** the alert is triggered if the results of a scheduled run of the search meet or exceed the set condition. For example, you can set up an alert for a scheduled search that sends out a notification *if the number of events* returned by the search *rises by* a threshold of *10* or more events since the last time the search was run.
- **For a real-time search,** the alert is triggered if the set condition occurs *within* the time range window of the search. For example, you could have a real-time search alert that triggers the moment that five or more events are returned by the search within its sliding 60-second window. (Just to be clear, this means that if the real-time search returns one event and then four more events five minutes later, the alert is *not* triggered. But if all five events are returned within a single 60-second span of time, the alert *is* triggered.)

Define an advanced conditional alert

Advanced conditional alerting enables you to set up an alerting condition that is based on the result of a conditional search that is applied to the results of the scheduled search. When the secondary search returns *any* results, the alert is triggered (so you should develop a secondary search that usually returns zero results unless the alerting conditions have been met).

By basing your alert conditions on the result of a secondary conditional search, you can define specific conditions for triggering alerts and reduce the incidence of false positive alerts.



Condition: If custom condition is met
search count > 10
This conditional search is evaluated against the results of the base search.

Follow this procedure to define an advanced conditional alert:

1. In the **Condition** list, click *if custom condition is met.* The **Conditional search** field appears.

2. Enter your conditional search in the **Custom condition search** field.

When the conditional search returns 1 or more events, the alert is triggered. If you have arranged for email to go out, take note that results of the *original* scheduled search are sent to stakeholders (*not* the results of the conditional search).

Note: Advanced conditional alerts work differently for real-time searches and standard searches that are set up to run on a schedule:

- For a **scheduled** search, the alert is triggered when the conditional search evaluates the results of a run of the original scheduled search and returns one or more results.
- For a **real-time** search, the secondary, conditional search runs in real time as well. It continuously evaluates the results returned in the time range window of the original real time search. When a single event is returned by the conditional search, the alert is triggered.

Advanced conditional alert example

Lets say you're setting up an alert for the following scheduled search, which is scheduled to run every 10 minutes:

```
failed password | stats count by user
```

This search returns the number of incorrect password entries associated with each user name.

What you want to do is arrange to have Splunk trigger the alert when the scheduled search finds more than 10 password failures for any given user. When the alert is triggered, an email containing the results of the triggering search gets sent to interested parties.

Now, it seems like you could simply append `| search count > 10` to the original scheduled search:

```
failed password | stats count by user | search count > 10
```

Unfortunately, if you create a basic conditional alert based on this search, where an alert is triggered if the number of events returned is greater than 1, you won't get the behavior you desire. This is because this new search only returns user names that are associated with 10+ failed password entries--the actual count values are left out. When the alert is triggered and the results are emailed to stakeholders, you want the recipients to have a listing that matches each user name to the precise number of failed password attempts that it is associated with.

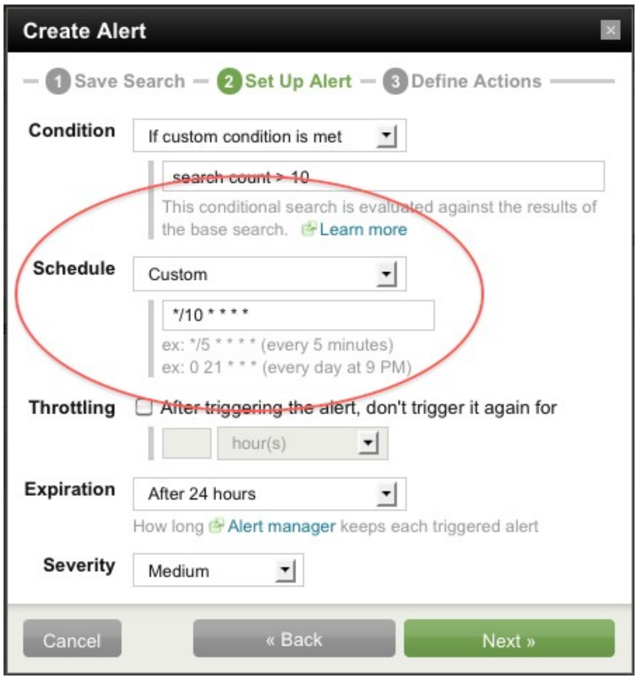
What you want to do is set **Condition** to *If custom condition is met* and then place `search count > 10` in the **Custom condition search** field (while removing it from the base search). This conditional search runs against the results of the original scheduled search (`failed password | stats count by user`). With this, the alert is triggered *only* when the custom condition is met--when there are 1 or more user names associated with 10 failed password entries. But when it is triggered, the results of the *original* search--the list of user names and their failed password counts--is sent to stakeholders via email.

Note: If this search used the same string but was running in real-time, it will work the same way as long as its time range window is set to be 10 minutes wide. As soon as the real-time search returns 10 failed password entries for the same user within that 10-minute span of time, the alert is triggered.

For more examples of scheduled and real-time alerts, see "Alert examples," in this manual.

Schedule the search

Alerts that are based on standard searches (searches that collect data from the past rather than in real time, in other words) require that those searches be scheduled to run on a regular interval, such as every 10 minutes, every two hours, or every night at midnight. Use the **Schedule** list in Set Up Alert, the second step of the **Create Alert** dialog box. to define this interval. The **Schedule** list will not display for real-time searches.


Create Alert

1 Save Search — 2 Set Up Alert — 3 Define Actions

Condition: If custom condition is met

search count > 10

This conditional search is evaluated against the results of the base search. Learn more

Schedule: Custom

*/10 * * * *

ex: */5 * * * * (every 5 minutes)
ex: 0 21 * * * (every day at 9 PM)

Throttling: ☐ After triggering the alert, don't trigger it again for [ ] hour(s)

Expiration: After 24 hours

How long Alert manager keeps each triggered alert

Severity: Medium

Then, pick an alert schedule. You can pick a pre-set schedule like *every 12 hours, every 30 minutes, every day at 6pm* and so on, or you can pick*custom*, which enables you to set up an interval using standard cron notation.

Note: Splunk only uses 5 parameters for cron notation, not 6. The parameters (`* * * * *`) correspond to `minute hour day month day-of-week`. The 6th parameter for `year`, common in other forms of cron notation, is not used.

Here are some cron examples:

```
*/5 * * * *         : Every 5 minutes
*/30 * * * *        : Every 30 minutes
0 */12 * * *        : Every 12 hours, on the hour
*/20  * * * 1-5     : Every 20 minutes, Monday through Friday
0 9 1-7 * 1         : First Monday of each month, at 9am.
```

Coordinate the search schedule with the search time range

When you set up an interval for a scheduled search, it's good to keep the time range that you've defined for the search in mind. This can be especially true for distributed search setups where event data may not reach the indexer exactly when it is generated. In this case, it can be a good idea to schedule your searches with at least 60 seconds of delay.

This example sets up a search that runs every hour at the half hour, but collects an hour's worth of event data, beginning an hour and a half *before* the search is actually run. This means that when the scheduled search kicks off at 3:30pm, it is collecting the event data that Splunk indexed from 2:00pm to 3:00pm.)

- Set an **Earliest time** value of *-90m* and a **Latest time** value of *-30m*.
- Set an **Alert schedule** *custom* cron notation of `30 * * * *` to have your search run every hour on the half hour.

For more information about the relative time modifier syntax for search time range definition, see "Syntax for relative time modifiers" in this manual.

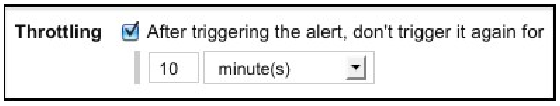
Manage the priority of concurrently scheduled searches

Depending on how you have your Splunk implementation set up, you may only be able to run one scheduled search at a time. Under this restriction, when you schedule multiple searches to run at approximately the same time, Splunk's search scheduler works to ensure that all of your scheduled searches get run consecutively for the period of time over which they are supposed to gather data. However, there are cases where you may need to have certain searches run ahead of others in order to ensure that current data is obtained, or to ensure that gaps in data collection do not occur (depending on your needs).

You can configure the priority of scheduled searches through edits to `savedsearches.conf`. For more information about this feature, see "Configure the priority of scheduled searches" in the Knowledge Manager manual.

Throttle frequent alerts

The **Throttling** controls in Set Up Alert, the second step of the **Create Alert** dialog box, help you determine how soon you would like to be notified again after receiving an alert. They are especially useful for real-time searches where triggering conditions might be met over and over again in a very short amount of time. You can use the **Throttling** settings to ensure that multiple occurrences of the alert are suppressed for a given number of *minutes*, *hours*, or *days*.


Throttling ☑ After triggering the alert, don't trigger it again for
10 minute(s)

For example, you could have a real-time search with a 60 second window that alerts every time an event with "disk error" event appears. If ten events with the "disk error" message come in within that window, five disk error alerts will be triggered--ten alerts within one minute. And if the alert is set up so that an email goes out to a set of recipients each time it's triggered (see "Specify alert actions," below), those recipients probably won't see the stack of alert emails as being terribly helpful.

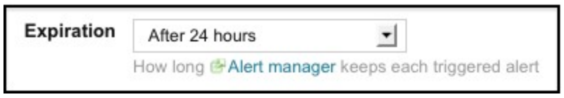
You can set the **Throttling** controls so that when one alert of this type is triggered, all successive alerts of the same type are suppressed for the next 10 minutes. When those 10 minutes are up, the alert can be triggered again and more emails can be sent out as a result--but once it is triggered, another 10 minutes have to pass before it can be triggered a third time.

In general, when you set up throttling for a real-time search it's best to start with a throttling period that matches the length of the base search's window, and then expand the throttling period from there, if necessary. This prevents you from getting duplicate notifications for a given event.

Note: Throttling settings are not usually required for scheduled searches, because only one alert is sent out per run of a scheduled search. But if the search is scheduled to run on a very frequent basis (every five minutes, for example), you can set the throttling controls to suppress the alert so that a much larger span of time--say, 60 minutes--has to pass before Splunk can send out another alert of the same type.

Define the alert retention time

You can determine how long Splunk keeps a record of your triggered alerts. When you are creating a new alert, use the **Expiration** list in Set Up Alert, the second step of the **Create Alert** dialog box. to define the amount of time that its triggered alert records (and their associated search artifacts) will be retained by Splunk. You can manage alert expiration for preexisting alerts in **Manager > Searches and Reports**. Just find the search that the alert is associated with and go to its detail page.


Expiration After 24 hours
How long Alert manager keeps each triggered alert

You can choose a preset expiration point for the alert records associated with this search, such as *after 5 hours*, or you can define a custom expiration point.

Note: If you set an expiration time for the alert records, be sure to also select the **Tracking** checkbox on Define Actions, the third step of the **Create Alert** dialog box. Splunk will not keep records of the triggered alerts in the Alert Manager unless **Tracking** is selected.

To review and manage your triggered alerts, go to the Alert manager by clicking the **Alerts** link in the upper right-hand corner of the Splunk interface. For more information about using it, see the "Review triggered alerts" topic in this manual.

Give the alert a severity level

Each alert can be labeled with a severity level that helps people know how important a triggered alert is in relation to other alerts. For example, an alert that lets you know that a server is approaching disk capacity could be given a *High* label, while an alert triggered by a "disk full" error could have a *Critical* label. You can define the alert severity label for a new alert in Set Up Alert, the second step of the **Create Alert** dialog box.

Severity labels are informational in purpose and have no additional functionality. You can use them to quickly pick out important alerts from the alert listing on the Alerts page, which you can get to by clicking the **Alerts** link in the upper right-hand corner of the Splunk interface.

Specify alert actions

When the results of a scheduled search meet the alerting conditions for that search, an alert is triggered, causing an alert action to take place. You define alert actions for a new alert in Define Actions, the third step of the **Create Alert** dialog box.

There are three kinds of alert actions:

- **Notification by email** - Splunk sends an email to a list of recipients that you define, and it can optionally contain the results of the triggering search job.
- **Addition to an RSS feed** - Splunk posts an alert notification to an RSS feed for the alert.
- **Run of a shell script** - Splunk runs a shell script that can perform some other action, such as the sending of an SNMP trap notification or the calling of an API. You determine which script is run.
- **Tracking** - Select to have triggered alerts display in the Alert manager.

You can enable any combination of these alert action types for a single alert.

Notification by email

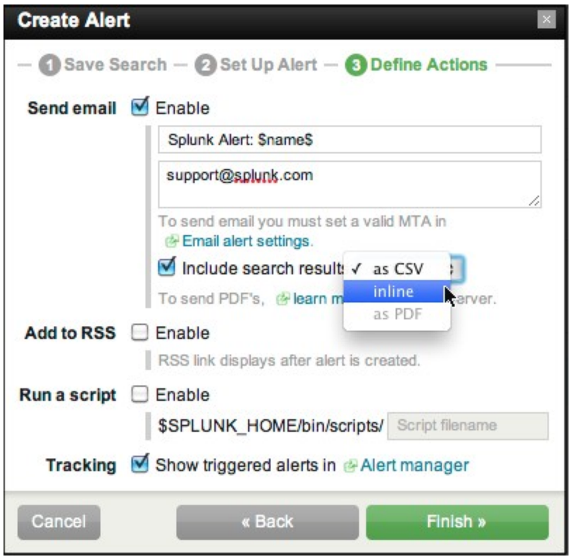
If you want Splunk to contact stakeholders when the alert is triggered, select *Enable* next to **Send email**.

You can enter a subject header for the email (by default it is set to be *Splunk Alert: $name$*, where *$name$* is replaced by the saved search name) and a comma-separated list of email addresses to which the alert should be sent. Select **Include search results' *and have that set to*** *as CSV*, *inline*, or *as PDF*. If you have the results sent as CSV or PDF, they will be an attachment to the email notification.

Note: For your email alert notifications to work correctly, you first need to have your email alert settings configured in Manager. See the subtopic "Configure email alert settings in Manager," below.

Send results in alert emails

You can arrange to have email alert notifications contain the results of the searches that trigger them. This works best when the search returns a truncated list (such as a list that returns the top 20 results) or a table. To do this, select **Include search results** and then use the list to identify the format that the results should be delivered in. You can have the results sent inline (as part of the body of the alert email, in other words), or you can have it delivered as a .csv or .pdf attachment with the alert email.



Send email ✓ Enable

Splunk Alert: $name$

support@splunk.com

To send email you must set a valid MTA in Email alert settings.

✓ Include search results [✓ as CSV] inline as PDF

To send PDF's, learn m… …erver.

Add to RSS ☐ Enable

RSS link displays after alert is created.

Run a script ☐ Enable

$SPLUNK_HOME/bin/scripts/ Script filename

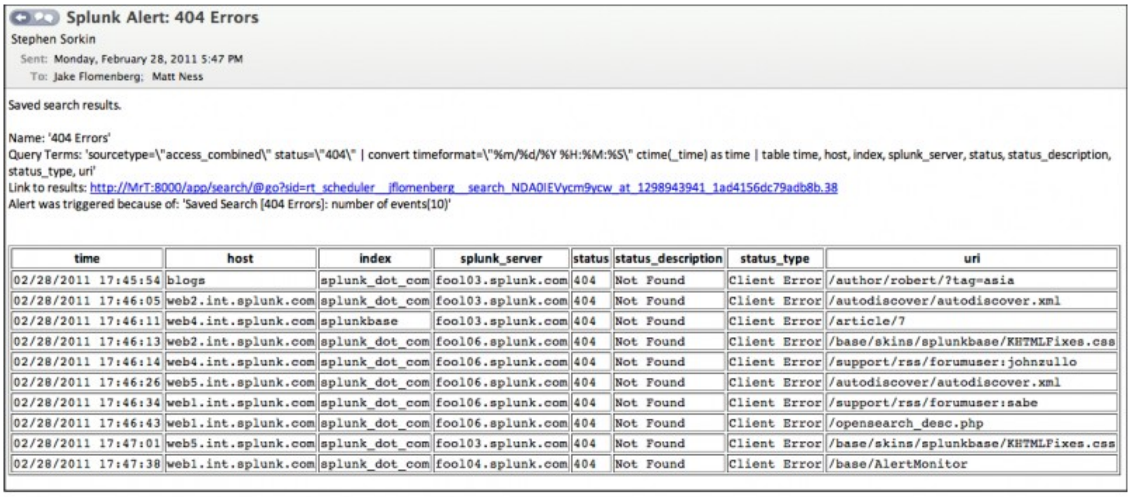
Tracking ✓ Show triggered alerts in Alert manager

The method of inclusion is controlled via `alert_actions.conf` (at a global level) or `savedsearches.conf` (at an individual search level); for more information see "Set up alerts in savedsearches.conf" in the Admin Manual.

Important: You cannot configure alerts to send results as PDF attachments until you install the PDF Printer app on a central Linux host. If you aren't running this app, the *as PDF* option for **Include search results** will be unavailable. To get PDF printing working, contact a system administrator. For more information see "Configure PDF printing for Splunk Web" in the Installation manual.

You can also arrange to have PDF printouts of dashboards delivered by email on a set schedule. For more information, see "Schedule delivery of dashboard PDF printouts via email" in this manual.

The following is an example of what an email alert looks like:



Splunk Alert: 404 Errors

Stephen Sorkin

Sent: Monday, February 28, 2011 5:47 PM

To: Jake Flomenberg; Matt Ness

Saved search results.

Name: '404 Errors'
Query Terms: 'sourcetype=\"access_combined\" status=\"404\" | convert timeformat=\"%m/%d/%Y %H:%M:%S\" ctime(_time) as time | table time, host, index, splunk_server, status, status_description, status_type, uri'
Link to results: http://MrT:8000/app/search/@go?sid=rt_scheduler__jflomenberg__search_NDA0IEVycm9ycw_at_1298943941_1ad4156dc79adb8b.38
Alert was triggered because of: 'Saved Search [404 Errors]: number of events(10)'

time	host	index	splunk_server	status	status_description	status_type	uri
02/28/2011 17:45:54	blogs	splunk_dot_com	foo103.splunk.com	404	Not Found	Client Error	/author/robert/?tag=asia
02/28/2011 17:46:05	web2.int.splunk.com	splunk_dot_com	foo103.splunk.com	404	Not Found	Client Error	/autodiscover/autodiscover.xml
02/28/2011 17:46:11	web4.int.splunk.com	splunkbase	foo103.splunk.com	404	Not Found	Client Error	/article/7
02/28/2011 17:46:13	web2.int.splunk.com	splunk_dot_com	foo106.splunk.com	404	Not Found	Client Error	/base/skins/splunkbase/KHTMLFixes.css
02/28/2011 17:46:14	web4.int.splunk.com	splunk_dot_com	foo106.splunk.com	404	Not Found	Client Error	/support/rss/forumuser:johnzullo
02/28/2011 17:46:26	web5.int.splunk.com	splunk_dot_com	foo106.splunk.com	404	Not Found	Client Error	/autodiscover/autodiscover.xml
02/28/2011 17:46:34	web1.int.splunk.com	splunk_dot_com	foo106.splunk.com	404	Not Found	Client Error	/support/rss/forumuser:sabe
02/28/2011 17:46:43	web1.int.splunk.com	splunk_dot_com	foo106.splunk.com	404	Not Found	Client Error	/opensearch_desc.php
02/28/2011 17:47:01	web5.int.splunk.com	splunk_dot_com	foo103.splunk.com	404	Not Found	Client Error	/base/skins/splunkbase/KHTMLFixes.css
02/28/2011 17:47:38	web1.int.splunk.com	splunk_dot_com	foo104.splunk.com	404	Not Found	Client Error	/base/AlertMonitor

Configure email alert settings in Manager

Email alerting will not work if the email alert settings in Manager are not configured, or are configured incorrectly. You can define these settings at **Manager > System settings > Email alert settings**. For details on how to fill out this screen, see "How alerting works" in the Admin manual.

If you don't see **System settings** or **Email alert settings** in Manager, you do not have permission to edit the settings. In this case, contact your Splunk Admin.

You can also use configuration files to set up email alert settings. You can configure them for your entire Splunk implementation in `alert_actions.conf`, and you can configure them at the individual search level in `savedsearches.conf`. For more information about .conf file management of saved searches and alert settings see "Set up alerts in savedsearches.conf" in the Admin Manual.

In Splunk Web, navigate to **Manager > System settings > Email alert settings**. Here you can define the **Mail server settings** (the mail host, username, password, and so on) and the **Email format** (link hostname, email subject & format, include inline results, and so on).

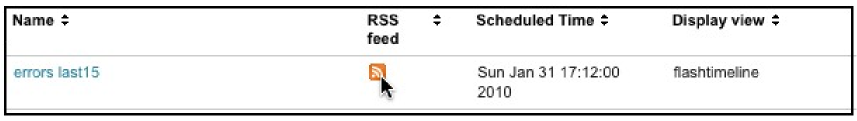
Note: If you are using PDF Server, the link hostname field must be the search head hostname for the instance sending requests to a PDF Report Server. Set this option only if the hostname that is autodetected by default is not correct for your environment.

Create an RSS feed

If you want Splunk to post this alert to an RSS feed when it is triggered, select *Enable* next to **Add to RSS**.

When the alert conditions are met for a scheduled search that has RSS feed notification enabled, Splunk sends a notification out to its RSS feed. The feed is located at `http://[splunkhost]:[port]/rss/[saved_search_name]`. So, let's say you're running a search titled "errors_last15" and have a Splunk instance that is located on `localhost` and uses port 8000, the correct link for the RSS feed would be `http://localhost:8000/rss/errors_last15`.

You can also access the RSS feed for a scheduled search through **Manager > Searches and Reports**. If a scheduled search has been set up to provide an RSS feed for alerting searches, when you look it up on the Searches and reports page, you will see a RSS symbol in the **RSS feed** column:



Name ≑	RSS feed ≑	Scheduled Time ≑	Display view ≑
errors last15		Sun Jan 31 17:12:00 2010	flashtimeline

You can click on this symbol to go to the RSS feed.

Note: The RSS feed for a scheduled search will not display any searches until the search has run on its schedule *and* the alerting conditions that have been defined for it have been met. If you set the search up to alert each time it's run (by setting **Perform actions** to *always*), you'll see searches in the RSS feed after first time the search runs on its schedule.

Warning: The RSS feed is exposed to any user with access to the webserver that displays it. Unauthorized users can't follow the RSS link back to the Splunk application to view the results of a particular search, but they can see the summarization displayed in the RSS feed, which includes the name of the search that was run and the number of results returned by the search.

Here's an example of the XML that generates the feed:

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0">
    <channel>
        <title>Alert: errors last15</title>
        <link>http://localhost:8000/app/search/@go?sid=scheduler_Z2d1cHRh</link>
        <description>Saved Searches Feed for saved search errors last15</description>
        <item>
            <title>errors last15</title>
            <link>http://localhost:8000/app/search/@go?sid=scheduler_Z2d1cHRh</link>
            <description>Alert trigger: errors last15, results.count=123 </description>
            <pubDate>Mon, 01 Feb 2010 12:55:09 -0800</pubDate>
        </item>
    </channel>
</rss>
```

Run a script

If you want Splunk to run an alert script when the alert is triggered, select *Enable* for **Run a script** and enter the file name of the script that you want Splunk to execute.

For example, you may want an alert to run a script that generates an SNMP trap notification and sends it to another system such as a Network Systems Management console when its alerting conditions are met. Meanwhile, you could have a different alert that, when triggered, runs a script that calls an API, which in turn sends the triggering event to another system.

Note: For security reasons, all alert scripts must be placed in the `$SPLUNK_HOME/bin/scripts` directory. This is where Splunk will look for any script triggered by an alert.

Check out this excellent topic on troubleshooting alert scripts on the Splunk Community Wiki.

For more details on configuring alerts, including instructions for configuring alerts using `savedsearches.conf`, see the Admin Manual topic on alerts.

Track triggered alerts

If you want to have the Alert Manager keep records of the triggered alerts related to a particular alert configuration, select the **Tracking** checkbox. The Alert Manager will keep records of triggered alerts for the duration specified in the **Expiration** field on the Set Up Alert step of the **Create Alert** dialog box.

For more information about the Alert Manager and how it is used, see the "Review triggered alerts" topic in this manual.

Setting up tracking when upgrading to 4.2

When you upgrade your Splunk instance to 4.2, be aware that by default existing alerts do NOT show up in the alert manager. To quickly update your existing alerts so that they show up in the alert manager, edit the relevant copy of `savedsearches.conf`. Add `alert.track = true` to the stanzas of each saved search that you have set up as an alert and want to see tracked in the Alert Manager. Review "About configuration files" in the Admin Manual for details about configuration files.

Specify fields to show in alerts through search language

When Splunk sends out alert emails with the results of the alerting search job, Splunk includes all the fields in those results. If you want to have certain fields included in or excluded from the results, you need to use the fields command in your saved search string to specify the field inclusions or exclusions.

- To eliminate a field from the search results, pipe your search to `fields - $FIELDNAME`.
- To add a field to the search results, pipe your search to `fields + $FIELDNAME`.

You can specify multiple fields to include and exclude in one string. For example, your Search field may be:

```
yoursearch | fields - $FIELD1,$FIELD2 + $FIELD3,$FIELD4
```

The alert you receive will exclude `$FIELD1` and `$FIELD2`, but include `$FIELD3` and `$FIELD4`.

Enable summary indexing

Summary indexing is an action that you can configure for any alert. You use summary indexing when you need to perform analysis/reports on large amounts of data over long timespans, which typically can be quite time consuming, and a drain on performance if several users are running similar searches on a regular basis.

With summary indexing, you base an alert on a search that computes sufficient statistics (a summary) for events covering a slice of time. The search is set up so that each time it runs on its schedule, the search results are saved into a summary index that you designate. You can then run searches against this smaller (and thus faster) summary index instead of working with the much larger dataset from which the summary index receives its events.

To set up summary indexing for an alert, go to **Manager > Searches and Reports**, and either add a new saved search or open up the detail page for an existing search or alert. (You cannot set up summary indexing through the **Create Alert** window.) To enable the summary index to gather data on a regular interval, set its **Alert condition** to *always* and then select **Enable** under **Summary indexing** at the bottom of the view.

Note: There's more to summary indexing--you should take care how you construct the search that populates the summary index and in most cases special reporting commands should be used. **Do not attempt to set up a summary index until you have read and understood "Use summary indexing for increased reporting efficiency" in the Knowledge Manager manual.**

Alert examples

Alert examples

This topic presents a variety of alerting use cases that show you how alerts can help provide operational intelligence around network security, system administration, SLA compliance, and more.

Alerting when an event occurs - "File system full" error

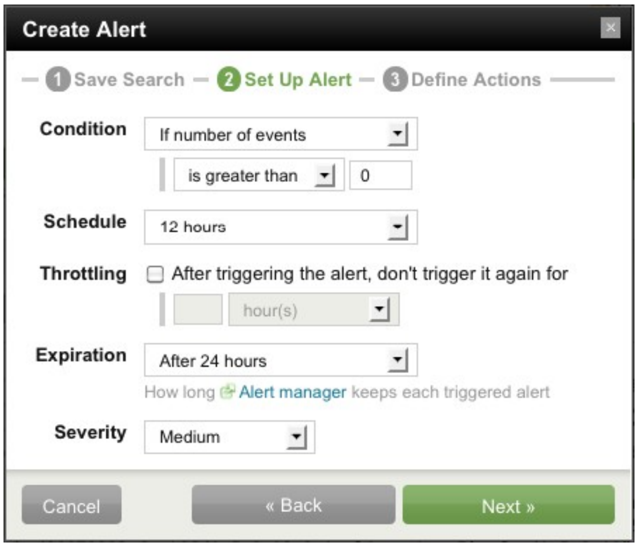
Imagine that you're a system administrator who needs prompt notification when a `file system full` error occurs on any host--but you don't want to get more than one alert per 10 minute period.

Here's an example of the sort of events that you're on the lookout for:

```
Jan 27 03:21:00 MMPAD last message repeated 3 times
Jan 27 03:21:00 MMPAD ufs: [ID 845546 kern.notice] NOTICE: alloc: /:file system full
Jan 27 03:21:49 MMPAD ufs: [ID 845546 kern.notice] NOTICE: alloc: /:file system full
Jan 27 03:22:00 MMPAD last message repeated 3 times
Jan 27 03:22:00 MMPAD ufs: [ID 845546 kern.notice] NOTICE: alloc: /:file system full
Jan 27 03:22:56 MMPAD ufs: [ID 845546 kern.notice] NOTICE: alloc: /:file system full
```

Now, you can easily set up a simple search to find these events:

```
file system full
```
It's pretty easy to define a **basic conditional alert** that sends out an alert when the search returns one or more file system full error events. When you create the alert, just set the **Condition** to *If number of events is greater than 0*.



Condition: If number of events / is greater than 0
Schedule: 12 hours
Throttling: After triggering the alert, don't trigger it again for [ ] hour(s)
Expiration: After 24 hours
How long Alert manager keeps each triggered alert
Severity: Medium

However, you also need to determine whether the alert should be scheduled as a scheduled alert or a real-time alert. The deciding factor here is: *How quickly do you feel that you need to know about the "file system full" error?* Do you want to know the moment the error occurs, or is it ok if you are notified on a "timely" basis (in other words, when the alert is triggered by a search running on a particular schedule).

Using a real-time search to catch the "file system full" error event

If you want to be alerted shortly after the first "file system full" error is received, you'll want to set up a real-time alert with a one minute window (by giving it a **Time range** of *rt-1m* to *rt*). We suggest you use a one minute window rather than a shorter window (such as 30 seconds) for alerts because the real-time search *only* finds events that occur within the real-time window. A 30 second window might be fine if there is no latency in your system, but all too often search loads and other issues can cause certain events to come in late and be "missed" by the real-time search because its window is too brief.

If you find that a one minute window is too long for your purposes (because the alert has to be suppressed for at least that length of time) you have another option: you can keep the window short, but offset it by a minute. A **Time range** of *rt-90s* to *rt-60s* sets up a one minute buffer that allows all events to come in before they are captured by the 30 second real-time search window.

But remember, you only want to get one alert every 10 minutes, and as you can see from the event listing excerpt above, you can get numerous "file system full" errors in the space of just a few minutes. To deal with that, set **Throttling** to *10 minutes*. This ensures that once the first "file system full" error alert is triggered, successive "file system error" events won't trigger another alert until 10 minutes have passed.

With a 10-minute **Throttling** setting, you might get an alert email shortly after you start the search, with information on the "file system error" event that triggered it, as well as any other events that were captured within the one minute time window at the time the triggering event came in. But after that first alert is triggered, the 10 minute throttling period kicks in. During the throttling period, subsequent "file system error" events *do not* trigger the alert. Once the 10 minute throttling period is up, the alert can be triggered again when a new "file system error" event is received. When the alert is triggered, another email goes out, and the throttling period is enforced again for another 10 minutes. And so on. Each time the alert is triggered, the email that goes out only provides information on the one event that triggered it, plus any others that happen to fit within its 30-second time window.

Using a scheduled search to catch the "file system full" error event

On the other hand, if you just want to get *timely* notification for a "file system full" error event, you can base the alarm on a scheduled search that runs every 10 minutes over the past 10 minutes. Because the search is run on a 10 minute interval, you won't get alerts the moment a "file system full" event is received; instead you'll get an alert after the search runs on its schedule. If an alert is triggered by this scheduled search it returns a list all of the "file system error" events that were received during that 10 minute period (although only one event needs to be received to trigger the alert).

You don't need to set up a 10 minute throttling period for this search because the search already runs on a 10 minute interval. However, if you would prefer there be a longer interval between triggered alert actions, you could throttle subsequent alerts for whatever period over 10 minutes you deem appropriate. For example, you could choose to set a throttling period of 60 minutes, 2 hours, or even 1 day. It's up to you.

Use a custom search to alert when a statistical threshold is reached

In this example, you want to get an alert if Splunk detects an sshd brute force attack. This could be indicated by 5 or more failed ssh attempts within one minute, from the same ip address and targeted against a single host.

Here's an example of the events that you're on the lookout for:

```
Jun 26 22:31:04 victim sshd[15384]:   Failed password for root from ::ffff:w.x.y.z port 30937 s
Jun 26 22:31:06 victim sshd[15386]:   Illegal user network from ::ffff:w.x.y.z
Jun 26 22:31:06 victim sshd[15386]:   error: Could not get shadow information for NOUSER
Jun 26 22:31:06 victim sshd[15386]: Failed password    for illegal user network from ::ffff:w.x
Jun 26 22:31:08 victim sshd[15388]:   Illegal user word from ::ffff:w.x.y.z
Jun 26 22:31:08 victim sshd[15388]:   error: Could not get shadow information for NOUSER
Jun 26 22:31:08 victim sshd[15388]: Failed password    for illegal user word from ::ffff:w.x.y.
Jun 26 22:31:10 victim sshd[15390]: Failed password    for root from ::ffff:w.x.y.z port 30980
Jun 26 22:31:11 victim sshd[15392]: Failed password    for root from ::ffff:w.x.y.z port 30992
Jun 26 22:31:13 victim sshd[15394]: Failed password    for root from ::ffff:w.x.y.z port 31007
Jun 26 22:31:15 victim sshd[15396]: Failed password    for root from ::ffff:w.x.y.z port 31021
Jun 26 22:31:17 victim sshd[15398]: Failed password    for root from ::ffff:w.x.y.z port 31031
Jun 26 22:31:19 victim sshd[15400]: Failed password    for root from ::ffff:w.x.y.z port 31049
Jun 26 22:31:20 victim sshd[15403]: Failed password    for root from ::ffff:w.x.y.z port 31062
Jun 26 22:31:22 victim sshd[15405]: Failed password    for root from ::ffff:w.x.y.z port 31073
```

How do you set up this alert? You might begin by basing it on a fairly simple search:

```
Failed password sshd
```

This search finds the failed ssh attempts, and if that was all you were looking for, you could set a basic **Condition** like *greater than 4* to alert you when 5 such events come in within a given time window (if you base the alert on a real-time search) or search interval (if you base the alert on a scheduled search).

But this doesn't help you set up an alert that triggers when 5 or more matching events that come from the same ip address and are targeted against a single host come in within a one minute window. To do this you need to define an **advanced conditional alert** that uses a secondary custom search to sort through the events returned by the base search. But before you design that, you should first fix the base search.

Base search - Refine to provide better, more useful results

The base search results are what gets sent in the alert email or rss feed update when the alert is triggered. You should try to make those results as informative as possible.

The following search goes through the "failed password sshd" events that it finds within a given one minute window or interval, groups the events together according to their `destip` and `srcip` combination, generates a count of each combination, and then arranges this information in an easy-to-read table.

```
Failed password sshd | stats count by srcip,destip | table srcip destip
count
```

The table below gives you an idea of the results that this search would return. It shows that four different groups of events were found, each with a specific `scrip/destip` combination, and it provides a count of the events in each group. Apparently, there were eight failed ssh password attempts from someone at `<srcip1>` that targeted `<destip2>`, all within a one minute timeframe.

srcip	destip	count
<srcip1>	<destip1>	3
<srcip1>	<destip2>	8

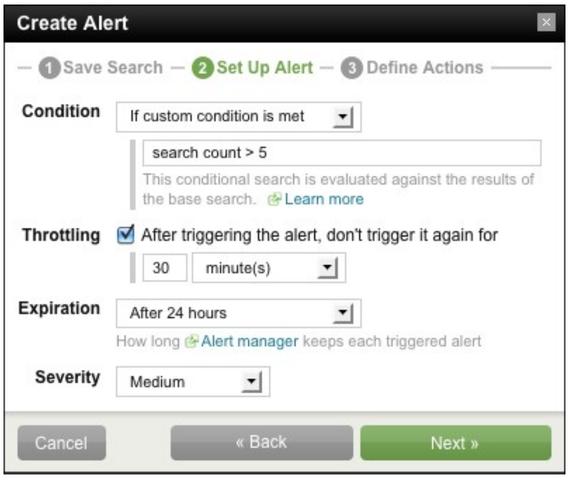
<srcip2>	<destip2>	2
<srcip2>	<destip3>	3

Custom conditional search - Alert when more than 5 events with the same destip-srcip combination are found

Then you set up an advanced conditional custom search that analyzes the results of the base search to find results where the count of any `destip/srcip` combination exceeds 5 in a 60 second timespan.

```
search count > 5
```

This custom search would trigger an alert when the results described in the table above come in, due to that set of 8 events.



Create Alert

1 Save Search — 2 Set Up Alert — 3 Define Actions

Condition: If custom condition is met
search count > 5
This conditional search is evaluated against the results of the base search. Learn more

Throttling: ✓ After triggering the alert, don't trigger it again for
30 minute(s)

Expiration: After 24 hours
How long Alert manager keeps each triggered alert

Severity: Medium

Should it be a real-time or scheduled alert?

You'll probably want to set this up as a real-time search. Scheduled searches can cause performance issues when they run over intervals as short as 1 minute. They can also be a bit inaccurate in situations where, for example, 2 qualifying events come in towards the end of one interval and 4 more come in at the start of the subsequent interval.

To set the alert up as a real-time alert, you would give its base search a start time of *rt-60s* and an end time of *rt*.

You'll likely also want to set up a throttling period for this search to avoid being overwhelmed by alerts. The setting you choose largely depends on how often you want to get notified about alerts of this nature. For example, you might set it up so that once an alert is triggered, it can't be triggered again for a half hour.

Alerting when a set of IDS solutions report a network attack more than 20 times within a 10 minute period

In this example, you're using a set of intrusion detection systems (IDS), such as Snort, to identify attacks against your network. You want to get an alert when three of these IDS technologies report an attack more than 20 times in 10 minutes. After you get the alert, you don't want another one until at least 10 minutes have passed.

Base search - Get the IDS types with more than 20 attack events in a given 10-minute span of time

To begin with, let's assume that you have set things up so that all of your IDS error events are grouped together by the `ids_violation` **source type** and `ids_attack` **event type**, and the IDS type is indicated by the `idstype` field. You can then create a base search for the alert that creates a table that matches each value of `idstype` found with the actual count of events for each ids type. Finally, it filters this table so that only IDS types with 20 or more events are displayed:

```
sourcetype=ids_violation eventtype=ids_attack | stats count by idstype |
search count >= 20
```

This base search gives you a table that could look something like this:

idstype	count
snort	12
ossec	32
fragrouter	27
BASE	26

You can set this base search up as a real-time search or a scheduled search, but either way it needs to capture data over 10 minute interval or window. The **Time Range** for the real-time search would be *rt-10m* to *rt*. The **Time Range** for the scheduled search would be *-10m@m* to *now* and its interval would be "every 10 minutes."

Custom conditional search - Alert when there are 3 or more IDS types with 20+ events in a given 10-minute span of time

Now that you've set up a base search that returns only those IDS types that have more than 20 attack events in a 10-minute span, you need to define a conditional search that goes further by insuring that the alert is only triggered when the results of the base search return 20 or more events *and* include 3 or more IDS types.

To do this, set **Condition** to *if custom condition is met* and then enter this conditional search, which looks at the events returned by the base search and triggers an alert if it finds more than 3 IDS types with 20 or more events each:

```
stats count(idstype) as distinct | search distinct >= 3
```

This custom search would trigger an alert if it evaluated the results represented by the table above, because 3 IDS types reported more than 20 attacks in the 10-minute timespan represented by the base search.

If you're setting this up as a real-time search, set **Throttling** to *10 minutes* to ensure that you don't get additional alerts until at least 10 minutes have passed since the last one. If you're running this as a scheduled search you don't need to set a throttling value since the search will only run on a 10 minute interval to begin with.

Review triggered alerts

Review triggered alerts

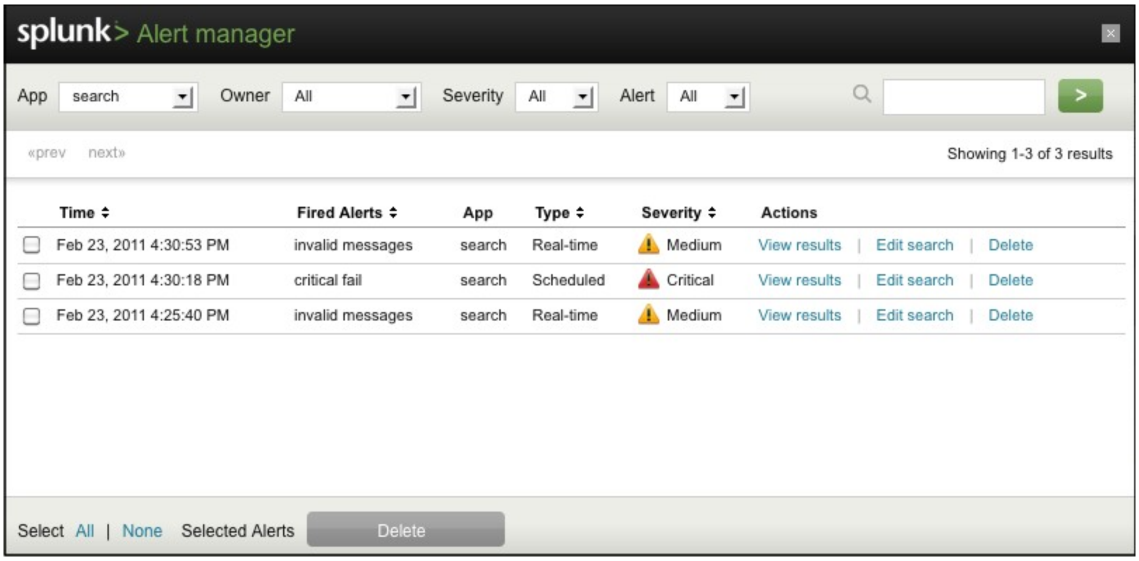
You can review a listing of your recently triggered email **alert** records in the Alert Manager. You open the Alert Manager by clicking the **Alerts** link at the upper right-hand corner of the Splunk UI. It opens as a new window.

The Alert Manager displays records of triggered alerts that have **Tracking** selected in their alert definition. **Tracking** is selected by default for all alerts except those that have a **Condition** setting of *Always*.

When you define an alert, you use the **Alert expiration** setting to determine how long Splunk holds on to the alert records after the alert is triggered. The alert records are removed from the Alert Manager by Splunk according to the **Alert expiration** setting of their parent alert. If the **Alert expiration** setting is *Never* then the alert records generated by that alert remain on the Alerts page until you delete them.

Note: The Alert Manager displays records for triggered alerts that are based on existing saved searches, even if you disable the alerting aspect of those searches after the alerts were triggered. The Alert Manager will not display records of triggered alerts that are based on deleted saved searches, however.

For more information about alerts and alert definition, see "Create an alert" in this manual.



	Time	Fired Alerts	App	Type	Severity	Actions
☐	Feb 23, 2011 4:30:53 PM	invalid messages	search	Real-time	⚠ Medium	View results | Edit search | Delete
☐	Feb 23, 2011 4:30:18 PM	critical fail	search	Scheduled	⚠ Critical	View results | Edit search | Delete
☐	Feb 23, 2011 4:25:40 PM	invalid messages	search	Real-time	⚠ Medium	View results | Edit search | Delete

You can filter the Alert Manager listing by app, alert severity, and alert type. You can also search for specific keywords using the search box. The keyword search applies to fired alert names (which are the same as the names of the searches or reports upon which the alerts are based) and the alert severity (so you can search specifically for alerts of *Critical* severity, if necessary).

Additionally, the Alert Manager enables you to delete individual alert records.

Note that the **Severity** column enables you to quickly spot those alert records that have been given a higher severity level (such as *High* or *Critical*). You define alert severity when you set up or update the alert definition.

Splunk associates each alert record with a saved search artifact that holds the results of the search that triggered the alert. Click the name of a specific alert record to see the results captured in the related search artifact, back in your main Splunk browser window.

Click the name of a specific alert record to see the results captured by that alert back in your main Splunk browser window. This is a **search job** artifact; it won't contain any events that weren't returned by the search job that originally triggered the alert.

For example, say you have a *Firewall breach* alert with an **Alert expiration** setting of *1 day.* If the *Firewall breach* alert is triggered at 3pm, the related alert record will be deleted from the Alert Manager at 3pm the next day.

Setting up tracking when upgrading to 4.2

When you upgrade your Splunk instance to 4.2, be aware that by default existing alerts do NOT show up in the alert manager. To quickly update your existing alerts so that they show up in the alert manager, edit the relevant copy of `savedsearches.conf`. Add `alert.track = true` to the stanzas of each saved search that you have set up as an alert and want to see tracked in the Alert Manager. Review "About configuration files" in the Admin Manual for details about configuration files.

Analyze and Report

About reports and charts

About reports and charts

Using Splunk's powerful IT search capabilities to investigate issues and gather important knowledge about your enterprise is only the first part of the overall equation. Draw on Splunk's ability to swiftly analyze the information you uncover through your searches and use it to create compelling visualizations in the form of reports and charts.

In this chapter you will:

- See a primer on the use of reporting commands.
- Discover how Splunk's report builder makes the definition, generation, and formatting of sophisticated reports a snap.
- Find a handy reference for the different kinds of reports and charts that you can create.
- Discover how to save your reports and share them with others.
- Learn how your enterprise can design dashboard views containing charts that display data critical to your day-to-day business processes.
- Find out how to have PDF printouts of dashboards emailed to interested parties on a regular schedule.

Launching the Report Builder

When you initiate a search, you'll see a **Build report** link in the job status bar above the the timeline. To launch the Report Builder, which you can use to define, generate, and fine-tune the formatting of your report, click this link.

Note: You can start building your report before the search completes. Splunk can dynamically update generated charts as it gathers search results.

Alternatively, you can access the Report Builder by:

- Clicking *Build report* in the **Actions** dropdown menu after you initiate a search.
- Clicking a field in the search results sidebar to bring up the interactive menu for that field. Depending on the type of field you've clicked, you'll see links to reports in the interactive menu such as **average over time**, **maximum value over time**, and **minimum value over time** (if you've selected a numerical field) or **top values over time** and **top values overall** (if you've selected a non-numerical field). Click on one of these links, and Splunk opens the Format report page of the Report Builder, where it generates the chart described by the link.

Note: If your search string includes reporting commands, you access the Report Builder by clicking **Show report**. Splunk will jump you directly to the formatting stage of the report-building process, since your reporting commands have already defined the report.

You don't need to have a strong understanding of reporting commands to use the Report Builder, but if you do have this knowledge the range of things you can do with the Report builder is increased.

Splunk displays the Report Builder in a pop-up window, so you can easily return to the search page and review your search results as you refine your reporting parameters.

To learn more about using the report builder to define basic report parameters, format charts, and export or print finished reports, see "Define reports and generate charts" in this manual.

Note: Keep in mind that report jobs are only preserved in the system for a set period of time. If you do not save them, they eventually expire, and you cannot generate reports for expired report jobs. For more information about job management, see "Supervise your search jobs" in this manual.

If you want to save a report job, select *Save results only* from the **Save** menu on the Report formatting page of the Report Builder.

Use reporting commands

Use reporting commands

You can add reporting commands directly to a search string to help with the production of reports and the summarizing of search results.

A reporting command primer

This subsection covers the major categories of reporting commands and provides examples of how they can be used in a search.

The primary reporting commands are:

- `chart`: used to create charts that can display any **series** of data that you want to plot. You can decide what field is tracked on the x-axis of the chart.
- `timechart`: used to create "trend over time" reports, which means that `_time` is always the x-axis.
- `top`: generates charts that display the most common values of a field.
- `rare`: creates charts that display the least common values of a field.
- `stats`, `eventstats`, and `streamstats`: generate reports that display summary statistics
- `associate`, `correlate`, and `diff`: create reports that enable you to see associations, correlations, and differences between fields in your data.

Note: As you'll see in the following examples, you always place your reporting commands after your search commands, linking them with a **pipe operator** ("|").

`chart`, `timechart`, `stats`, `eventstats`, and `streamstats` are all designed to work in conjunction with statistical functions. The list of available statistical functions includes:

- count, distinct count
- mean, median, mode
- min, max, range, percentiles

* standard deviation, variance
* sum
* first occurrence, last occurrence

To find more information about statistical functions and how they're used, see "Functions for stats, chart, and timechart" in the Search Reference Manual. Some statistical functions only work with the `timechart` command.

Note: All searches with reporting commands generate specific structures of data. The different chart types available in Splunk require these data structures to be set up in particular ways. For example not all searches that enable the generation of bar, column, line, and area charts *also* enable the generation of pie charts. Read the "Chart data structure requirements" subtopic of the "Chart gallery" topic in this manual to learn more.

Creating time-based charts

Use the timechart reporting command to create useful charts that display statistical trends over time, with time plotted on the x-axis of the chart. You can optionally split data by another field, meaning that each distinct value of the "split by" field is a separate series in the chart. Typically these reports are formatted as line or area charts, but they can also be column charts.

For example, this report uses internal Splunk log data to visualize the average indexing thruput (indexing kbps) of Splunk processes over time, broken out by processor:

```
index=_internal "group=thruput" | timechart avg(instantaneous_eps) by
processor
```

Creating charts that are not (necessarily) time-based

Use the chart reporting command to create charts that can display any **series** of data. Unlike the `timechart` command, charts created with the `chart` command use an arbitrary field as the x-axis. You use the `over` keyword to determine what field takes the x-axis.

Note: The `over` keyword is specific to the `chart` command. You won't use it with `timechart`, for example, because the `_time` **default field** is already being used as the x-axis.

For example, the following report uses web access data to show you the average count of unique visitors over each weekday.

```
index=sampledata sourcetype=access* | chart avg(clientip) over date_wday
```

You can optionally split data by another field, meaning that each distinct value of the "split by" field is a separate series in the chart. If your search includes a "split by" clause, place the `over` clause before the "split by" clause.

The following report generates a chart showing the sum of kilobytes processed by each `clientip` within a given timeframe, split by `host`. The finished chart shows the `kb` value taking the y-axis while `clientip` takes the x-axis. The delay value is broken out by host. You might want to use the **Report Builder** to format this report as a stacked bar chart.

```
index=sampledata sourcetype=access* | chart sum(kb) over clientip by host
```

Another example: say you want to create a stacked bar chart that splits out the http and https requests hitting your servers. To do this you would first create `ssl_type`, a search-time **field extraction** that contains the inbound port number or the incoming URL request, assuming that is logged. The finished search would look like this:

```
sourcetype=whatever | chart count over ssl_type
```

Again, you can use the Report Builder to format the results as a stacked bar chart.

Visualizing the highs and lows

Use the top and rare reporting commands to create charts that display the most and least common values.

This set of commands generates a report that sorts through firewall information to show you a list of the top 100 destination ports used by your system:

```
index=sampledata | top limit=100 dst_port
```

This string, on the other hand, utilizes the same set of firewall data to generate a report that shows you the source ports with the lowest number of denials. If you don't specify a limit, the default number of values displayed in a `top` or `rare` is ten.

```
index=sampledata action=Deny | rare src_port
```

A more complex example of the top command

Say you're indexing an alert log from a monitoring system, and you have two fields:

- `msg` is the message, such as `CPU at 100%`.
- `mc_host` is the host that generates the message, such as `log01`.

How do you get a report that displays the top `msg` and the values of `mc_host` that sent them, so you get a table like this:

Messages by mc_host
CPU at 100%
log01
log02
log03
Log File Alert
host02
host56
host11

To do this, set up a search that finds the top `message` per `mc_host` (using `limit=1` to only return one) and then `sort` by the message `count` in descending order:

```
source="mcevent.csv" | top limit=1 msg by mc_host | sort -count
```

Create reports that display summary statistics

Use the stats and eventstats reporting commands to generate reports that display summary statistics related to a field.

To fully utilize the `stats` command, you need to include a "split by" clause. For example, the following report won't provide much information:

```
sourcetype=access_combined | stats avg(kbps)
```

It gives you the average of `kbps` for all events with a sourcetype of `access_combined`--a single value. The resulting column chart contains only one column.

But if you break out the report with a split by field, Splunk generates a report that breaks down the statistics by that field. The following report generates a column chart that sorts through the `access_combined` logs to get the average thruput (kbps), broken out by host:

```
sourcetype=access_combined | stats avg(kbps) by host
```

Here's a slightly more sophisticated example of the `stats` command, in a report that shows you the CPU utilization of Splunk processes sorted in descending order:

```
index=_internal "group=pipeline" | stats sum(cpu_seconds) by processor |
sort sum(cpu_seconds) desc
```

The `eventstats` command works in exactly the same manner as the `stats` command, except that the aggregation results of the command are added inline to each event, and only the aggregations that are pertinent to each event.

You specify the field name for the `eventstats` results by adding the `as` argument. So the first example above could be restated with "avgkbps" being the name of the new field that contains the results of the `eventstats avg(kbps)` operation:

```
sourcetype=access_combined | eventstats avg(kbps) as avgkbps by host
```

When you run this set of commands, Splunk adds a new `avgkbps` field to each `sourcetype=access_combined` event that includes the `kbps` field. The value of `avgkbps` is the average kbps for that event.

In addition, Splunk uses that set of commands to generate a chart displaying the average kbps for all events with a sourcetype of `access_combined`, broken out by host.

Look for associations, statistical correlations, and differences in search results

Use the associate, correlate and diff commands to find associations, similarities and differences among field values in your search results.

The associate reporting command identifies events that are associated with each other through field/field value pairs. For example, if one event has a `referer_domain` of "http://www.google.com/" and another event has a `referer_domain` with the same URL value, then they are associated.

You can "tune" the results gained by the `associate` command with the *supcnt*, *supfreq*, and *improv* arguments. For more information about these arguments see the Associate page in the Search Reference.

For example, this report searches the access sourcetypes and identifies events that share at least three field/field-value pair associations:

```
sourcetype=access* | associate supcnt=3
```

The correlate reporting command calculates the statistical correlation between fields. It uses the `cocur` operation to calculate the percentage of times that two fields exist in the same set of results.

The following report searches across all events where `eventtype=goodaccess`, and calculates the co-occurrence correlation between all of those fields.

```
eventtype=goodaccess | correlate type=cocur
```

Use the diff reporting command to compare the differences between two search results. By default it compares the raw text of the search results you select, unless you use the *attribute* argument to focus on specific field attributes.

For example, this report looks at the 44th and 45th events returned in the search and compares their ip address values:

```
eventtype=goodaccess | diff pos1=44 pos2=45 attribute=ip
```

Real-time reporting

Real-time reporting

You can use Splunk's real-time search to calculate metrics in real-time on large incoming data flows without the use of summary indexing. However, because you are reporting on a live and continuous stream of data, the timeline will update as the events stream in and you can only view the table or chart in preview mode. Also, some search commands will be more applicable (for example, streamstats and rtorder) for use in real-time.

For a primer of reporting commands, see "Use reporting commands" in this manual.

Real-time reporting is accessible through Splunk Web and the CLI. For more information, read "About CLI searches" in the *Search reference manual*.

This feature is discussed in more detail and with examples in the User manual's "Search and Investigate" chapter. Refer to the topic "Real-time search and reporting".

Chart gallery

Chart gallery

You can use Splunk's report builder, combined with Splunk's powerful reporting command language, to generate charts that visualize report data in a number of different ways.

In this topic we discuss the different kinds of charts that Splunk makes available to you and describes some of the situations that each chart type is best suited for.

We also discuss the data structure requirements of the different chart types. For example, we explain why you cannot build a chart using a table that does not include a valid y-axis field, and we explain why some searches enable you to generate bar, column, line, and area charts, but not pie charts.

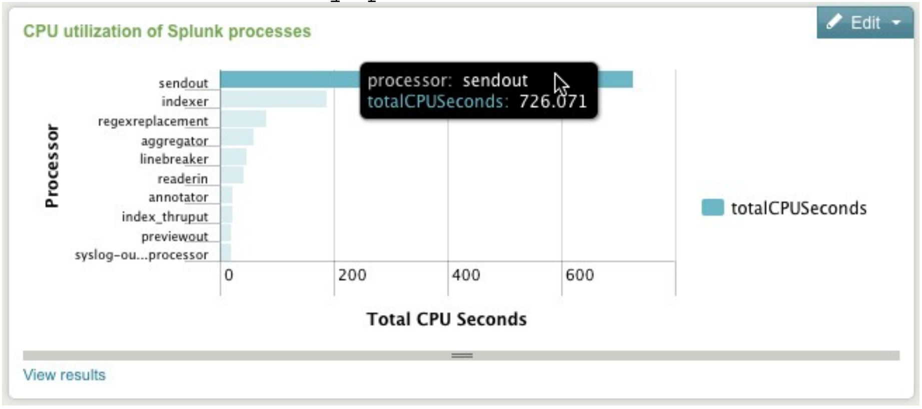
For more information about the report builder, see "Define reports and generate charts" in this manual.

Column and bar charts

Use a column chart or bar chart to compare the frequency of values of fields in your data. In a column chart, the x-axis values are typically field values (or time) and the y-axis can be any other field value, count of values, or statistical calculation of a field value. Bar charts are exactly the same, except that the x-axis and y-axis values are reversed.

The following bar chart presents the results of this search, which uses internal Splunk metrics. It finds the total sum of CPU_seconds by processor, and then arranges the processors with the top ten sums in descending order:

```
index=_internal "group=pipeline" | stats sum(cpu_seconds) as
totalCPUSeconds by processor | sort 10 totalCPUSeconds desc
```

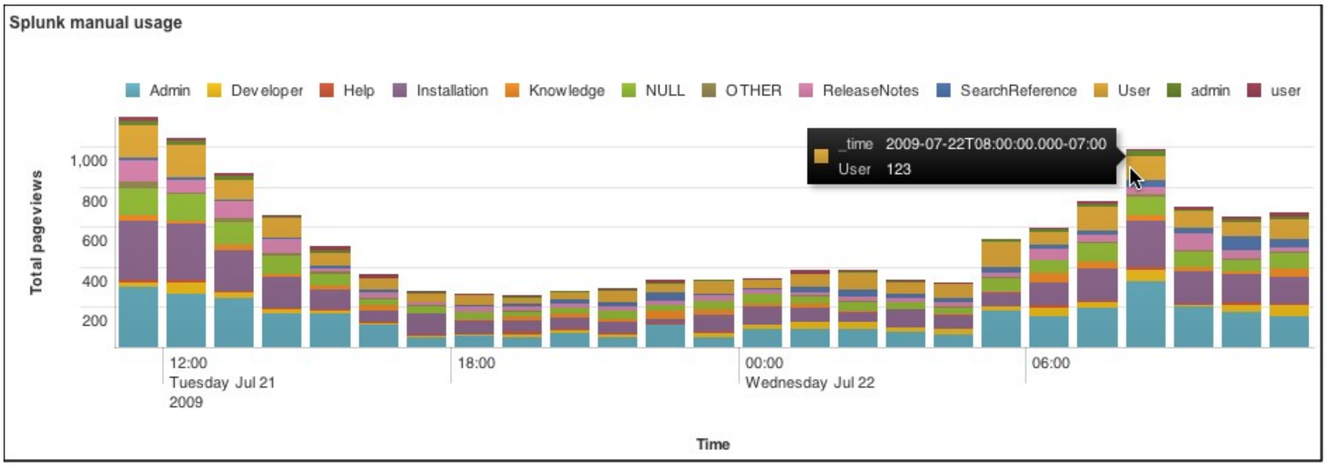


Stacked column and bar charts

You can also use stacked column charts and stacked bar charts to compare the frequency of values of fields in your data. A stacked column chart is the same as a regular column chart, except all of the columns are segments of a single column. The value of the total column is the sum of the segments.

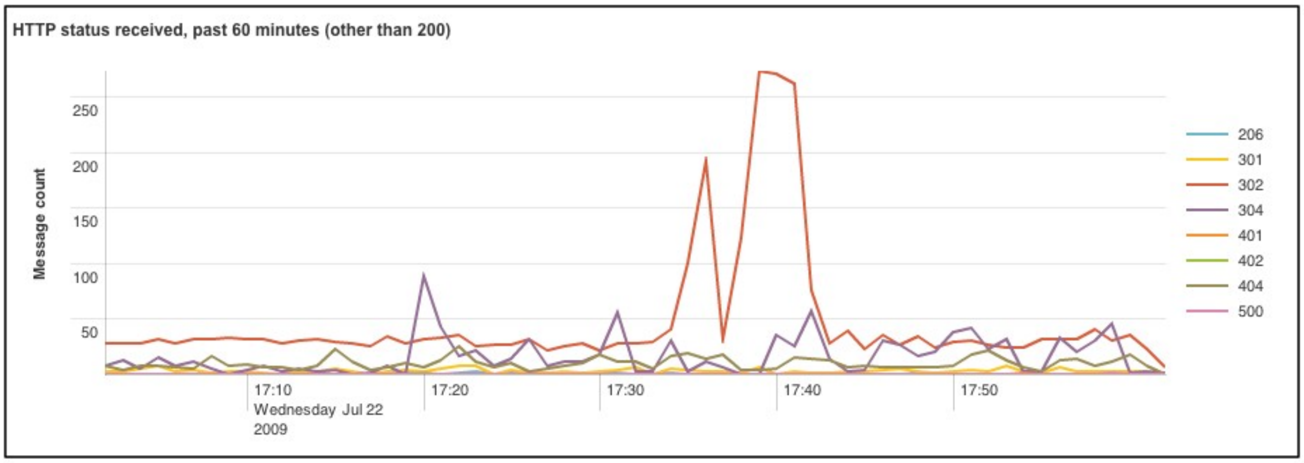
Note: You can use a stacked column or bar chart to highlight the relative weight (importance) of data within a set of your data.

The following chart illustrates the usage of Splunk manuals shortly after the release of Splunk 4. Each column segment represents the number of page views for a particular manual in a 10-minute-long slice of time:



Line chart

Use a line chart to show trends in your data over time or another field. You can display more than one series in a line chart.

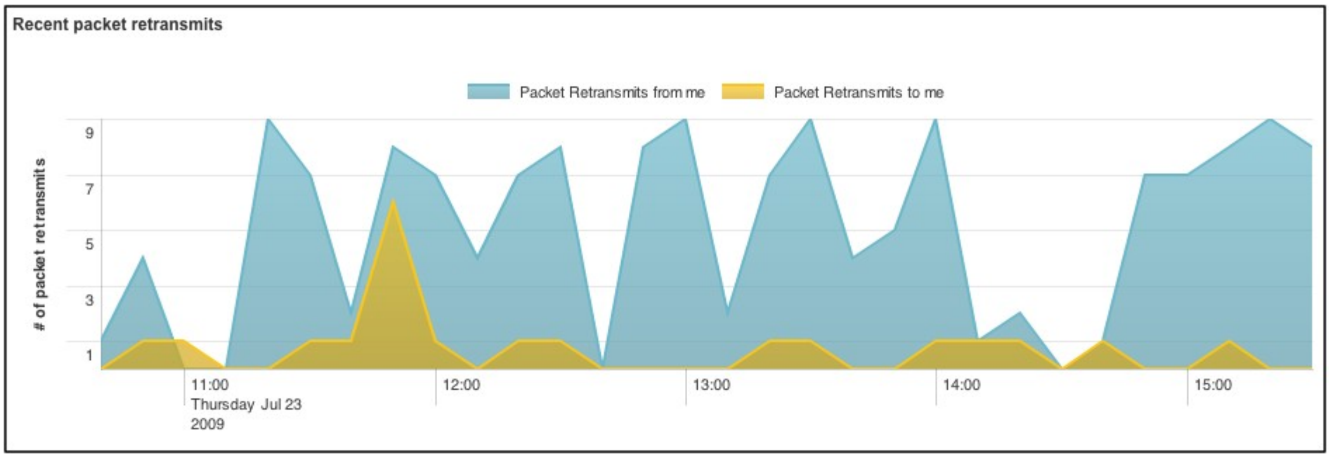


Area chart

Use an area chart to display trends in your data either over time or in comparison to another field value. The shaded areas under the data points help emphasize quantities.

The following area chart is derived from this search:

```
sourcetype="tcptrace" | search host1_rexmt_data_pkts>0 OR
host2_rexmt_data_pkts>0 | timechart
max(host1_rexmt_data_pkts),max(host2_rexmt_data_pkts) | fillnull value=0 |
rename max(host1_rexmt_data_pkts) as "Packet Retransmits from
me",max(host2_rexmt_data_pkts) as "Packet Retransmits to me"
```

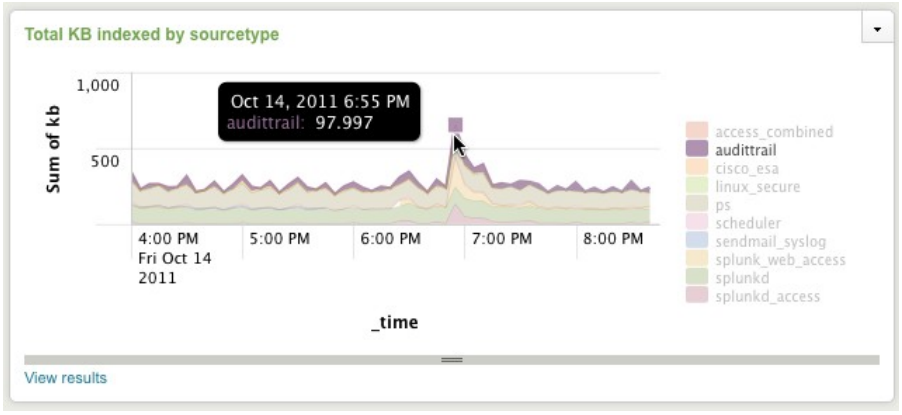


Stacked area chart

Use a stacked area chart to show multiple series among the trends in your data the way an area chart can. A stacked area chart shows how each data series relates to the entire set of data as a whole.

The following chart is another example of a chart that presents information from internal Splunk metrics. The search used to create it is:

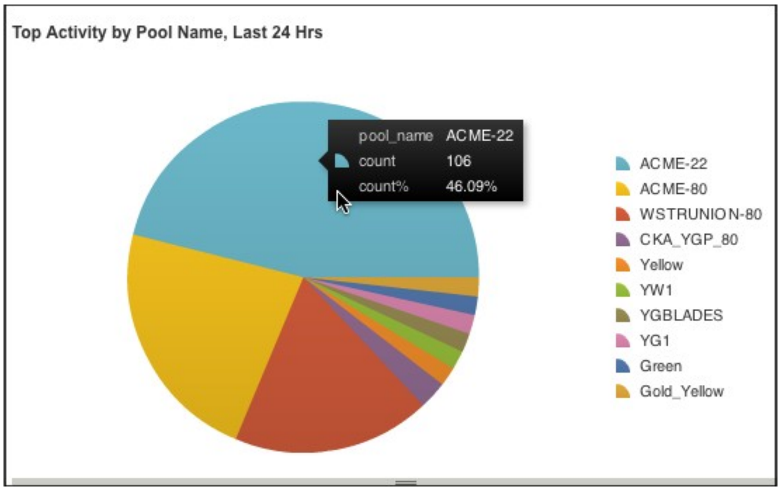
```
index=_internal per_sourcetype_thruput | timechart sum(kb) by series
useother=f
```



Pie chart

Use a pie chart to show the relationship of parts of your data to the entire set of data as a whole. The size of a slice in a pie graph is determined by the size of a value of part of your data as a percentage of the total of all values.

The following pie chart presents the network traffic "pools" with the most activity over the past 24 hours. Note that you can get metrics for individual pie chart wedges by mousing over them.

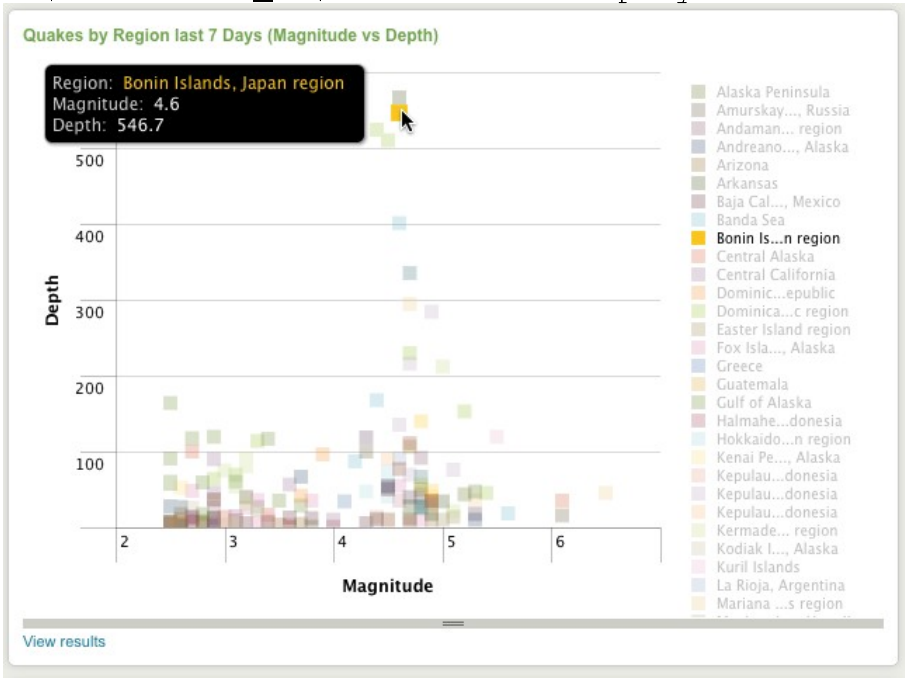


Scatter chart

Use a scatter chart (or "scatter plot") to show trends in the relationships between discrete values of your data. Generally, a scatter plot shows discrete values that do not occur at regular intervals or belong to a series. This is different from a line graph, which usually plots a regular series of points.

Here's an example of a search that can be used to generate a scatter chart. It finds all of the packets received from various client IP addresses and then orders them according to the number of bytes in each packet.

```
* | fields - _* | fields clientip bytes
```



For more information about the search used to generate this scatter chart, and about the data structures that scatter charts require, see the "Chart data structure requirements" subtopic, below.

Bubble chart

Use a bubble chart to show trends and the relative importance of discrete values in your data.

The size of a bubble indicates a value's relative importance. It represents a third dimension on top of the x-axis and y-axis values that plot the bubble's position on the chart. This dimension determines the bubble's size relative to the others in the chart.

Note: Bubble charting via the Report Builder is disabled in the current version of Splunk. However, you can set up bubble charts in dashboards using Splunk's view XML. For more information see the Custom charting configuration reference chapter in the Developer manual.

Gauge

The gauge chart types enable you to see a single numerical value mapped against a range of colors that may have particular business meaning or business logic. As the value changes over time, the gauge marker changes position within this range. Gauges are designed to provide an especially dynamic visualization for **real-time searches**, where the value returned fluctuates as events are returned, causing the gauge marker to visibly bounce back and forth within the range as you watch it.

You can define the overall numerical range represented by the gauge, and you can define the size of the colored bands within that range. By default you can set three bands of green, yellow, and red, displayed in that order. For example, if your range extends from 1 to 100, you could have the 0-59 portion of the range be green, the 60-84 portion of the band be yellow, and the 85-100 portion of the range be red.

The easiest way to set the range is by defining a search that uses the new gauge search command. The `gauge` command enables you to indicate the field whose value will be tracked by the gauge. If you want to use the color bands, you can add four "range values" to the search string that indicate the beginning and end of the range as well as the relative sizes of the color bands within it.

For example, to set up a gauge that tracks a `hitcount` field value with the ranges mentioned above, you might use the following search string:

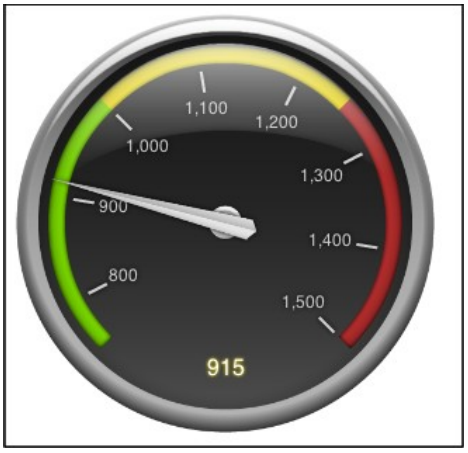
```
...| gauge hitcount 0 60 85 100
```

There are three types of gauges that you can choose from: radial, filler, and marker.

Note: The gauge chart type is similar in behavior to the "single value" dashboard panel type in that both visualizations are designed to be used in conjunction with searches that return a single numerical result and associate color codes with the number returned.

Radial gauge

The radial gauge type looks essentially like a speedometer or pressure valve gauge. It has an arced range scale and a rotating needle. The current value of the needle is displayed at the bottom of the gauge (in the case of the example below, the value is *915*). If the value ever falls outside of the top range of the gauge, the needle "flutters" at the boundary and a warning icon appears.

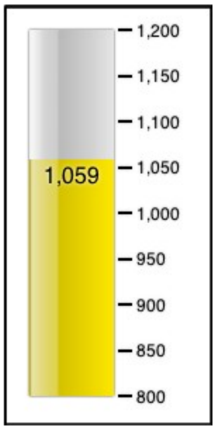


This radial gauge example represents this simple search, which is run in real time with a one-minute window:

```
index=_internal | stats count as myCount | gauge myCount 750 1000 1250
1500
```

Filler gauge

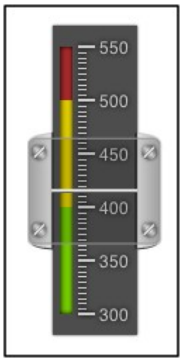
The filler gauge is similar in appearance to a thermometer, with a liquid-like filler indicator that changes color as it rises and passes gauge boundaries. So if you have set up three boundaries, the liquid will appear to be green when it is near the bottom, yellow when it reaches the midpoint boundary, and red when it gets to the top. The current value of the gauge fill is displayed at the left side of the filler indicator. If the value ever falls outside of the upper or lower ranges of the filler gauge, a warning icon appears.



The filler gauge is oriented vertically by default but can be oriented horizontally through custom charting configuration.

Marker gauge

The marker gauge is a linear version of the filler gauge. It is already "filled"; a gauge marker rests at the value returned by the search. If the gauge is displaying the results of a real-time search, the marker can appear to slide back and forth across the range as the returned value fluctuates over time. If the returned value falls outside of the upper or lower ranges of the marker gauge, the marker appears to vibrate at the upper or lower boundary and a warning icon appears.



The marker gauge is oriented vertically by default but can be oriented horizontally through custom charting configuration.

Marker gauges have display issues with numbers exceeding 3 digits in length. To manage this, you can set up a search that divides a large number by a factor that reduces it to a smaller number. For example, if the value returned is typically in the tens of thousands, set your search up so the result is divided by 1000. Then a result of 19,100 becomes 19.1.

You can also deal with large numbers by setting the chart configuration options so the range is expressed as a percentage. For more about that, see the next subsection.

Changing the gauge display defaults

You can change the default display mode of the three gauge types in dashboard panels by setting up custom charting configurations in the xml behind the panel. For details, see the "Custom charting configuration reference" chapter in the Developer Manual (or, more specifically, the "Chart and legend properties" topic in that same manual).

Among the chart configuration properties available for gauges, there are three that we think most users will want to be aware of. They apply to all three gauge types.

- `style` - Controls the overall appearance of the gauge. There are two possible values: `shiny` and `minimal`. The shiny style is the default and presents the gauge with chrome, shading and other graphic qualities that cause it to mimic the appearance of real-world gauges. The minimal style is a stripped-down, "back-to-basics" version of the gauge.
- `rangeValues` - Represents the overall numerical range represented by the gauge, and the relative size of the color-coded subranges within that overall range. For example, a range of `[O,30,70,100]` would indicate that the gauge starts at zero, ends at `100` and has three subranges that are each identified by a different color. **It is important to note** that when you specify range values in the xml, they override range values that are specified through the search upon which the dashboard panel is based.
- `gaugeColors` - Specifies a list of hexadecimal color values from which the range band colors are generated. Colors display in the order indicated in the array. You can specify any number of colors. If your gauge has more or less range intervals (specified either through the underlying search language or via the `rangeValues` parameter) Splunk will interpolate the colors as necessary. The default `gaugeColors` are green-yellow-red (`[0x84E900,0xFFE800,0xBF3030]`).

Here's an example of xml code for a chart where the minimal style is selected, the `rangeValues` are defined, and the `gaugeColors` are set up to display in reverse order (red-yellow-green):

```
<param name="charting.chart">radialGauge</param>
<param name="charting.chart.style">minimal</param>
<param name="charting.chart.rangeValues">[0,30,70,100]</param>
<param name="charting.gaugeColors">[0xBF3030,0xFFE800,0x84E900]</param>
```

You can also:

- arrange to have filler and marker gauges appear in a horizontal orientation by setting `<param name="charting.chart.orientation">x</param>`.
- have a gauge in a dashboard panel display its range as a percentage. This is one way you can manage ranges that are over four digits in size. The parameters that affect this are `usePercentageRange` and `usePercentageValue`; set them to `true` to format gauge and range values as percentages.

Other chart types

Splunk enables the creation of other chart types in dashboard panels using Splunk's view XML (you cannot currently use the Report Builder to generate them). These chart types include:

- Histograms
- Range marker charts
- Ratio bar charts
- Value marker charts

For more information about these chart types, the data structures required to support them, and their view XML properties, see the Custom charting configuration reference chapter in the Developer manual.

Chart data structure requirements

Each chart type requires specific structures of data in order to create meaningful charts. This subsection covers these different data structures, and explains why some searches allow you to generate bar and column charts, but not pie charts.

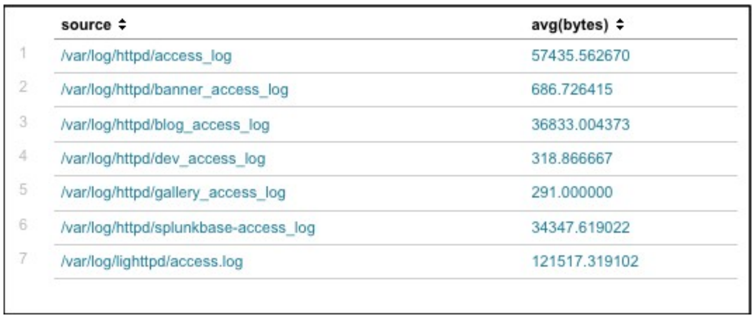
Column, line, and area charts

It's important to understand that column, line, and area charts are two-dimensional charts supporting one or more **series**. They plot data on a Cartesian coordinate system, working off of tables that have at least two columns, where the first column contains x-axis values and the subsequent columns contain y-axis values (each column represents a series). This is why "Values over time" searches and searches that include splitbys are among those that are available as column, line, and area charts.

If you want to generate a column, line, or area chart from a search, that search must produce a table matching the description provided in the preceding paragraph. For example, any search using the `timechart` reporting command will generate a table where `_time` is the first column (and therefore the x-axis of any column, line, or area chart generated from those results). You'll get the same result with most basic searches involving reporting commands.

For example, a search like this, where the `over` operator indicates that `source` is the x-axis:

```
...| chart avg(bytes) over source
```
produces a two-column, single-series table like this:

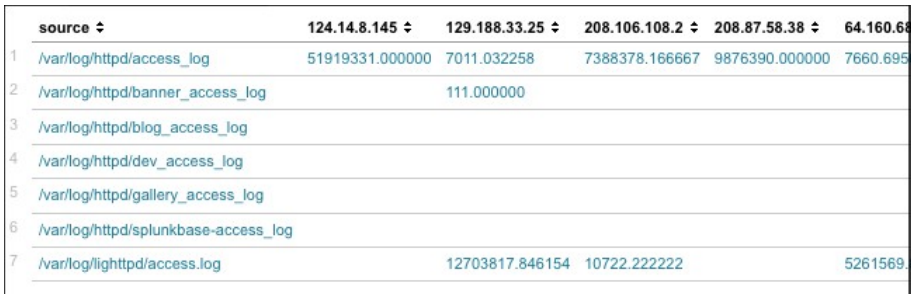


	source	avg(bytes)
1	/var/log/httpd/access_log	57435.562670
2	/var/log/httpd/banner_access_log	686.726415
3	/var/log/httpd/blog_access_log	36833.004373
4	/var/log/httpd/dev_access_log	318.866667
5	/var/log/httpd/gallery_access_log	291.000000
6	/var/log/httpd/splunkbase-access_log	34347.619022
7	/var/log/lighttpd/access.log	121517.319102

In this table, the x-axis is `source`, and the y-axis is `avg(bytes)`. With it you can produce a column chart that compares the average number of bytes passed through each source.

Say you change up the search a bit by adding `clientip` as a splitby field:

```
...| chart avg(bytes) over source by clientip
```
This produces a table that features multiple series:



	source	124.14.8.145	129.188.33.25	208.106.108.2	208.87.58.38	64.160.68
1	/var/log/httpd/access_log	51919331.000000	7011.032258	7388378.166667	9876390.000000	7660.695
2	/var/log/httpd/banner_access_log		111.000000			
3	/var/log/httpd/blog_access_log					
4	/var/log/httpd/dev_access_log					
5	/var/log/httpd/gallery_access_log					
6	/var/log/httpd/splunkbase-access_log					
7	/var/log/lighttpd/access.log		12703817.846154	10722.222222		5261569.

In this table, the x-axis is still `source`, and the y-axis is still `avg(bytes)`, but it now breaks out the `avg(bytes)` by `clientip`, creating a table with multiple series. You might generate a *stacked* column chart to represent this data.

You run into trouble when you design a complex search that returns a result table that lacks a valid x-axis or y-axis value. This can happen when you use the `eval` and `fields` commands to force a particular arrangement of columns in the finished table, for example.

Bar charts

Bar charts have the same data structure requirements as column, line, and area charts, except that the x- and y-axes are reversed. So they are working off of tables that have at least two columns, where the first column contains y-axis values and the subsequent columns contain x-axis values.

Pie charts

Pie charts are one dimensional and only support a single **series**. They work off of tables with just two columns, where the first column contains the labels for each slice of the pie, and the second column contains numerical values that correspond to each label, determining the relative size of each slice. If the table generated by the search contains additional columns, those extra columns have no meaning in the terms of the pie chart and are ignored.

Of the two "column, line, and area charts" search examples noted above, the first is the only one that could be used to make a pie chart. The `source` column would provide the wedge labels, and the `avg(bytes)` column would provide the relative sizes of the wedges (as percentages of the sum of `avg(bytes)` returned by the search).

Scatter charts

Scatter charts are cartesian charts that render data as scattered markers. They help you visualize situations where you may have multiple y-axis values for each x-axis value, even when you're not charting multiple series. Their data set can be in one of two forms:

- A **single series** setup, where the chart is structured on a 2-column data table, where the first column (column 0) contains the values to be plotted on the x-axis, and the second column (column 1) contains the values to be plotted on the y-axis.
- A **multiple series** setup, where the chart is structured on a data table that contains 3 columns. The first column (column 0) contains the series names, and the next two columns contain the values to be plotted on the x- and y-axes, respectively.

To generate a scatter chart you need to graph events directly with a search like:

```
* | fields - _* | fields clientip bytes
```

This search finds all of the packets received from various client IP addresses and then orders them according to the number of bytes in each packet.

- Note that the search removes all fields with a leading underscore, such as the `_time` field.
- The second `fields` command isolates the two fields that you want for the x- and y-axis of the chart, respectively. The y-axis value should be numerical for best results. (So in this case, the x-axis is `clientip` while the y-axis is `bytes`.)

Note: To create a scatter plot chart with a search like this, you need to enter the reporting commands directly into the Report Builder by clicking **Define report data using search language** in the Report Builder. You can run this report from the search bar, but when you open up Report Builder, it adds a timechart command that you should remove before formatting the report.

More complex scatter charts can be set up in dashboards using Splunk's view XML. For more information see the Custom charting configuration reference chapter in the Developer manual.

Gauges

A gauge is a visualization of a search that returns a single numerical field value. It shows where this value exists within a range defined in the search language using the `gauge` command. A simple example is a search that returns a count of the number of events matching a set of search criteria that come in within a specific time period, or a real-time window, if you are using a real-time search. If you base a gauge on a real-time search, the chart's range marker will appear to fluctuate as the value displayed within the real-time search window changes over time.

Familiarize yourself with the data structure requirements for the different visualization types, to better understand how to design searches for them.

Familiarize yourself with the data structure requirements for the different visualization types, to better understand how to design searches for them.

Understand basic table and chart drilldown actions

Understand basic table and chart drilldown actions

Splunk's table and chart drilldown actions enable you to delve deeper into the details of the information presented to you in tables and charts. With a simple click on a table row or a bar in a bar chart, you can kick off searches that drill down to provide more information about those discrete selections.

This topic provides some examples of this functionality as configured with simple XML via the visual dashboard designer. It also briefly goes over some of the drilldown functionality that can be configured through the advanced XML.

To learn how to enable or update basic chart and table drilldown functionality for panels in dashboards that have been created using the visual dashboard editor, see "Create simple dashboards with the visual dashboard editor" in this manual.

Clicks on drilldown-enabled dashboard panels usually result in the drilldown search opening in the current browser window. However, if you hold down the **Crtl** key (**Command** for Macs) when you click, Splunk opens a separate window for the search.

Note: Drilldown functionality does not work for simple XML dashboard panels that are based on searches of summary indexes. To set up drilldown for dashboards utilizing these types of searches, you need to perform "custom wiring" with the advanced XML. For more information, see "Advanced drilldown behavior" in this topic.

Basic data table drilldown functionality

As we explain in "Create simple dashboards with the visual dashboard editor," you have three basic drilldown options when you define data table panel types with the visual dashboard editor:

- *None* - Drilldown functionality is turned off.
- *Row* - A click on a row launches a drilldown search on the x-axis value (the value in the first column of the table) for that row.
- *Cell* - A click on a cell launches a drilldown search on both the x-axis and y-axis values represented in that cell.

In general, when the search involved in the creation of the original table uses **transforming commands**, the drilldown wipes out the final transforming command and replaces it with arguments that drill down on the specific x-axis value or x- and y-axis value combination caught by the click. See the subsections below for examples of how this works.

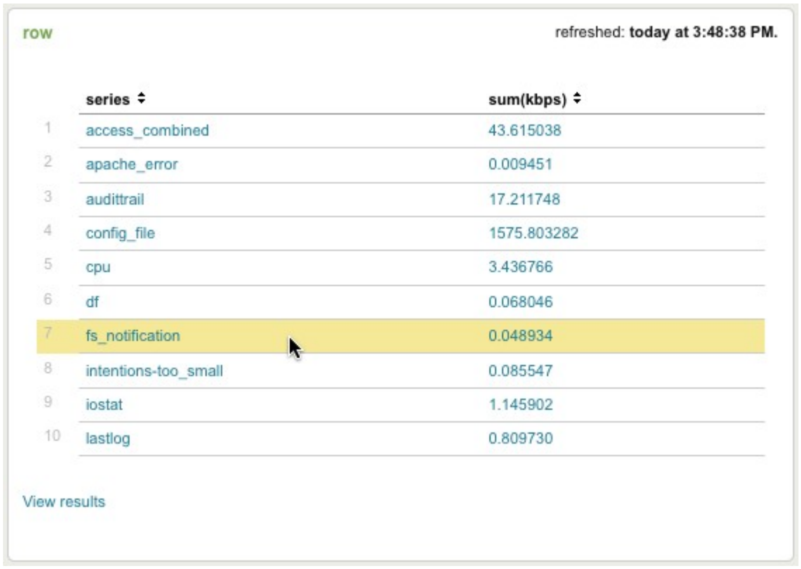
Row drilldown

When a dashboard data table has a **Drilldown** value of *Row*, you can initiate drilldown searches along whole rows by clicking on them.

Imagine that you have a dashboard data table panel that is based on this search:

```
index="_internal" group="per_sourcetype_thruput" | chart sum(kbps) over
series
```

In this table, a "row click" drilldown search would concentrate on the x-axis value of the selected row, which in this case would be a value of the *series* field, such as `fs_notification`:



	series	sum(kbps)
1	access_combined	43.615038
2	apache_error	0.009451
3	audittrail	17.211748
4	config_file	1575.803282
5	cpu	3.436766
6	df	0.068046
7	fs_notification	0.048934
8	intentions-too_small	0.085547
9	iostat	1.145902
10	lastlog	0.809730

This click sets off the following search in the Search app, which finds six results:

```
index="_internal" group="per_sourcetype_thruput" series="fs_notification"
```

Note that the drilldown search is basically the same as the original search, except that the **transforming command** has been removed and replaced with a drilldown search term of `series="fs_notification"`.

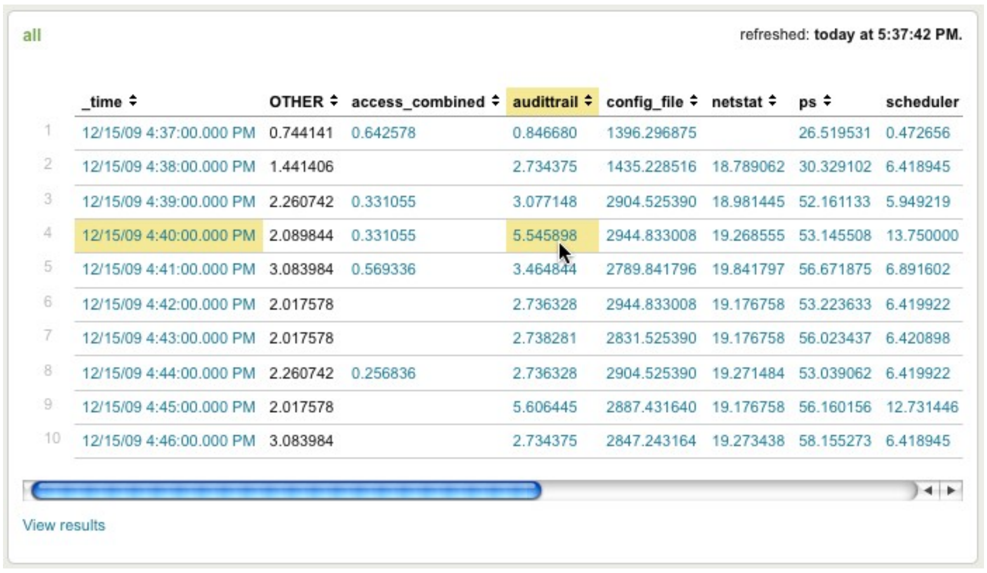
Cell drilldown

When a dashboard data table has a **Drilldown** value of *Cell*, you can initiate drilldown searches for specific cells by clicking on them.

Say you have a table generated by the following search:

```
index="_internal" source="*metrics.log" group="per_sourcetype_thruput" |
timechart sum(kb) by series
```

In this table, a "cell click" drilldown search would concentrate on a combination of the x-axis value (the value in the first column for the cell's row) and the y-axis value (the value of the cell's column).


all

refreshed: **today at 5:37:42 PM.**

	_time	OTHER	access_combined	audittrail	config_file	netstat	ps	scheduler
1	12/15/09 4:37:00.000 PM	0.744141	0.642578	0.846680	1396.296875		26.519531	0.472656
2	12/15/09 4:38:00.000 PM	1.441406		2.734375	1435.228516	18.789062	30.329102	6.418945
3	12/15/09 4:39:00.000 PM	2.260742	0.331055	3.077148	2904.525390	18.981445	52.161133	5.949219
4	12/15/09 4:40:00.000 PM	2.089844	0.331055	5.545898	2944.833008	19.268555	53.145508	13.750000
5	12/15/09 4:41:00.000 PM	3.083984	0.569336	3.464844	2789.841796	19.841797	56.671875	6.891602
6	12/15/09 4:42:00.000 PM	2.017578		2.736328	2944.833008	19.176758	53.223633	6.419922
7	12/15/09 4:43:00.000 PM	2.017578		2.738281	2831.525390	19.176758	56.023437	6.420898
8	12/15/09 4:44:00.000 PM	2.260742	0.256836	2.736328	2904.525390	19.271484	53.039062	6.419922
9	12/15/09 4:45:00.000 PM	2.017578		5.606445	2887.431640	19.176758	56.160156	12.731446
10	12/15/09 4:46:00.000 PM	3.083984		2.734375	2847.243164	19.273438	58.155273	6.418945

View results

In this example, the clicked on cell initiates the following drilldown search over the 4:40:00pm to 4:40:59pm time range on 12/15/09 (the x-axis value) and adds a focus on the `audittrtail` value of the **series** field (the y-axis value):

```
index="_internal" source="*metrics.log" group="per_sourcetype_thruput"
series="audittrail"
```

Note that this drilldown search removes the last **transforming command** from the originating search.

Note: The y-axis value will not come into play in all cell drilldown searches. Cell-click interactions are designed to work with tables and charts generated by searches containing a "split by" clause. Cell clicks in charts based on reporting commands like `timechart max(eps) min(eps) avg(eps)` will always behave like row clicks. Such tables should always be configured for row-click drilldown; this approach is less confusing for users of the table.

Basic chart drilldown functionality

As we explain in "Create simple dashboards with the visual dashboard editor," you have two basic drilldown options when you define chart panel types with the visual dashboard editor:

- *Off* - Drilldown functionality is turned off.
- *On* - A click on a portion of a chart launches a drilldown search into the values that that portion of the chart represents.

In general, when the search involved in the creation of the original table uses **transforming commands**, the drilldown wipes out the final transforming command and replaces it with arguments that drill down on the specific x-axis value or x- and y-axis value combination caught by the click. See the subsections below for examples of how this works.

Bar, column, line, and area chart drilldown

Drilldown searches on dashboard bar, column, line, and area charts behave differently depending on whether you click in the body of the chart or in the chart legend, if a legend is displayed.

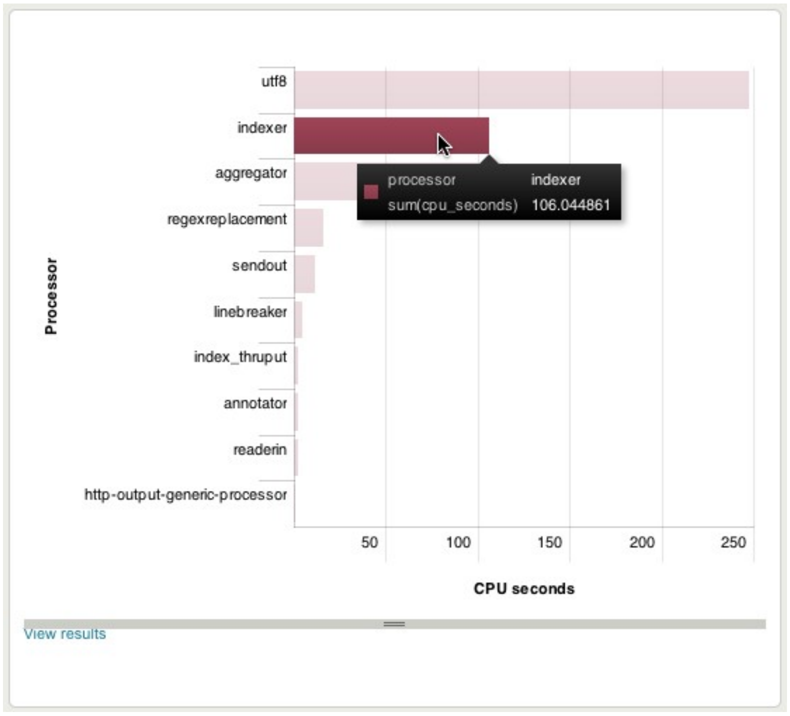
In general, no matter what you click in the body of a row, column, line, or area chart, Splunk creates a drilldown search that:

- duplicates the search that originated the chart, except with the final transforming commands removed.
- adds a new search term based on the "x-axis" value that you select in the chart.
- possibly adds a "y-axis" value, depending on whether a meaningful y-axis value exists in the originating search. For example, most split-by values work as "y-axis" values. But things like `avg(eps)` will not.

Say you have a bar chart based on the following search:

```
index="_internal" source="*metrics.log" group="pipeline" | chart
sum(cpu_seconds) over processor | sort 10 - sum(cpu_seconds)
```

In this chart, the x-axis is the `processor` value, while the y-axis is the `cpu_seconds` sum over the given time range (the last 60 minutes).



If you click in the body of this chart, the drilldown search drills down on the x-axis value represented by that bar:

```
index="_internal" source="*metrics.log" group="pipeline"
processor="indexer"
```

Note that the drilldown search is identical to the original search except that the final set of transforming commands has been removed and a focus has been added on the `aggregator` value of `processor`.

Drilldown searches on legend items are different. Drilldown searches for chart legends only work when there is a split-by (or y-axis) field in the chart. For example, legend items for a line chart based on `timechart avg(eps) by series` are values of `series`, such as `audittrail`. A click on the `audittrail` item results in a drilldown search in which `series=audittrail` is added to the originating search. Legend item drilldown searches always run over the same time range as the originating search.

Note: Sometimes the legend element is something that can't really be drilled down into, like `avg(eps)`. Clicks on such legend items return an error message.

Pie chart drilldown

Pie charts provide identical drilldown behavior whether you click in the body of the chart--a pie slice, in other words--or the label pointing to that slice. Either way, the drilldown focuses on the value represented by the slice or label you click on.

So if the pie chart displays the top processors being utilized over the past 24 hours, and you click on the chart portion or legend item representing the `indexer` processor, then the drilldown search will be the same as the original, only with the transforming command removed and `processor=indexer` added. You'll get the same result if you click on the `indexer` label.

Advanced drilldown behavior

The default table and chart drilldown functionality that you can get out of dashboards created with simple XML is just the start. When you create dashboards using advanced XML, you have a range of table/chart drilldown customization options that can greatly enhance your users' dashboard experience.

For example, you can set up dashboards that:

- Open the search in a view other than the default "flash timeline" search view.
- Have a drilldown click open up a new table or chart beneath the initial panel. Click on a table cell, and see a line chart open up underneath that table that displays the drilldown results.
- Include a nested series of drilldown searches. A click in a bar chart opens a table. A click in that table opens a line chart. And click in that line chart opens a search in a separate window.
- Launch a different search than the search that generates the data in the table or chart. For example, if you've built many charts and tables on searches of a particular summary index, you might want to send your users to a search that isn't based on that summary index.

For more information about setting up advanced drilldown actions like the ones described above, see "How to customize drilldown options" in the Developer manual.

Define reports and generate charts

Define reports and generate charts

Splunk's Report Builder makes it easy to generate sophisticated reports using the results from any completed or finalized search. It offers a wide range of reporting options, both in terms of reporting parameters and chart types.

With the Report Builder, you don't need to have an advanced understanding of reporting commands like `stats`, `top`, `chart`, and `timechart` in order to build robust, information-rich reports. However, you can still use these commands in your search if you're more comfortable with them.

For examples of how reporting commands are used, see "Use reporting commands" in this manual.

The Report Builder is broken up into two stages: Define Report Contents and Format Report. Use the Define Report Contents page to set up your initial report parameters, such as the type of report and the fields that you're reporting on.

Once you've defined these initial details, you can go to the Format Report page, where Splunk generates the chart and corresponding table. On this page you can fine-tune the chart formatting, review the related table, and save, print, and export the results.

If you're not sure how to launch the Report Builder, see "Launching the Report Builder" in this manual.

Define report contents

The Define Report Contents page gives you the freedom to define your report parameters in the manner that feels most comfortable to you. If you're familiar with reporting commands and want to define your report contents using sophisticated search language, you can do that.

But you should use the default form-based mode for report content definition if you:

- are not that familiar with reporting commands.
- want to quickly and efficiently set up a report using drop-down lists and don't necessarily know what fields you can report on.

Both modes of the Define Report Contents page display a search bar with your search preloaded into it, and a time range picker list that lets you change the report time-range.

Note: If you use the time range picker to change the time range for your report, take care to choose a range of time that includes the fields on which you plan to report.

Set up report contents using the form-based mode

The form-based mode of the Define report content page helps you quickly set up your reporting parameters through a set of list fields. In this mode, you cannot manually update the language in the search bar, but as you use the form to set up your reporting parameters you'll see that the search bar automatically updates with equivalent reporting commands.

There are three basic report types to choose from:

- ***Values over time*** for reports that display trends in field values over a selected time range. These reports use the `timechart` reporting command. They can display as a bar, column, line, or area chart.

- ***Top values*** for reports that display the most common field values to turn up in a search. These reports use the `top` command, and can display as a bar, column, or pie chart.
- ***Rare values*** for reports that display the most uncommon field values to turn up in a search. These reports use the `rare` command, and can display as a bar, column, or pie chart.

Note: The grayed-out ***Distribution of values*** and ***Original values*** report types are coming in a future Splunk release. They'll handle reports that you can currently build with the Report Builder if you define your report directly using reporting commands, such as `chart`.

If you choose **Values over time** you can define reports that involve multiple field series or split-by fields. These report types also let you define the time span for each bin.

After you define your **Report Type** you can select the fields that you want to report on. If you've chosen a *Values over time* report type you'll also associate a statistical function (such as *count*, *direct count*, *average*, *mode*, *median*, and so on) with your primary field. (For more information about statistical functions and how they're used, see "Functions for stats, chart, and timechart" in the Search Reference Manual.

Once you have your initial report parameters set up, click **Next Step: Format Report**. Splunk takes you to the Format report page of the Report Builder, where it generates a version of the report using default formatting parameters.

Note: At any point during your use of the form interface you can switch over to the search language mode, to refine the reporting commands that have been appearing there. For example, say you set a **Report Type** of *Top Values* with a **Fields** value of *Host*. As you select these values, this search appears in the search box:

```
... | top host limit=1000
```
Splunk's default limit for a top report built through the Report Builder is 1000, which means that Splunk captures the top thousand items found in the search in the resulting table and report. If you're dealing with a search that is bringing back a large number of results, you can change this default by going into search language entry mode (see below) and manually changing the limit to a value that better fits your needs (such as `limit=20`).

Set up report contents using search language

If you're on the Define report content page of the Report Builder and you want to manually define the reporting language for your report, use the search language entry mode for that page. Click **Define report using search language** to enter this mode.

When you are in the search language entry mode, you can enter reporting commands directly into the search bar, with the freedom to make them as simple or sophisticated as your situation requires.

For examples of how reporting commands are used, see "Use reporting commands" in this manual.

Note: If you include reporting commands in your *initial* search, the **Show report** button that appears takes you straight to the Format report page of the Report Builder, bypassing the 'Define report content page entirely.

As in the form-based mode, once you have your initial report parameters set up, click **Next Step: Format Report**. Splunk takes you to the Format report page of the Report Builder, where it generates a version of the report using default formatting parameters.

Format reports

The Format report page enables you to fine-tune the default formatting of your report. The report is broken up into two major sections:

- the **Chart** section, which displays your report results as a chart.
- the **Table** section, which displays your report results as a table.

When Splunk opens the Format report page, it generates a chart using default reporting parameters that are associated with the report type, as well as the statistical operators involved in the search. For example, if on the Define Report Contents page you chose a **Report type** of *Trend over time* and use a *count* or *distinct count* statistical operator, Splunk renders it as a column chart by default. (If you use a different statistical operator, such as *average*, Splunk renders a line chart instead.)

Note: If you have a search that includes reporting commands and you want the chart that is generated from that search to include custom formatting (such as a pie chart in place of the default bar chart) be sure to save it as a report from the report builder once you have it formatted to your liking. Saved searches do ***not*** include chart formatting parameters--to get those you need a saved report. This is especially important if you are planning to base a dashboard panel on the saved report, and you expect that panel to display with your custom formatting parameters.

At the top of the Chart section you'll find the *Formatting options* subsection, which contains the formatting options for your chart.

In this section, you can redefine the chart type (change it from a column chart to a bar chart, for example) and select a variety of other formatting options. Under **Format**, toggle between *General*, *Y-axis*, and *X-axis* sets of formatting controls. After you make changes, click the **Apply** button to have Splunk regenerate your chart with your formatting changes applied to the design.

Note: When you try to fine-tune the formatting for a report after the report job that it's based upon expires, Splunk draws an empty chart. You will not have this problem if you are building a report based on a saved report job. For more information about saving search and report jobs see Managing Jobs in this manual.

Choose a chart type

Use the **Chart Type** drop-down list to change how Splunk visualizes your report data. The list includes the following chart types:

- column
- bar
- line
- area
- pie
- scatter
- bubble (disabled in current version)

The **Chart Type** options that are actually available to you at any given time depend on the type of report that you've created. For example, if you've set up a *Values over time* report type on the Define Report Contents page, then the only **Chart Type** values that are available to you are *bar,*, *column, line*, and *area.*

For more details about the types of charts that you can create with the Splunk Report builder see the "Chart gallery" topic in this manual. It includes visual examples of each chart type and information about the kinds of situations that each chart type is best suited for. It also tells about the commands and Report Builder setups that get you to each chart type.

For more information about why certain chart types work for some searches but not others (why you can't always use the same search to generate both a bar and a pie chart, for example), see "Chart data structure requirements" in the "Chart gallery" topic.

Update general chart formatting options

The **General** chart formatting options available to you differ depending upon the type of chart you've selected. If you're working with a column, bar, line, or area chart, you can update the **Stack mode**. If you're working with a line or area chart, you can additionally adjust the way the chart displays **Null values**.

You can update the **Chart title** and **Legend placement** no matter what chart type you're working with.

Update X-axis and Y-axis formatting options

With the *X-axis* and *Y-axis* formatting option you can:

- Redefine the X- and Y-axis titles for the chart.
- Change the maximum and minimum values of the Y-axis for column, line, and area charts.
- Change the maximum and minimum values of the X-axis for bar charts.
- Turn display markers on and off for line and area charts.
- Switch the Y-axis scale from *linear* to *log* (as in "logarithmic") for column, line, area, scatter, and bubble charts (and do the same for the X-axis scale of bar charts).

You may decide you want to adjust the maximum and minimum values of the Y-axis (or X-axis, for bar charts) to focus on the differences between an otherwise fairly similar group of results.

For example, say you're looking at a column chart where all of the Y-axis values are between 114 and 145. You can set the minimum Y-axis value to 110, and the maximum Y-axis value to 150. This creates a chart that focuses the viewer's attention on the differences between each column while leaving out the nonessential similarities.

Similarly, putting the chart on a logarithmic scale can be handy for situations where values have wide variances. For example, you might have a column chart where most of the values come between 10 and 50, but a handful are as high as 1000. You can use the logarithmic scale to better see the differences between the lower values.

Save reports and share them with others

Save reports and share them with others

If you're happy with a report that you've created, in the **Report Builder**, you have multiple options for saving that report and sharing it with others.

Save reports or report results

There are three ways you can go when you select the **Save** drop-down list in the Format reports page of the Report Builder. You can:

- **save the report search string, time range, and associated formatting parameters**, so that you can run new reports based on those settings.
- **save the report and add it to a dashboard**. You can create a dashboard panel based on the saved report and add it to a new or preexisting dashboard.
- **save the report results as a report job**, so you can review the results of this specific run of the report at a later time.

Create a saved report

From the Format report page of the Report Builder, you can click **Save** and then select *Save report...* to open the **Save Report** dialog box. **Save Report** enables you to save the report search string and associated content and formatting parameters with a unique name. You can also change the time range for the report if necessary. After you save the report, it will appear under **Searches & Reports** in the top-level navigation bar, using the unique name that you assigned to it.

[add screenshot of Save Report window]

Note: To make it easier to find your report in the **Searches & Reports** list, you might include the word "report" or "chart" in its title to distinguish it from searches. If you are saving a large number of reports, consider developing a naming strategy to make individual saved reports easy to find.

When you run a saved report, it creates a new report using the search string, time range, and **chart** formatting parameters (the chart type, chart title, legend placement, and so on) that are associated with the original report. This is important to keep in mind, especially if you are planning to share the saved report with others or base dashboard panels on it . (Saved searches do ***not*** include chart formatting parameters--to capture these, generate a report based on the search, and save it instead.)

Note: Saved reports are a type of Splunk **knowledge object**, along with saved searches, event types, tags, and other items that enrich your Splunk data and make it easier to do what you need to do with Splunk. When you first save a Splunk knowledge object, it is only available to you in the app that you're using when you create it. *If you have the privileges to do so* **Save Report** enables you to share the report with all the users of the app you're currently using, with read-only access.

If you have the ability to set search and report object **permissions**, you can enable other kinds of access to your saved reports through **Manager > Searches and reports**, including sharing them with users in multiple apps and giving users write access so they can update saved report definitions. For more information, see "Curate Splunk knowledge with Manager" in the Knowledge Manager Manual.

Create a saved report and add it to a dashboard

You have the option of saving the report you've defined and simultaneously adding it a **dashboard** as the basis for a new dashboard **panel**. When you do this, the report is saved as well. From the Format report page of the Report Builder, you can click **Save** and then select *Save report and add to dashboard*. This brings up *Add to Dashboard*, which enables you to save the report and then add it to a new or preexisting dashboard as a dashboard panel.

Add to Dashboard also enables you to give your panel a title, identify the **panel type** (such as a chart panel or table panel), and determine a search schedule for the panel--it can run each time the dashboard loads, or it can be run as a **scheduled search** (this can make the panel load faster than it would otherwise).

[add screenshot of Add to Dashboard dialog, on Set up panel tab]

For a detailed explanation of how **Add to Dashboard** works, read the "Add a new search to a new or existing dashboard" subtopic of "Create and edit simple dashboards" in this manual.

When you create a new dashboard, the dashboard is only available to you, unless you have the ability to set dashboard object permissions, in which case **Add to Dashboard** also gives you the option to share it with other users of the app you are currently in.

The saved report that "Add to Dashboard" creates takes on the same permissions as the dashboard it is being added to, even if you are adding it to a preexisting dashboard.

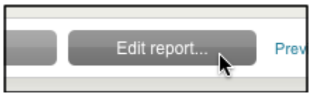
If you have the ability to set dashboard object **permissions**, you can enable other kinds of access to dashboards through **Manager > User interface > Views**, including sharing them with users in multiple apps and giving users write access so they can update dashboard definitions. For more information, see "Curate Splunk knowledge with Manager" in the Knowledge Manager Manual.

Edit a saved report

You may find that you need to update a report after you've saved it. For example, you might want to fine-tune the search language that the report is based on. Or you may not like the way the report's chart displays, in which case you can adjust the chart formatting parameters for the report.

To edit a saved report:

1. First, rerun the saved report (easily accomplished by selecting it from the top-level navigation of your app). When you rerun a saved report, Splunk opens it in the Report Builder and generates a chart that is based on a new **search job** but is constructed using the chart formatting parameters that were saved as part of the report definition. 2. To update the report's search string, or update its chart formatting parameters, select the **Edit report** button that appears near the top of the page:



3. A new instance of the Report Builder appears in a separate window. You can update both the search language and the chart formatting parameters for the report. 4. When you are dunThe Report Builder appears, enabling you to update both the search language and the report formatting. When you're done, click **Save** and then select *Save report* (to update the original saved report with your changes) or *Save report as...* (to capture your updates in a new saved report with a different name). You can also select *Save results only* to just save the results of this particular run of the report (see "Save report results only," below, for more information).

Note: If you run a saved report and it doesn't open in the Report Builder, but rather runs as a search using the search timeline, then it wasn't actually saved as a report with specific report formatting parameters. You'll need to *save it as a report* following the instructions in the subtopic above.

Save report results only

When you're on the Formatting results page of the Report Builder, you can click **Save** and select *Save results only* if you want to save the *results* of a particular run of a report, so you can review them at a later point in time. When you do this, you're saving a report "job," which you can access through the Jobs page.

If you tend to save a lot of jobs, you can save yourself some time looking for the job later in the Jobs page by selecting the **Get link...** button. This both saves the report job and gives you a direct link to the job. You can bookmark this link for future reference and/or share it with others.

For more information on managing search and report jobs on the Jobs page see "Supervise your search jobs" in this manual.

Managing saved report navigation

When you save a report, it should appear in one of the drop-down lists in the top-level navigation menu. In the Search app, for example, saved reports appear in the **Searches & Reports** list by default.

If you have write permissions for an app, you can change this default location, and even set things up so that reports with particular keywords in their names are automatically placed in specific categories in the navigation menu (so that Splunk automatically places reports with the word "website" in their name in a list of website-related searches and reports in the navigation menu. You can also move reports that have already been saved to the default list to different locations in the top-level navigation menu.

For more information, see "Define navigation for saved searches and reports" in the Knowledge Manager manual and "Customize navigation menus" in the Developer manual.

Share reports with others

There's no need to keep your completed reports to yourself if they contain useful information that should be seen by others. Splunk gives you a variety of ways to share them.

- **Export the event data to a file.** You can export the event data from your report to a csv, txt, json, or xml file. You can then archive this raw tabular data or input it into a third-party application such as MS Excel.
- **Print the report.** Splunk can send an image of the report you've created and its corresponding table straight to a printer, or save the report as a .pdf file depending upon available printer drivers.
- **Set up delivery of .pdf report printouts with email alerts** You can also arrange to have .pdf report printouts delivered with alert emails. For more information, see "Create an alert" in this manual.
- **Get (and share) a link to the results.** Select **Get link...** to get a URL link to the report results. You can share this link with other interested parties, who can view it as long as they have access to your instance of Splunk.

Note: Selecting **Get link...** automatically saves your report job, which you can access thereafter through the Jobs page. The Get Link to Results popup window also includes a link that enables you to undo this save action.

Add reports to views and dashboards

It's also possible to design specialized views and dashboards that include reports that you've defined. Dashboards can be made up of multiple panels that each display charts, lists, and other data that are generated by hidden, predefined searches.

Splunk provides a visual dashboard editor that enables you to quickly create simple dashboards. For more information, see "Create simple dashboards with the dashboard editor" in this manual. For information about creating views and more sophisticated dashboards, see the Developer manual.

Use report-rich dashboards and views

Use report-rich dashboards and views

Splunk's Search app comes packaged with a set of useful dashboards and views that also serve to demonstrate a few different configurations of our search and reporting modules. As such, they may help you come up with some ideas of how you might want to design some dashboards and views of your own.

Every page in a Splunk app is a view. For example, the core search page in the Search app is a default view that ships with that app. You can construct your own views as you design your own apps.

Dashboards are one of the most common types of views, and they are among the easiest to build. Each dashboard is made up of panels that can contain charts, tables, event lists, HTML, and text. Most panels are hooked up to searches that kick off when the dashboard is loaded, providing you with up-to-the-moment metrics and analysis. You can design dashboards to provide insight into just about any aspect of your IT data, from real-time breakdowns of online sales revenue to lists and charts detailing recent firewall attacks and other security concerns.

You can quickly get simple dashboards up and running. You can:

- Click **Add to dashboard** after you run a search or report that would be the good foundation for a dashboard panel. You can add the new dashboard panel to an existing dashboard or make it the first panel in a new dashboard.
- Use the **Visual Dashboard Editor** to create new dashboards and edit existing ones. You can create dashboard panels that are based on saved searches and reports, rearrange a dashboard's panel order, and much more.

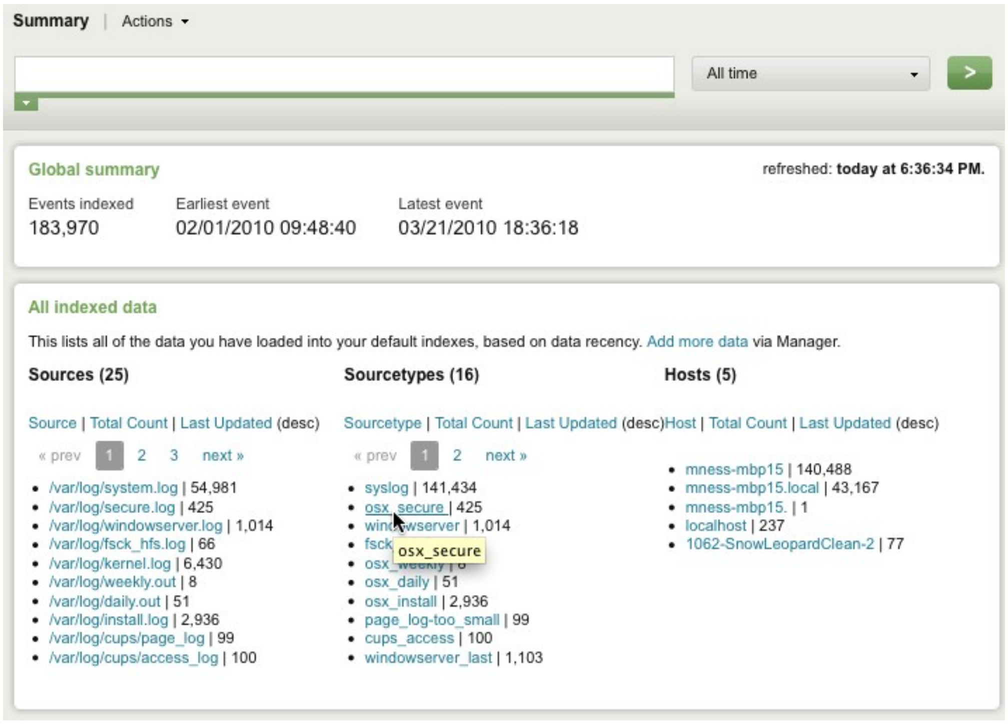
For more information, see "Create and edit simple dashboards" in this manual.

To learn how to create more sophisticated dashboards, see the "Build dashboards" chapter of the Developer manual.

Summary dashboard

The Summary dashboard is the first thing you see as you enter the Search app. It provides a search bar and time range picker which you can use to input and run your initial search. Below that, you'll find some elemental indexing metrics for this instance of Splunk, all of which are generated by inline searches and saved searches linked to the dashboard. You'll find a count of the total amount of events indexed, and the timestamps for the earliest and latest events indexed.

You'll also see lists displaying the various sources, sourcetypes, and hosts indexed by your Splunk instance, ordered by the total amount of events indexed for each field. Select a list item to kick off a search for occurrences of that particular field.



Global summary

refreshed: **today at 6:36:34 PM.**

Events indexed	Earliest event	Latest event
183,970	02/01/2010 09:48:40	03/21/2010 18:36:18

All indexed data

This lists all of the data you have loaded into your default indexes, based on data recency. Add more data via Manager.

Sources (25)

Source | Total Count | Last Updated (desc)

- /var/log/system.log | 54,981
- /var/log/secure.log | 425
- /var/log/windowserver.log | 1,014
- /var/log/fsck_hfs.log | 66
- /var/log/kernel.log | 6,430
- /var/log/weekly.out | 8
- /var/log/daily.out | 51
- /var/log/install.log | 2,936
- /var/log/cups/page_log | 99
- /var/log/cups/access_log | 100

Sourcetypes (16)

Sourcetype | Total Count | Last Updated (desc)

- syslog | 141,434
- osx_secure | 425
- windowserver | 1,014
- fsck
- osx_weekly | 8
- osx_daily | 51
- osx_install | 2,936
- page_log-too_small | 99
- cups_access | 100
- windowserver_last | 1,103

Hosts (5)

Host | Total Count | Last Updated (desc)

- mness-mbp15 | 140,488
- mness-mbp15.local | 43,167
- mness-mbp15. | 1
- localhost | 237
- 1062-SnowLeopardClean-2 | 77

Note: Keep in mind that index permissions are set at the role level. This means that viewers of the Summary dashboard can only see indexing information for indexes that they have permissions to see, according to their role. For more information about users, roles, and role-based index permissions, see the "Add and manage users" section of the Admin manual.

Not finding the events you're looking for?

When you add an input to Splunk, that input gets added relative to the app you're in. Some apps, like the *nix and Windows apps that ship with Splunk, write input data to a specific index (in the case of *nix and Windows, that is the 'os' index). If you review the summary dashboard and you don't see data that you're certain is in Splunk, be sure that you're looking at the right index. You may want to add the 'os' index to the list of default indexes for the role you're using. For more information about roles, refer to the topic about roles in the Admin Manual.

Status dashboards

The Search app includes five collections of dashboards that display different kinds of Splunk status information. You can find them under **Status** in the top-level navigation bar.

Note: These dashboards are only visible to users with Admin role permissions. For more information about users and roles, see the "Add and manage users" section of the Admin manual. the Admin manual. For more information about setting up permissions for dashboards, see the Knowledge Manager manual.

- **Search activity** - This dashboard collection provides at-a-glance info about search activity for your Splunk instance. You can find out when searches are running, the amount of load they're putting on the system, which searches are the most popular, which search views and dashboards are getting the most usage, and more.
- **Index activity** - This collection of dashboards expands upon the basic indexing statistics presented in the summary dashboard. You'll see the total events indexed (broken out by index), the top five indexed sourcetypes, the indexing rate by sourcetype over the past 24 hours, lists of indexing errors, and a number of other useful stats.
- **Server activity** - This small collection of dashboards provides metrics related to splunkd and Splunk Web performance. You'll find the numbers of errors reported, lists of the most recent errors, lists of timestamping issues and unhandled exceptions, a chart displaying recent browser usage, and more.
- **Inputs activity** - This dashboard displays information about your Splunk inputs. You can see your most recently processed files and your most recently ignored files.
- **Scheduler activity** - This collection of dashboards gives you insight into the work of the search scheduler, which ensures that both ad hoc and scheduled searches are run in a timely manner.

Advanced charting view

In the top-level navigation bar's **Views** list, you can find the **Advanced charting** view. This example of view construction enables you to build charts without opening up a separate Report Builder window. Enter a search that uses reporting language into the search bar, and the resulting chart appears in the results area.

Manage views

The **Manage views** link in the **Views** list takes you to the Views page in Manager, where you can review and update the views that you have permission to manage, change their permissions, and add new views. To create or update views here you need to be familiar with XML and have an understanding of how views are developed in Splunk. For more information see the Developers manual.

Note: You can also get to the Views page by navigating to **Manager > User interface > Views.**

Send dashboard printouts to interested parties

Splunk enables you to generate .pdf format printouts of selected dashboards and send them out to interested parties on a regular schedule (daily, biweekly, monthly...whatever fits your need). This is an excellent option if you have executive teams, managers, or project stakeholders that need to see specific information displays on a regular basis. For more about this feature, see "Schedule delivery of dashboard PDF printouts via email" in this manual.

Create and edit simple dashboards

Create and edit simple dashboards

Splunk gives you tools that enable you to build out a simple **dashboard** without having to touch a bit of the XML behind it. You have two methods at your disposal:

- **Add new searches directly to a dashboard** - When you run a search that you think would be the good basis for a dashboard **panel**, you can click **Add to dashboard** to open a window that enables you to save the search and then add it as a panel to a existing or new dashboard. (You can also select *Add to dashboard* from the **Actions** dropdown to open the same window.)
- **Use the Visual Dashboard Editor to create and edit simple dashboards** - If you have a set of **saved searches** and/or **saved reports** that you would like to use to populate a set of dashboard panels with useful tables and charts, you can use the Visual Dashboard Editor to both define the dashboard panels and manage their arrangement in the dashboard. You can also use the **Visual Dashboard Editor** to edit dashboards that have been created or updated through usage of the **Add to dashboard** link. To use it to create a new dashboard, select *Create dashboard* from the **Views** menu.

Note: These dashboard creation methods are great for getting simple, functional dashboards up and ready for use in a matter of minutes. However, if you want to design more complex dashboards that include form inputs, special drilldown actions, and similar advanced features, you'll need to work directly with Splunk's advanced XML syntax--you can't create dashboards with that level of sophistication using the methods described in this topic alone. For more information, see the "Build dashboards" section of the Developer manual.

This topic discusses both of these dashboard creation and management methods in greater detail. The first subsection shows you how to add a new search to a new or existing dashboard by clicking **Add to dashboard**. The following sections show you how to use the Visual Dashboard Editor to create new dashboards and edit existing ones.

Add a new search to a new or existing dashboard

When you run a search or report that would make a good foundation for a dashboard panel, click **Add to dashboard.** This opens the Add to Dashboard window, which enables you to make the search the basis for a new dashboard panel in a new or existing dashboard.

You can use a simple search as the basis for a "list" style dashboard panel that simply provides a listing of returned events. You can use reports (searches that use reporting commands) as the basis for panels that display tables and charts. And you can also set up searches that power "single value" panels.

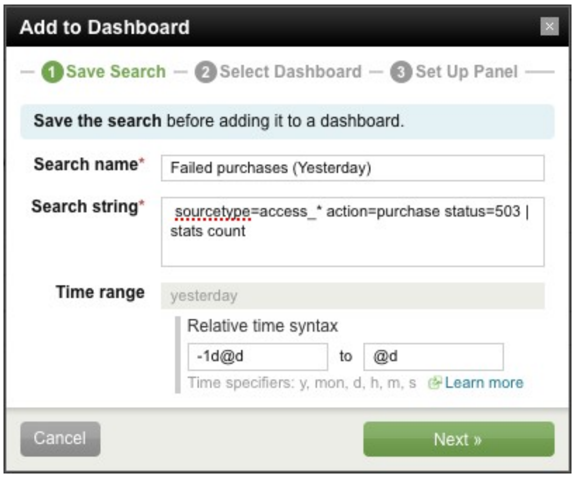
The Add to Dashboard window is broken up into three steps:

- **Save Search**, where you name the search, refine the search string (if necessary), and define its time range (or the duration of its real-time window, if it is a real-time search).
- **Select Dashboard**, where you decide whether the dashboard panel you are creating is going to an existing dashboard or a new one. If it's going to a new dashboard you can name the dashboard and determine whether it is private or shared.
- **Set Up Panel**, which enables you to define basic panel information, such as the panel title, panel type, and refresh interval.

The following subsections discuss these steps in more detail.

Saving the dashboard panel search

The **Save Search** step of the Add to Dashboard window is the same as the Save Search window, which is accessed by clicking the **Save search** link after running a search. It is discussed in "Save searches and share search results," in this manual. For the full rundown on saving searches, please go to this topic.


Add to Dashboard

Save the search before adding it to a dashboard.

Search name*: Failed purchases (Yesterday)

Search string*: sourcetype=access_* action=purchase status=503 | stats count

Time range: yesterday

Relative time syntax: -1d@d to @d

Time specifiers: y, mon, d, h, m, s

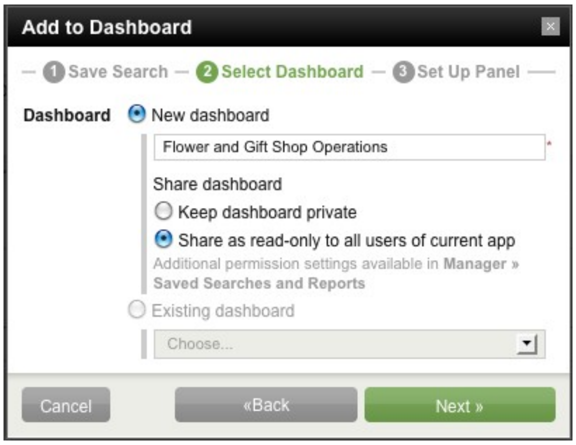
For more information about defining search time ranges with relative time syntax (where there's a set start and end time) see "Change the time range to narrow your search" in this manual.

For more information about setting up real-time searches and real-time search windows, see "Search and report in real time" in this manual.

Selecting a dashboard

In the **Select Dashboard** step you determine where the new dashboard panel that you are creating resides. You can create a new simple dashboard and add the your new panel to it. Or you can add it to a simple dashboard that already exists within the app that you are currently using.

NOTE: The Add to Dashboard window does not allow the addition of dashboard panels to dashboards created with the advanced XML syntax.



Dashboard [✓ New dashboard]
Flower and Gift Shop Operations
Share dashboard
Keep dashboard private
[✓ Share as read-only to all users of current app]
Additional permission settings available in Manager » Saved Searches and Reports
Existing dashboard
Choose...

Adding the panel to a new dashboard

If you select **New dashboard** Splunk makes the search that you just saved the basis for the first dashboard panel in a new dashboard. You need to provide a unique name for the dashboard. This dashboard name will appear in the **Views** dropdown near the top of the Splunk interface.

You also need to set the permissions for the new dashboard (and for the search that powers the panel that you're adding to it); they determine who can see and use the dashboard. Under **Share dashboard,** the permission settings are set to *Keep search private*, which means that you are the only person who can view and use the dashboard and search. But you can change the setting to *Share as read-only to all users of current app.* This setting ensures that the dashboard and search are accessible by all other users of the app you're currently in. It also ensures that they can see the search that the new dashboard panel is based upon.

Note: After Splunk creates the new dashboard, you can go to **Manager > User interface > Views** to widen or narrow the permissions for the dashboard and **Manager > Searches and Reports** to do the same for the permissions of the associated search. For example, you can restrict it so that it is only viewable by users with particular roles. Or you can make it globally available to all apps in your Splunk implementation. For more information about permission management, see "Curate Splunk knowledge with Manager" in the Knowledge Manager Manual.

Adding the panel to an existing dashboard

When you choose to have Splunk add the new dashboard panel to an existing dashboard, Splunk adds the panel to the bottom of the dashboard that you select. If necessary, you can use the Visual Dashboard Editor to reorder the arrangement of panels on that dashboard. For more information about that, see "Set up the panel layout," below.

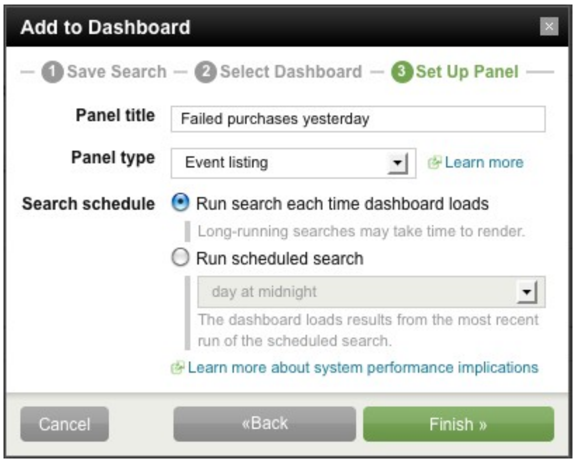
Note: The **Existing dashboard** list only shows dashboards that utilize **simplified XML** and which your **role** grants you write **permissions** for.

When you add the new dashboard panel to an existing dashboard, the new search associated with that panel (you saved it in the previous step) gets the same permissions as that dashboard. If you want to adjust those search permissions, go to **Manager > Searches and Reports**. For more information about permission management, see "Curate Splunk knowledge with Manager" in the Knowledge Manager Manual.

Setting up the new dashboard panel

In the **Set up Panel** step of the Add to Dashboard dialog box, you define the title, type, search schedule, and refresh interval of the new panel. All of these settings can be edited in the Visual Dashboard Editor.

Here's an example of what the panel looks like if your base search is a standard search (it provides slightly different options for real-time searches):


Add to Dashboard

1 Save Search — 2 Select Dashboard — 3 Set Up Panel

Panel title: Failed purchases yesterday

Panel type: Event listing

Search schedule: (●) Run search each time dashboard loads
Long-running searches may take time to render.
( ) Run scheduled search
day at midnight
The dashboard loads results from the most recent run of the scheduled search.

You can choose from four different panel types:

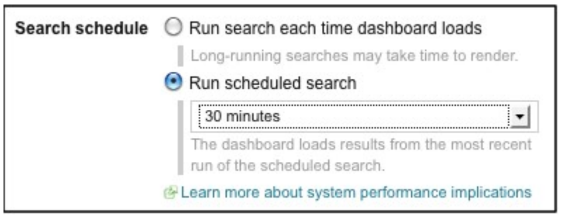
- **Data table**, which displays search results in tabular form.
- **Single value**, which is designed for searches that return a single value. You can use the `rangemap` command in the base search to have the panel display geen, yellow, or red, depending on the value returned.
- **Chart**, which displays the results of reports as a chart (such as a column chart or line chart).
- **Event listing**, which displays search results as a simple list of returned events.

For more information about the available panel types, see the "Create your first panel" subtopic in the Visual Dashboard Editor documentation, below.

Determine what happens when you load a panel based on a standard search

Search schedule enables you to determine what happens with your new panel when the dashboard it is associated with is loaded. The options you get here differ depending on whether the search is based on a standard search or a real-time search.

If you're defining a dashboard panel that is based on a standard search, such as a search that only looks at events from the preceding hour when it is run, you can opt to run the search each time the dashboard loads, or have the search run in the background on a regular schedule.


Search schedule
( ) Run search each time dashboard loads
Long-running searches may take time to render.
(●) Run scheduled search
30 minutes
The dashboard loads results from the most recent run of the scheduled search.
Learn more about system performance implications

If you choose *Run search each time dashboard loads* this means that the standard search upon which the panel is based is kicked off when the dashboard loads, displaying results relative to the load time of the dashboard. If your search takes a short time to load, you may want to go with this option.

If you select *Run scheduled search* you are setting up the base search for the panel as a **scheduled search**. This means that it will run on the interval that you define here, such as every 15 minutes, every 2.5 hours, or every day at midnight (choose an interval that aligns with the time range of your search for best results). When the dashboard loads, the panel you've added instantly displays results from the most recent run of the scheduled search. This is a good option for searches that usually have long run times.

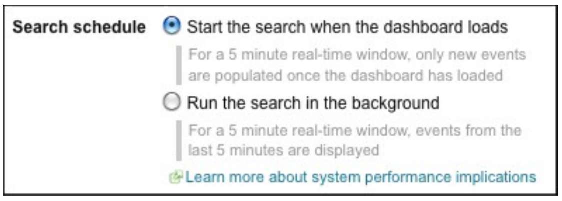
You can choose a preset scheduled search interval from the list, or you can define your own by selecting *Custom* and then defining a schedule in the provided field using standard cron notation.

Using scheduled searches for dashboards can burden your system, especially if you have a number of scheduled and backgrounded searches running simultaneously.

For more information about scheduling searches, including examples of standard cron notation usage, see the "Schedule the search" portion of the "Create an alert" topic, in this manual.

Determine what happens when you load a panel based on a real-time search

If you're defining a dashboard panel that is based on a real-time search, you get a different set of **Search schedule** options for that panel. You can opt to have the real-time search start when the dashboard loads, or you can have the real-time search run in the background.


Search schedule
(●) Start the search when the dashboard loads
For a 5 minute real-time window, only new events are populated once the dashboard has loaded
( ) Run the search in the background
For a 5 minute real-time window, events from the last 5 minutes are displayed
Learn more about system performance implications

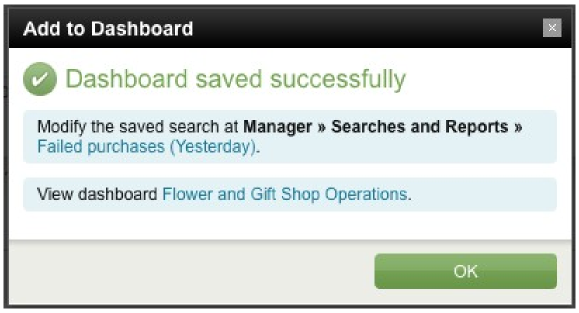
If you choose *Start the search when the dashboard loads* the real-time search starts when the dashboard loads. The panel appears to be empty at first, and is populated with events as Splunk identifies them in the real-time event feed. This can be a good option if the real-time search uses a relatively brief time range window, such as 30 or 60 seconds, because your users won't have to wait long to see results (if any are found).

If you select *Run the search in the background*, the search starts when the alert is saved and runs continuously in the background. This method ensures that when the dashboard is loaded, the panel is fully populated with the matching events for the real-time search in progress.

Note: Continuously running real-time searches in the background can negatively affect the performance of your Splunk implementation. If your system seems to be running slow, consider redefining dashboard panels so their real-time searches start when the dashboard loads.

Saving the new dashboard panel

After you click **Finish** on the **Set Up Panel** step of the Add to Dashboard dialog box, save the new dashboard panel, a screen appears. It provides a link to the detail information about the base saved search (in case you want to update it), and a link to the dashboard that holds the panel that you have created.



Modify the saved search at **Manager » Searches and Reports »** Failed purchases (Yesterday).

View dashboard Flower and Gift Shop Operations.

Create dashboards with the Visual Dashboard Editor

The Visual Dashboard Editor enables you to build fully functional dashboards, complete with panels that are linked to searches. These searches can exist among your saved searches, or be entered inline into the panel definition. You can also use the Visual Dashboard editor to update dashboards that were created through the Add to Dashboard dialog box (see the preceding subsection).

The following procedure gets you started with the Visual Dashboard Editor:

1. Open the **Views** menu and click*Create dashboard*.

2. Name your new dashboard. Designate a unique **ID** for it and give it the **Name** that it will be identified by in the top-level navigation menu.

3. Click **Create** to create your new dashboard.

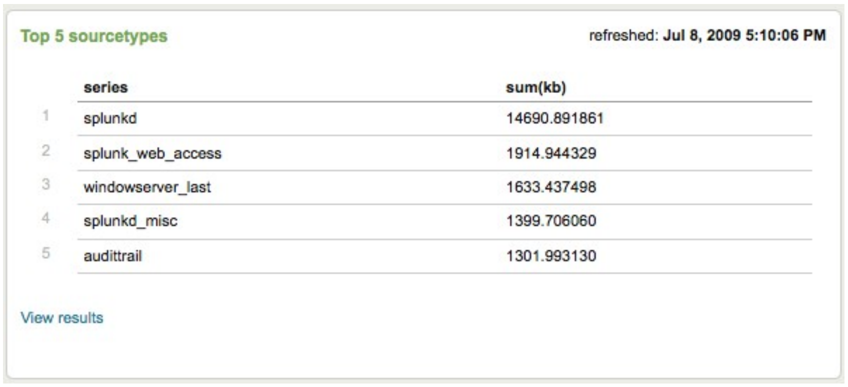
4. When your new dashboard appears, it is empty. To start defining panels for it, click **Edit the dashboard** to open the Visual Dashboard Editor.

Create your first panel

In the Visual Dashboard Editor, start by choosing a **Panel type**. The Visual Dashboard Editor enables you to create four types of dashboard panels: data tables, charts, event lists, and single value panels. (There are two other panel types--list panels and HTML panels--that you can only create by working directly with the simple XML for the dashboard.)

The data table panel type

The **Data table** panel type presents report results in tabular format:



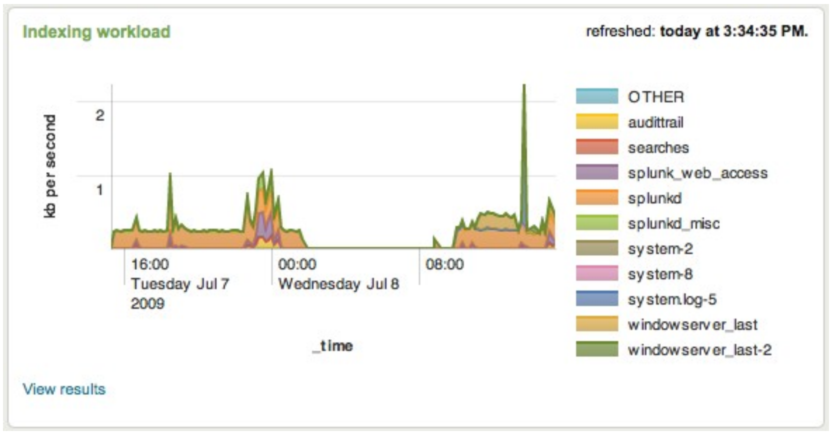
Top 5 sourcetypes — refreshed: **Jul 8, 2009 5:10:06 PM**

	series	sum(kb)
1	splunkd	14690.891861
2	splunk_web_access	1914.944329
3	windowserver_last	1633.437498
4	splunkd_misc	1399.706060
5	audittrail	1301.993130

View results

The chart panel type

The **Chart** panel type displays report results as a chart.



Now, if you want the stacked area chart panel shown here to display as a line chart instead, you have two options:

- You can edit the formatting options for the saved report so that it displays as a stacked area chart.
- You can override the default chart formatting by modifying the simple XML for the panel.

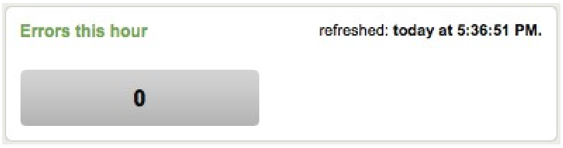
Important Note: Be aware that the chart panel type takes its default chart formatting parameters from the saved report that feeds it. For example, say you have a saved report that is designed to appear as a column chart. If you use this saved report in a chart panel, the chart panel will display a column chart by default.

It's also important to understand that a straightforward search without a transforming command ***will not*** show you any useful data when it is set up as a chart panel. The simple search `index=_internal` just returns a list of events that can't be represented in chart form. However, the search `index=_internal | timechart count span=1h` returns hourly totals of events, which can easily be represented as a bar, column, line, or area chart.

It may be helpful to think of things this way: if the data returned by the search can be represented as a table (rather than just a list of events) it can likely be turned into a chart of some kind. For more information about this, see the subtopic on "Chart data structure requirements" in this manual.

The single value panel type

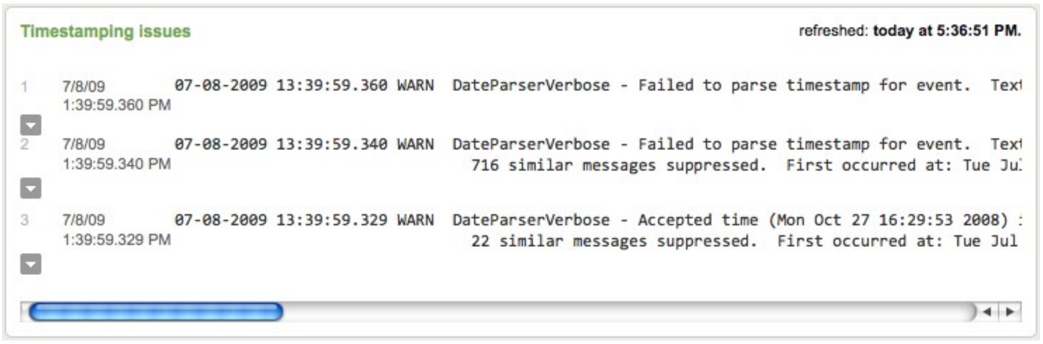
The **Single value** panel type displays a single numerical value as its result. For example, you could connect this to a search that returns the total number of 404 errors in your server over the past hour, or which displays the average access delay for a webserver. The panel retrieves the value from the first field in the first result.



Note: You can set single value panels to appear green, yellow, or red depending on the value they display. To do this you need to add the `rangemap` command to the base search and work with the XML behind the panel. You can find instructions in "Add a single value" in the Developer manual.

The event listing panel type

The **Event listing** panel type displays a listing of events returned by a search. This panel type is good for searches for particularly rare kinds of events, such as events that contain a significant but uncommon error message.



```
1  7/8/09         07-08-2009 13:39:59.360 WARN  DateParserVerbose - Failed to parse timestamp for event.  Text
   1:39:59.360 PM
2  7/8/09         07-08-2009 13:39:59.340 WARN  DateParserVerbose - Failed to parse timestamp for event.  Text
   1:39:59.340 PM                                 716 similar messages suppressed.  First occurred at: Tue Jul
3  7/8/09         07-08-2009 13:39:59.329 WARN  DateParserVerbose - Accepted time (Mon Oct 27 16:29:53 2008) :
   1:39:59.329 PM                                 22 similar messages suppressed.  First occurred at: Tue Jul
```

Enter a name for the panel and then select a saved search or report to associate with it. Click **Add panel** to add your new panel to the **Panel layout** section.

Note: We recommend that you design your dashboard panels to use scheduled searches whenever possible, especially if you expect them to have a significant number of users. When you use a scheduled search to populate a dashboard panel, Splunk just retrieves the data associated with the last scheduled run of that search when the dashboard is refreshed. This impacts system performance far less than if you have it rerun all of the dashboard reports from scratch at each refresh, and it helps you avoid situations where too many reports are being run concurrently by multiple users.

For more information about defining scheduled searches, see "Schedule saved searches" in this manual.

Set up the dashboard panel layout

Create additional panels using the same method as the first one. As they appear in the **Panel layout** section, you can click their titles and drag them to adjust their arrangement in the dashboard.

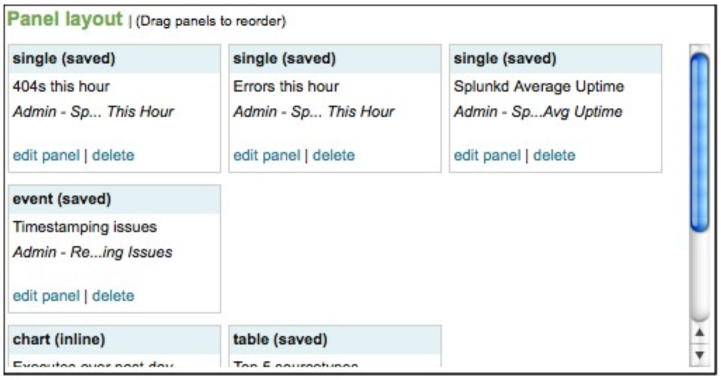
The Visual Dashboard Editor enables you to set up dashboards with rows of one to three panels. By default Splunk sets these up so that each panel in a row has an equivalent width, but the panel height can differ depending on the panel type and the information the panel is displaying.

Note: At current, the maximum number of panels per row is three. This limitation will be removed in an upcoming release.

Here are a few guidelines that you might want to follow when creating dashboard layouts with groups of panels.

- Because single value panels are small, they appear best when arranged in rows of three. They display too much white space when arranged in rows of one or two.
- Event listing panels display best in single panel rows, because they display lines of event data that would otherwise need to be seen via a horizontal scroll bar.
- Data table panels and chart panels work best in rows of one or two panels. You can mix table and chart panels together on the same row, but data table panels can vary in height depending on the length of the tables populating them (consider having the searches that feed them return only the top or last five values).

Here's an example of a dashboard layout that uses the above guidelines. Note that the top row contains three single value panels, the middle row has just one event listing panel, and the bottom row has a chart panel and data table panel, respectively.



Panel layout | (Drag panels to reorder)

single (saved)
404s this hour
Admin - Sp... This Hour
edit panel | delete

single (saved)
Errors this hour
Admin - Sp... This Hour
edit panel | delete

single (saved)
Splunkd Average Uptime
Admin - Sp...Avg Uptime
edit panel | delete

event (saved)
Timestamping issues
Admin - Re...ing Issues
edit panel | delete

chart (inline)

table (saved)

Note: Most of these display issues can be dealt with by simple adjustments to the XML behind the dashboard. You can add paging controls for long data table panel types, group panels together under the same heading, change chart formatting parameters, and more. You can access the XML by clicking **Edit Name/XML** at the bottom of the Visual Dashboard Editor window. For more information about editing XML for dashboards created with the Visual Dashboard Editor, see "Panel reference for simple XML" in the Developer manual.

Change the dashboard name or XML configuration

Select **Edit name/XML** to edit the dashboard name and the simple XML behind the dashboard. For more information about editing XML for dashboards created with the Visual Dashboard Editor, see the "Build dashboards" chapter in the Developer manual.

Change dashboard permissions

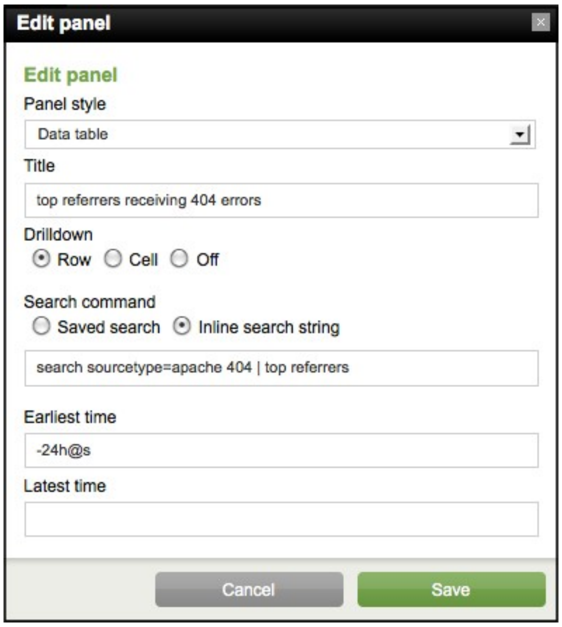
Select **Edit permissions** to expand or restrict the role-based read and write permissions for the dashboard. When you set dashboard permissions you can also define the app availability of the app. The dashboard can be:

- A private view available only to yourself.
- Available to one app only (the app it was designed in) and the people who have permission to use it.
- "Globally" available to all Splunk apps in your system (and therefore all of your Splunk users).

Set up or update individual dashboard panels

You can use the Visual Dashboard Editor to define aspects of individual panels. To do this, click **Edit panel** on a panel that has been added to the **Panel layout** section of the editor. The Edit panel window appears for that panel. You can use this window to:

- Update the **Panel style** and **Title**.
- Select basic table and chart drilldown options for dashboard panels.
- Add inline search strings to dashboard panels.


Edit panel

Edit panel

Panel style: Data table

Title: top referrers receiving 404 errors

Drilldown: [✓ Row] Cell Off

Search command: Saved search [✓ Inline search string]

search sourcetype=apache 404 | top referrers

Earliest time: -24h@s

Latest time:

All *data table* and *chart* panel types can have "drilldown" functionality, where you click on the table or chart to set off a search that drills down on a particular aspect of that table or chart. For example, the panel being defined in the above example of the Edit panel window might create a table that looks like this:

referrer	count
http://prettypwnny.com	243
http://deepthaduke.com	65

If this table is set up for row drilldown, when you click on the first row of the panel, Splunk will move to the Search view and run the following search:

```
search sourcetype=apache 404 referrer="http://prettypwnny.com"
```

....which provides detail information on the 404 error events associated with the PrettyPwnny referrer over the specified search duration.

Data table panel types have three drilldown options under **Drilldown**. They are:

- *None*, which turns off the drilldown functionality for the table.
- *Row*, which means that a click on a row sets off a search across the x-axis value represented by that row. For example, if the row represents a specific period of time, then a click on that row sets off a search that is identical to the search that generated the chart, except that it only covers the time range that the row represents.
- *Cell*, which sets off a search that is restricted to the x-axis value (the row) and the y-axis value (the column) represented by the cell, when the originating search includes a "split by" clause.

  For example, you could use a cell-click in a table resulting from a "timechart count by clientip" search, where the columns are values of `clientip`, like `192.168.0.14`. The resulting search displays a histogram that shows when those events occurred during that period.

Note: *Data table* panels are set to the *Row* click type by default when they are created.

Chart panels have two drilldown options under **Drilldown**. They are:

- *Off*, which turns off the drilldown functionality for the chart.
- *On*, which lets you drill down on a particular part of a chart or legend by clicking on it. For example, when you click on a particular bar of a bar chart, Splunk runs a search (based on the original search used to generate the bar chart) that covers only the block of time represented by that bar.

Note: Chart panels have their drilldown value set to *On* by default when they are created.

For more information about how table and chart drilldown actions actually *work*, see "Understand basic table and chart drilldown actions" in this manual.

You can specify much more complex drilldown actions for tables when you design them using advanced XML. For more information about designing drilldown actions for dashboards and views see the Developer manual.

Define searches for dashboard panels

All dashboard panels are associated with searches. You can determine whether a panel runs off of a predefined, saved search, or whether it uses a search that has been specifically designed for the panel and associated with it in an "inline" manner.

In the **Search command** section, select either *Saved search* or *Inline search string*.

- If you select *Saved search* you can select a saved search for the panel from a list of all of the saved searches that are associated with dashboard's parent app (the app you are in when you edit the dashboard). The dashboard will run this search for the panel every time you open it or refresh it.

- If you select *Inline search string* you can define a search string that is specific to this panel. Specify the inline search time range by placing relative time modifiers in the **Earliest time** and **Latest time** fields.

Note: Keep in mind that the Visual Dashboard Editor does not enable you to set up the formatting parameters for chart panels. If you design a chart panel with an inline search and find that you want to adjust the chart formatting, you have to edit the simple XML behind the dashboard.

Editing dashboards created with the Visual Dashboard Editor

You can edit any dashboard that was created with the Visual Dashboard Editor (or which uses **simple XML**, such as dashboards and panels created by the Add to Dashboard dialog box) by bringing up the dashboard and then clicking on **Edit dashboard...** in the **Actions** menu. The Visual Dashboard Editor appears.

Managing dashboard navigation

Because dashboards are a type of view, by default any new dashboard you create will appear in the View drop-down list in the top-level navigation menu. You can edit the XML behind the navigation menu to:

- Change the location of your unclassified dashboards. You can move dashboards to existing lists (or "view collections") in the navigation menu, or create new lists for them.
- Create nested collections (view collections within navigation bar lists) that classify similar dashboards together. For example, under your Dashboards dropdown, you could have a "Web Server" collection that groups together a set of dashboards that display different kinds of firewall information for your web server.

Note: Navigation is managed on an app by app basis. If your dashboard has been promoted globally to all of the apps in your system, it initially appears in the default drop-down list for "unclassified" views in those apps' top-level navigation menus. Users with write permissions for those apps can move the dashboard to its proper location in the app navigation menus as appropriate.

For an overview of navigantion menu management see "Define navigation for saved searches and reports" in the Knowledge Manager manual.

If you have write permissions for your app, you can access its navigation menu XML by opening Manager, clicking **Navigation Menus**, and then clicking the name of the navigation menu for your app. See the "Build navigation for your app" topic in the Developer manual for details about working with the navigation menu code.

Learn how to change and format dashboard panel visualizations with the Visualization Editor.

Learn how to change and format dashboard panel visualizations with the Visualization Editor.

Schedule delivery of dashboard PDF printouts via email

Schedule delivery of dashboard PDF printouts via email

Splunk enables you to have .pdf printouts of dashboards generated and sent as email attachments to project stakeholders on a regular schedule.

There are many situations that might warrant the scheduled delivery of dashboard views via email. For example, you could have an executive team that wants to receive an operations report in their inbox on a daily or weekly basis. To address this, you would first set up a dashboard view that presents high-level operations metrics in table and chart form. Then you would use this functionality to generate and distribute it on a regular schedule to the stakeholders via email, as a .pdf attachment.

To do this:

1. Go to the dashboard view that you want to share and select *Schedule for PDF delivery* from the **Actions** menu.

2. Define a schedule for the report in much the same way that you schedule a saved search, using standard cron notation.

3. Identify one or more email recipients.

4. Click **Save**.

On the schedule that you set up, Splunk will run the selected dashboard, generate a .pdf printout of it, and send that printout to the identified email recipients.

Important: Use of this feature requires the setup of the PDF Printer app on a central Linux host. If you don't have this set up, contact a system administrator. For more information see "Configure PDF printing for Splunk Web" in the Installation manual.

For more information about dashboards in Splunk apps, start with "Use report-rich dashboards and views" in this manual. From there you can get to topics that show you how to create simple dashboards with the visual dashboard editor and design more complex ones using XML.

Note: You can also send .pdf printouts of scheduled searches and reports as email attachments. For more information on setting this up, see "Create an alert" in this manual.

Search Examples and Walkthroughs

What's in this chapter

What's in this chapter

This chapter contains links to examples that walk you through constructing searches and, in some cases, building reports and setting up alerts. Some of these examples are in the following topics, others are throughout the manuals.

Reporting: Build a chart of multiple data series
: Splunk's reporting commands do not support a direct way to define multiple data **series** in your charts (or timecharts). This example demonstrates how you can produce a chart of multiple data series using the stats and xyseries commands.

Reporting: Compare hourly sums between multiple days
: This is an example of how you might want to use `chart` to *compare* values collected over several days.

Reporting: Use rangemap to group together ranges of results
: This is one example of how you can use the rangemap command to group ranges of numerical values into a field that you can then use in charts and tables.

Monitor and alert on Windows disk usage
: This example walks you through setting up a **basic conditional alert** that sends an email when the disk usage falls below a certain percentage.

Reporting: Build a chart of multiple data series

Reporting: Build a chart of multiple data series

Splunk's reporting commands do not support a direct way to define multiple data **series** in your charts (or timecharts). However, you CAN achieve this using a combination of the `stats` and `xyseries` commands.

The `chart` and `timechart` commands both return tabulated data for graphing, where the x-axis is either some arbitrary field or `_time`, respectively. When these commands are used with a split-by field, the output is a table where each column represents a distinct value of the split-by field.

In contrast, the `stats` command produces a table where each row represents a single unique combination of the values of the group-by fields. You can then use the `xyseries` command to redefine your data series for graphing.

For most cases, you can simulate the results of "... | chart n by x,y" with "... | stats n by x,y | xyseries x y n". (For the `timechart` equivalent of results, x = `_time`.)

Scenario

Let's say you want to report on data from a cluster of application servers. The events gathered from each server contain information such as counts of active sessions, requests handled since last update, etc. and are placed in the `applications_servers` index. You want to display each server instance and the number of sessions per instance on the same timechart so that you can compare the distributions of sessions and load.

Ideally, you want to be able to run a timechart report, such as:

```
index=application_servers | timechart sum(handledRequests) avg(sessions)
by source
```

However, timechart does not support multiple data series; so instead, you need run a search similar to the following:

```
index=application_servers | stats sum(handledRequests) as hRs,
avg(sessions) as ssns by _time,source | eval s1="handledReqs sessions" |
makemv s1 | mvexpand s1 | eval
yval=case(s1=="handledReqs",hRs,s1=="sessions",ssns) | eval
series=source+":"+s1 | xyseries _time,series,yval
```

Walkthrough **`... | stats sum(handledRequests) as hRs, avg(sessions) as ssns by _time,source`**

This uses the `stats` command to calculate statistics for each source value: The sum of `handledRequests` values are renamed as `hRs`, and the average number of `sessions` are renamed as `ssns`.

```
... | eval s1="handledReqs sessions" | makemv s1 | mvexpand s1
```

This uses the `eval` command to add a single-valued field "s1" to each result from the stats command. Then, the makemv command converts sl into a multivalued field, where the first value is "handleReqs" and the second value is "sessions". The mvexpand then creates separate series for each value of s1.

```
... | eval yval=case(s1=="handledReqs",hRs,s1=="sessions",ssns)
```

This uses the eval command to define a new field, yval, and assign values to it based on the case that it matches. So, if the value of s1 is "handledReqs", yval is assigned the "hRs" value. And, if the value of s1 is "sessions", yval is assigned the "ssns" value.

```
... | eval series=source+":"+s1
```

This uses the eval command to define a new field, series, which concatenates the value of the host and s1 fields.

```
... | xyseries _time,series,yval
```

Finally, the xyseries command is used to define a chart with _time on the x-axis, yval on the y-axis, and data defined by series.

Reporting: Compare hourly sums between multiple days

Reporting: Compare hourly sums between multiple days

While the `timechart` is great for creating charts that show trends over time, it has strict boundaries limiting what it can do. There are times when you're better off with the `chart` command, which can provide more flexibility.

This is an example of how you might want to use `chart` to *compare* values collected over several days. This is something you can't do with `timechart`

Scenario

You have two searches that are nearly identical. They both show the hourly sum of the `P` field over a 24-hour period. The only difference is that one search covers a period ten days in the past, while the other covers a period nine days into the past:

Search 1:

```
earliest=-10d latest=-9d | timechart span="1h" sum(P)
```

Search 2:

```
earliest=-9d latest=-8d | timechart span="1h" sum(P)
```

What you'd like to do is create a column chart that combines the results of these two searches, so you can see the sum of `P` for 3pm, ten days ago side-by-side with the sum of `P` for 3pm, nine days ago.

Solution

You can't pull this off with the `timechart` command. But you can do it with the `chart` command, and it's pretty simple. Set up a search that covers both days, and then have it create a "sum of P" column for each distinct `date_hour` and `date_day` combination found in the search events.

The finished search looks like this:

```
earliest=-10d latest=-8d | chart sum(P) by date_hour date_day
```

This produces a single chart with 24 slots, one for each hour of the day. Each slot contains two columns that enable you to compare hourly sums between the two days covered by the time range of the report.

For a primer on reporting searches and how they're constructed, see "Use reporting commands" in the User Manual.

For more information about `chart>` and `timechart` functions, see "Functions for stats, chart, and timechart" in the Search Reference Manual.

Reporting Use rangemap to group together ranges of results

Reporting Use rangemap to group together ranges of results

Monitor and alert on Windows disk usage

Monitor and alert on Windows disk usage

This example discusses searches you can use to monitor and report on Windows disk usage. It also walks through the steps for setting up a conditional alert that sends an email when the disk usage falls below a certain percentage.

Scenario

I am setting up a search to alert me when a Windows host or Linux host runs below a certain percentage of Diskspace.

I have tried to schedule alerts based upon Windows Event codes:

```
host="*" source="wineventlog:system"(\"EventID=4133\"OR \"EventID=1082\")
```

However it is not as useful as measuring the disks usage and alerting when the usage falls below say 10%:

```
index="os" sourcetype="df" host=* | multikv fields FileSystem, UsePct |
strcat host '@' Filesystem Host_FileSystem | convert rmunit(UsePct) |
search UsePct < 11 | timechart
```

Disk Utilization Report`source="wmi:localphysicaldisk" "Name=Total" | timechart avg (UsePct) as "Disk Space", avg(DiskUsage) as "Disk Usage %"`

Set up conditional alert

content coming soon!