Splunk Essentials

Leverage the power of Splunk to efficiently analyze machine, log, web, and social media data

Betsy Page Sigman

Splunk Essentials

Leverage the power of Splunk to efficiently analyze machine, log, web, and social media data

Betsy Page Sigman

[PACKT] enterprise
PUBLISHING
professional expertise distilled

BIRMINGHAM - MUMBAI

Splunk Essentials

Credits

Author
Betsy Page Sigman

Reviewers
Mikael Bjerkeland
Dr. Benoit Hudzia
Diego Armando Ojeda
Russell Uman

Commissioning Editor
Dipika Gaonkar

Acquisition Editors
Richard Harvey
Rebecca Youé

Content Development Editor
Mohammed Fahad

Technical Editor
Utkarsha S. Kadam

Copy Editors
Veena Mukundan
Alfida Paiva

Project Coordinator
Purav Motiwalla

Proofreaders
Simran Bhogal
Maria Gould
Paul HIndle

Indexer
Tejal Soni

Graphics
Valentina D'silva

Production Coordinator
Shantanu N. Zagade

Cover Work
Shantanu N. Zagade

About the Author

Betsy Page Sigman is a distinguished professor at the McDonough School of Business at Georgetown University in Washington, D.C. She has taught courses in statistics, project management, databases, and electronic commerce for the last 16 years, and has been recognized with awards for teaching and service. Before arriving at Georgetown, she worked at George Mason University, the U.S. Bureau of the Census, Decision/Making/Information, the American Enterprise Institute, and the Social Science Data Center (now Roper Center) at the University of Connecticut.

Recent publications include a Harvard Business case study and a Harvard Business review article, articles in the *Decision Sciences Journal of Innovative Education* and *Decision Line*, and a case study in *Educause Review Online*. Additionally, she is a frequent media commentator on technological issues and big data.

About the Reviewers

Mikael Bjerkeland has over 10 years of professional experience in the IT industry, having worked with technologies such as real-time streaming of audio and video, SQL servers, Linux systems, and Cisco routing and switching.

He lives in Oslo, Norway, and currently works for Datametrix as a senior consultant in the fields of network management and big data, working with numerous government, private, and public organizations in the sectors of energy, banking, securities, and Internet service providers. He provides services that aid his customers to tame their vast amounts of untouched machine data in order to improve their operational efficiency.

Mikael received a Splunk Revolution award in October 2014 for his work on his Cisco Networks app, one of the most downloaded and highly rated apps for Splunk Enterprise.

When Mikael is not at work, he likes to go mountain biking and cross-country skiing, and tries to spend as much time as possible in nature with his family, with his cellphone turned off, and his coffee pot boiling on the fire.

Dr. Benoit Hudzia is a cloud/system architect working on designing the next generation of cloud technology as well as running the Irish operations for Stratoscale.

Previously, he worked as a senior researcher architect for SAP working primarily with the HANA enterprise cloud.

Benoit has authored more than 20 academic publications and is also the holder of numerous patents in the domain of virtualization, OSes, the cloud, distributed systems, and so on. His code and ideas are included in various SAP commercial solutions and open source solutions such as Qemu/KVM Hypervisor, Linux Kernel, and OpenStack.

His research currently focuses on bringing together the flexibility of virtualization, cloud, and high-performance computing (also known as the Lego cloud). This framework aims at providing memory, I/O, and CPU resource disaggregation of a physical server while enabling dynamic management and aggregation capabilities to Linux-native applications and Linux/KVM VMs using commodity hardware.

Diego Armando Ojeda is a software developer who specializes in many features that inhabit the web development realm, such as application frameworks, testing frameworks, libraries, helpers, utility belts, task runners, dependency managers, automation tools, and so on.

As a person who has too many interests, he enjoys mixing the experience and metaphors that he acquires from different fields with his programming activities. Hopefully, this turns his code-crafting activities into a diversified, entertaining, and unique journey that strives to achieve readable, organized, clean, creative, and valuable source code.

www.PacktPub.com

Support files, eBooks, discount offers, and more

For support files and downloads related to your book, please visit `www.PacktPub.com`.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.PacktPub.com` and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `service@packtpub.com` for more details.

At `www.PacktPub.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

https://www2.packtpub.com/books/subscription/packtlib

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can search, access, and read Packt's entire library of books.

Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print, and bookmark content
- On demand and accessible via a web browser

Free access for Packt account holders

If you have an account with Packt at `www.PacktPub.com`, you can use this to access PacktLib today and view 9 entirely free books. Simply use your login credentials for immediate access.

Instant updates on new Packt books

Get notified! Find out when new books are published by following `@PacktEnterprise` on Twitter or the *Packt Enterprise* Facebook page.

Table of Contents

Preface

Splunk Enterprise Software, or Splunk, is an extremely powerful tool for searching, exploring, and visualizing data of all types. Splunk is becoming increasingly popular, as more and more businesses, both large and small, discover its ease and usefulness. Analysts, managers, students, and others can quickly learn how to use the data from their systems, networks, web traffic, and social media to make attractive and informative reports.

This is a straightforward, practical, and quick introduction to Splunk that should have you making reports and gaining insights from your data in no time. Throughout the book, we have provided step-by-step instructions, pointers, and illustrations to help you on your way.

What this book covers

Chapter 1, *Introducing Splunk*, introduces you to Splunk Enterprise Software and its powerful capabilities.

Chapter 2, *An Introduction to Indexing and Searching*, explains indexing in Splunk and shows you how to do a simple search.

Chapter 3, *More on Using Search*, further develops your skills in using Splunk's search command.

Chapter 4, *Reports in Splunk*, shows you how to create reports and dashboards.

Chapter 5, *Splunk Applications*, explores the wide variety of Splunk apps and add-ons.

Chapter 6, *Using the Twitter App*, illustrates how to use the Twitter app for analyzing live Twitter data streams.

Chapter 7, *Monitoring and Creating Alerts in Splunk*, instructs you on how to monitor systems and create useful alerts that can help control processes and prevent problems.

What you need for this book

Most personal computers today can run Splunk easily. For more technical details see `http://docs.splunk.com/Documentation/Splunk/6.1.5/Installation/Chooseyourplatform`.

Who this book is for

Splunk Essentials is intended for the businessperson, analyst, or student who wants to quickly learn how to use Splunk to manage data. Perhaps you have heard about this technology that is being used quite often now in fields like systems analysis, cyber security, and machine data management. In a matter of hours, this book will help you understand how to bring in data of all types, store it, and use it to create effective reports and dashboards. It would be helpful to have a bit of familiarity with basic computer concepts, but no prior experience is required.

Conventions

In this book, you will find a number of text styles that distinguish between different kinds of information. Here are some examples of these styles and an explanation of their meaning:

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "We can include other contexts through the use of the `include` directive."

A block of code is set as follows:

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

Any command-line input or output is written as follows:

```
buttercupgames | timechart count by itemId limit=10
```

New terms and **important words** are shown in bold. Words that you see on the screen, for example, in menus or in dialog boxes, appear in the text like this: "Under **List by tag name**, click on "**Add new**.""

> Warnings or important notes appear in a box like this.

> Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you thought about this book – what you liked or disliked. Reader feedback is important for us as it helps us develop titles that you will really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com` and mention the book's title in the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide at `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you get the most out of your purchase.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. So if you find a mistake in one of our books – maybe a mistake in the text or the code – we would be grateful if you could report this to us. By doing so, you can save other readers from frustration, and also help us improve subsequent versions of this book. Hence, if you find any errata, please report them by visiting `http://www.packtpub.com/submit-errata`, selecting your book, clicking on the **Errata Submission Form** link, and entering the details of the errata. Once the errata are verified, your submission will be accepted, and the errata will be uploaded to our website or added to any list of existing errata under the Errata section of that title.

To view the previously submitted errata, go to `https://www.packtpub.com/books/content/support` and enter the name of the book in the search field. The required information will appear under the **Errata** section.

Piracy

Piracy of copyrighted material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors and our ability to bring you valuable content.

Questions

If you have a problem with any aspect of this book, you can contact us at `questions@packtpub.com`, and we will do our best to address the problem.

1
Introducing Splunk

Splunk, whose name was inspired by the process of exploring caves, or splunking, helps analysts, operators, programmers, and many others explore data from their organizations by obtaining, analyzing, and reporting on it. This multinational company, cofounded by Michael Baum, Rob Das, and Erik Swan, has a core product called **Splunk Enterprise**. This manages searches, inserts, deletes, and filters, and analyzes big data that is generated by machines, as well as other types of data. They also have a free version that has most of the capabilities of Splunk Enterprise and is an excellent learning tool.

> Throughout the book, I will be covering the fundamental, bare-bones concepts of Splunk so you can learn quickly and efficiently. I reserve any deep discussion of concepts to Splunk's online documentation. Where necessary, I provide links to help provide you with the practical skills, and examples so you can get started quickly.

To learn Splunk, it is important for you to first understand the following concepts:

- How to install Splunk for different operating systems and use it for the first time
- How Splunk works with big data
- Data sources for Splunk
- Events, event types, and fields in Splunk
- How to add data to Splunk

How to install Splunk

Downloading a free version of Splunk is easy and can be done by following the steps on the website.

Splunk setup instructions

Please be sure to use the appropriate instructions for your operating system. If you have any questions, please contact an instructor.

> Note that you can also find videos for setting up Splunk on Windows or Linux at `http://www.splunk.com/view/education-videos/` `SP-CAAAGB6`. This video shows you how to install version 6; in this chapter, you will install version 6.1.5.

Setting up Splunk for Windows

To install Splunk for Windows, please do the following:

1. Firstly, you need to go to `http://www.splunk.com/` and click on **Sign Up** in the top-right corner to create a Splunk.com account.

> Make note of your username and password. This is your Splunk.com account information and will be referred to as such from here on.

2. Once you have successfully created an **account** and have logged in, click on **Free Splunk** in the upper-right corner. Since there are sometimes slight changes in the instructions, remember that you can link to `http://www.` `splunk.com/download/`.

3. Choose your operating system, being careful to select 32- or 64-bit (whichever is appropriate in your case; most will select 64-bit), and then install version 6.1.5.

4. Follow the installation steps as instructed. Be sure you install as *local user* as you will be using data coming into your own machine.

5. Once Splunk has been successfully installed, open the application by selecting it from your start menu. Splunk opens in a web browser as it is a web-based application.

6. The first time you log in, you will need to enter `admin` as the username and `changeme` as the password. You will then be prompted to change the password.

> Please note that the account that uses `admin` as the username is different from the Splunk.com account you have previously created. So please use this one in order to get Splunk started.

7. Log out of Splunk and log back in. This will conclude the installation.

Splunk for Mac

To install Splunk on your Mac OS X, we will follow the following steps:

1. Go to `http://www.splunk.com/` and click on **Sign Up** in the top-right corner to create a Splunk.com account.

> Make note of your username and password. This is your Splunk.com account information and will be referred to as such from here on.

2. Once you have successfully created an account and have logged in, go to the **Products** menu and select **Splunk Enterprise**. On the resulting page (**What is Splunk Enterprise?**), click on the green **Download Splunk** button.

3. On the downloads page, click on the first download link (similar to `splunk-6.1.5-XXXXXX-macosx-10.7-intel.dmg`, where `XXXXXX` is replaced by a set of numbers) underneath the OS X downloads list.

4. Open the DMG (disk image) file after it finishes downloading. A window with a shortcut to **Install Splunk** should appear. Double-click on this icon to launch the Splunk installer.

5. Go through the installer. After the install completes, Splunk will prompt you to start the Splunk service that runs in the background and will eventually open a browser interface to Splunk.

> During installation, you may be prompted to install the command-line developer tools; if you see this message, you can click on **Not Now** and continue with the installation.

6. Log in with the default credentials (`admin : changeme`). Change the password if desired.

> These credentials are what you'll use to log in to Splunk on your machine and are different from the credentials of the Splunk.com account you previously created.

7. Congratulations! You can now access the Splunk dashboard. To shut down or restart Splunk, open the Splunk app in your `Applications` folder.

Starting up Splunk

Before getting into the practical details of Splunk, it is important to know what is really going on behind the scenes. When you start up Splunk, you are really starting up two different processes: splunkd and splunkweb. Here is the difference between the two:

- In the name splunkd, the *d* stands for daemon, meaning a process that is started up and then runs in the background, without interaction with the user. Splunkd is actually a C or C++ server that can process and index data even if it is streaming, or even if it is quickly moving data. It can also process and index static data files, of course. Splunkd is responsible for searching and indexing, which it does through the Splunk API, or **Application Programming Interface** (**API**). Everything that you do in Splunk requires the API, and it is also through the API that the two services communicate with each other.

- Splunkweb is the service we will interact directly with most often. It is a web interface, based on Python, which gives us a way to give commands to Splunk to get the data analysis we need. It also lets us start up and stop Splunk.

The functions of Splunk

Now it's time to look at the four main functions that Splunk carries out. These are collecting data, indexing data, searching for data, and analyzing data:

- **Data collection**: The process of collecting data with Splunk is enhanced, as its system makes it easy to get data from many different types of computerized systems, which are increasingly becoming the producers of most data today. Such data is frequently referred to as machine data. And since much of this is streaming data, Splunk is especially useful, as it can handle streaming data quickly and efficiently. Additionally, Splunk can collect data from many other sources. The use of specialized apps and add-ons to do this will be discussed in *Chapter 4*, *Reports in Splunk*.

- **Data indexing**: Before data can be searched, it needs to be indexed. To create an index actually requires two steps: parsing and indexing. Parsing, which is basically separating the data into events, involves several steps.

> Some of this discussion is beyond the scope of this text, but more details can be found at `http://docs.splunk.com/Documentation/Splunk/latest/Indexer/Howindexingworks`.

  In short, in addition to breaking up chunks of data, it adds metadata (or data about data), such as host (what device did the data come from), source (where did the event originate from), and sourcetype (the format of the data), as well as timestamps and other necessary information. The next step, indexing, breaks the events into segments that can subsequently be searched. It creates a data structure for the index and then writes the raw data and index files to disk. With this index structure, searches in Splunk can be quickly done on massive data sets.

- **Data searching**: This quick searching capability is extremely valuable for users of Splunk. Users often go to Splunk to find data they can use to answer questions. Splunk makes it easy to search on different dimensions of the data. Since Splunk indexes data before it is searched, the search process goes very quickly. Data searching in Splunk helps enable the analysis of data (which is described next).

- **Data analysis**: Lastly, Splunk can be used to quickly and easily analyze data. Its indexing creates a centralized data repository that can house data of many types from a variety of sources. Splunk has a variety of default data visualizations for reports and dashboards, and these can also be customized with little difficulty, thereby letting users to target analyses to improve decision-making.

Splunk and big data

Splunk is useful for datasets of all types, and it allows you to use big data tools on datasets of all sizes. But with the recent focus on big data, its usefulness becomes even more apparent. Big data is a term used everywhere these days, but one that few people understand. In this part of the chapter, we will discuss the aspects of big data and the terms that describe those aspects.

The three Vs

The following are the three key *V* words used to describe big data, as well as a discussion of how each of these helps to differentiate big data from other data:

- **Volume**: The most obvious of the descriptors is simply the size of data we are talking about. Instead of talking in millions (megabytes) or billions (gigabytes), we talk in terabytes, petabytes, or exabytes (adding many zeros as we go).

- **Variety**: This term refers to the fact that big data can include all kinds of data, and it often refers to data that is not traditionally structured. In reality, little data is completely without any structure, but there is a vast amount of data that is categorized as basically unstructured. Semi-structured or unstructured data, as well as structured data, can be searched and processed quickly using the methods of big data.

- **Velocity**: The last V refers to the speed at which the data comes into the system. An example of where velocity of data is a requirement is the Large Hadron Collider at CERN, located on the border between France and Switzerland. Every second, 600 million particles collide in its underground accelerator, and each day the CERN Data Center processes one petabyte of data describing what has happened. Scientists at CERN must thus deal with large amounts of data that needs quick processing.

Other big data descriptors

There are other terms that are necessary to understand when talking about big data. These are:

- **Streaming data**: Much of the data that is large and comes quickly does not need to be kept. For instance, consider a mechanical plant. There can sometimes be many sensors that collect data on all parts of the assembly line. The significance of this data is primarily to be able to alert someone to a possible upcoming problem (through noticing a bad trend) or to a current problem (by drawing attention to a metric that has exceeded some designated level); much of it does not need to be kept for a long period of time. This type of data is called streaming data, and Splunk, with its abilities to create alerts, allows organizations to use this data to make sure they prevent or act quickly on problems that can occur.

  > Later, in *Chapter 6*, *Using the Twitter App*, we'll use streaming Twitter data for analysis.

- **Latency of data:** The term latency in regards to data refers to delay in how speedily it is entered into the system for analysis. Splunk is able to analyze data in real-time with no latency issues when deployed on hardware that is sufficient to handle the indexing and searching workload. For example, if an alert goes off, a system can be immediately shut down if there is no latency in the data. If a denial of service attack is taking place, the system can be quickly used to figure out what is happening right at that very time.

- **Sparseness of data:** Splunk is also excellent for dealing with sparse data. Much data in retailing environments is considered sparse. Consider a store that has many products but where most people just buy a few of them on any given shopping trip. If the store's database has fields specifying how many items of a particular type have been purchased by each customer, most of the fields would be empty if the time interval under consideration was short. We would say then that the data is sparse. In Splunk, the sparseness of data in a search ranges from dense (meaning that a result is obtained 10 percent of the time or more) to sparse (from 0.01 to 1 percent of the time). This can also extend to super sparse, or, for a better definition, trying to find a needle in a haystack (which is less than 0.01 percent), and even to rare, which is just a handful of cases.

> More information on this can be found at `http://docs.splunk.com/Documentation/Splunk/6.1.5/Installation/HowsearchtypesaffectSplunkperformance`.

Splunk data sources

Splunk was invented as a way to keep track of and analyze machine data coming from a variety of computerized systems. It is a powerful platform for doing just that. But since its invention, it has been used for a myriad of different types of data, including machine data, log data (which is a type of machine data), and social media data. The various types of data that Splunk is often used for are explained as follows:

- **Machine data**: As mentioned previously, much of Splunk's data is machine data. Machine data is data that is created each time a machine does something, even if it is as seemingly insignificant as a tick on a clock. Each tick has information about its exact time (down to the second) and source, and each of these becomes a field associated with the event (the tick). The term machine data can be used in reference to a wide variety of data coming from computerized machines – from servers to operating systems to controllers for robotic assembly arms. Almost all machine data includes the time it was created or when the actual event took place. If no timestamp is included, then Splunk will to find a date in the source name or filename based on the file's last modification time. As a last resort, it will stamp the event with the time it was indexed into Splunk.

- **Web logs**: Web logs are invaluable sources of information for anyone interested in learning about how their website is used. Deep analysis of web logs can answer questions about which pages are visited most, which pages have problems (people leaving quickly, discarded shopping carts, and other aborted actions), and many others. Google, in early 2014, was registering as many as 20 billion websites each day, about which you can find more information at `http://www.roche.com/media/roche_stories/roche-stories-2014-01-22.htm`.

- **Data files**: Splunk can read in data from basically all types of files containing clear data, or as they put it, any data. Splunk can also decompress the following types of files: `tar`, `gz`, `bz2`, `tar.gz`, `tgz`, `tbz`, `tbz2`, `zip`, and `z` along with many other formats. Splunk can even process files when they are being added to!

- **Social media data**: An enormous amount of data is produced by social media every second. Consider the fact that 829 million people log in to Facebook each day (more information can be found at `http://newsroom.fb.com/company-info/`) and they spend, on average, 20 minutes at a time interacting with the site. Any Facebook (or any other social media) interaction creates a significant amount of data, even those that don't include many data-intensive acts, such as posting a picture, audio file, or a video. Other social media sources of data include popular sites such as Twitter, LinkedIn, Pinterest, and Google+ in the U.S., and QZone, WeChat, and Weibo in China. As a result of the increasing number of social media sites, the volume of social media data created continues to grow dramatically each year.

- **Other data types**: You will see the other data types listed when we add data to Splunk shortly.

Understanding events, event types, and fields in Splunk

An understanding of events and event types is important before going further.

Events

In Splunk, an event is not just one of the many local user meetings that are set up between developers to help each other out (although those can be very useful), but also refers to a record of one activity that is recorded in a log file. Each event usually has:

- A timestamp indicating the date and exact time the event was created
- Information about what happened on the system that is being tracked

Event types

An event type is a way to allow users to categorize similar events. It is field-defined by the user. You can define an event type in several ways, and the easiest way is by using the SplunkWeb interface.

One common reason for setting up an event type is to examine why a system has failed. Logins are often problematic for systems, and a search for failed logins can help pinpoint problems. For an interesting example of how to save a search on failed logins as an event type, visit `http://docs.splunk.com/Documentation/Splunk/6.1.3/Knowledge/ClassifyAndGroupSimilarEvents#Save_a_search_as_a_new_event_type`.

Why are events and event types so important in Splunk? Because without events, there would be nothing to search, of course. And event types allow us to make meaningful searches easily and quickly according to our needs, as we'll see later.

Sourcetypes

Sourcetypes are also important to understand, as they help define the rules for an event. A sourcetype is one of the default fields that Splunk assigns to data as it comes into the system. It determines what type of data it is so that Splunk can format it appropriately as it indexes it. This also allows the user who wants to search the data to easily categorize it.

Some of the common sourcetypes are listed as follows:

- `access_combined`, for NCSA combined format HTTP web server logs
- `apache_error`, for standard Apache web server error logs
- `cisco_syslog`, for the standard syslog produced by Cisco network devices (including PIX firewalls, routers, and ACS), usually via remote syslog to a central log host
- `websphere_core`, a core file export from WebSphere

(Source: `http://docs.splunk.com/Documentation/Splunk/latest/Data/Whysourcetypesmatter`)

Fields

Each event in Splunk is associated with a number of fields. The core fields of host, course, sourcetype, and timestamp are key to Splunk. These fields are extracted from events at multiple points in the data processing pipeline that Splunk uses, and each of these fields includes a name and a value. The name describes the field (such as the `userid`) and the value says what that field's value is (`susansmith`, for example). Some of these fields are default fields that are given because of where the event came from or what it is. When data is processed by Splunk, and when it is indexed or searched, it uses these fields. For indexing, the default fields added include those of host, source, and sourcetype. When searching, Splunk is able to select from a bevy of fields that can either be defined by the user or are very basic, such as action results in a purchase (for a website event). Fields are essential for doing the basic work of Splunk – that is, indexing and searching.

Getting data into Splunk

It's time to spring into action now and input some data into Splunk. Adding data is simple, easy, and quick. In this section, we will use some data and tutorials created by Splunk to learn how to add data:

1. Firstly, to obtain your data, visit the tutorial data at `http://docs.splunk.com/Documentation/Splunk/6.1.5/SearchTutorial/GetthetutorialdataintoSplunk` that is readily available on Splunk.

2. Here, download the folder `tutorialdata.zip`. Note that this will be a fresh dataset that has been collected over the last 7 days. Download it but don't extract the data from it just yet.

3. You then need to log in to Splunk, using `admin` as the username and then by using your password.

4. Once logged in, you will notice that toward the upper-right corner of your screen is the button **Add Data**, as shown in the following screenshot. Click on this button:
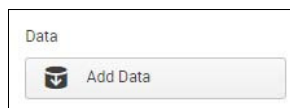
Button to Add Data

5. Once you have clicked on this button, you'll see a screen similar to the following screenshot:
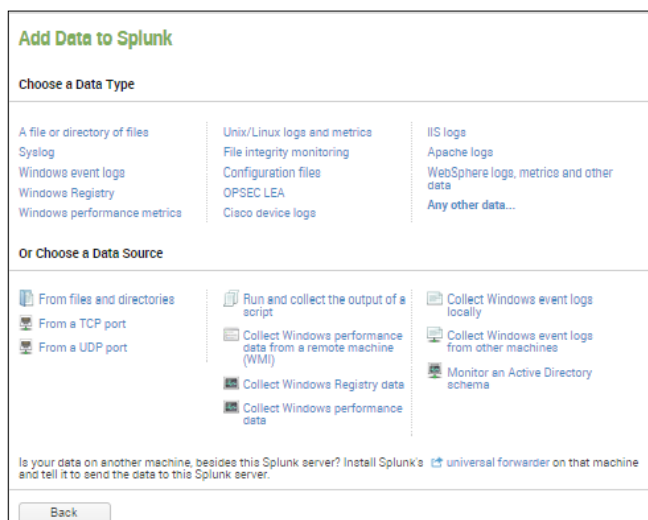
Add Data to Splunk by Choosing a Data Type or Data Source

6. Notice here the different types of data that you can select, as well as the different data sources. Since the data we're going to use is a file, under **Or Choose a Data Source**, click on **From files and directories**.

7. Once you have clicked on this, you can then click on the radio button next to **Skip preview**, as indicated in the following screenshot, since you don't need to preview the data now. You then need to click on **Continue**:

Preview data

You can download the tutorial files at: `http://docs.splunk.com/Documentation/Splunk/6.1.5/SearchTutorial/GetthetutorialdataintoSplunk`

8. As shown in the next screenshot, click on **Upload and index a file**, find the `tutorialdata.zip` file you just downloaded (it is probably in your **Downloads** folder), and then click on **More settings**, filling it in as shown in the following screenshot. (Note that you will need to select **Segment in path** under **Host** and type `1` under **Segment Number**.) Click on **Save** when you are done:

Can specify source, additional settings, and source type

9. Following this, you should see a screen similar to the following screenshot. Click on **Start Searching**. Even though we won't really do a search until the next chapter, we will look at the data now:
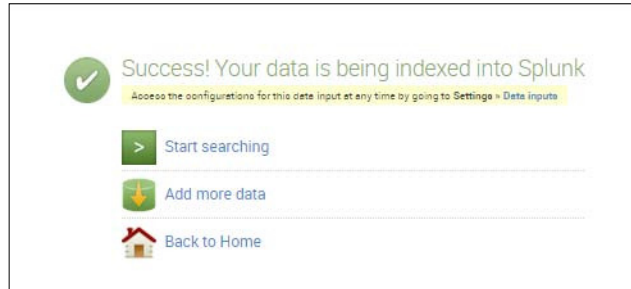
You should see this if your data has been successfully indexed into Splunk.

10. You will now see a screen similar to the following screenshot. Notice that the number of events you have will be different, as will the time of the earliest event. At this point, click on **Data Summary**:
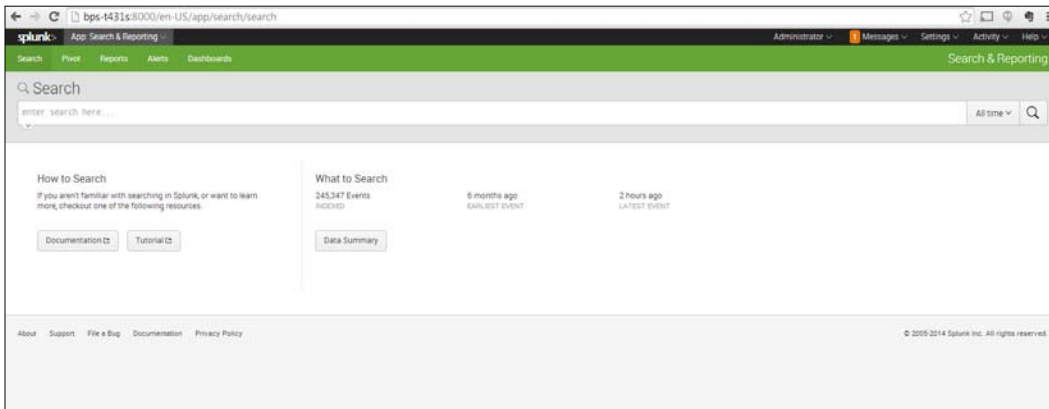
The Search screen

11. You should see the **Data Summary** screen like in the following screenshot. However, note that the **Hosts** shown here will not be the same as the ones you get. Take a quick look at what is on the **Sources** tab and the **Sourcetypes** tab. Then find the most recent data (in this case **127.0.0.1**) and click on it.
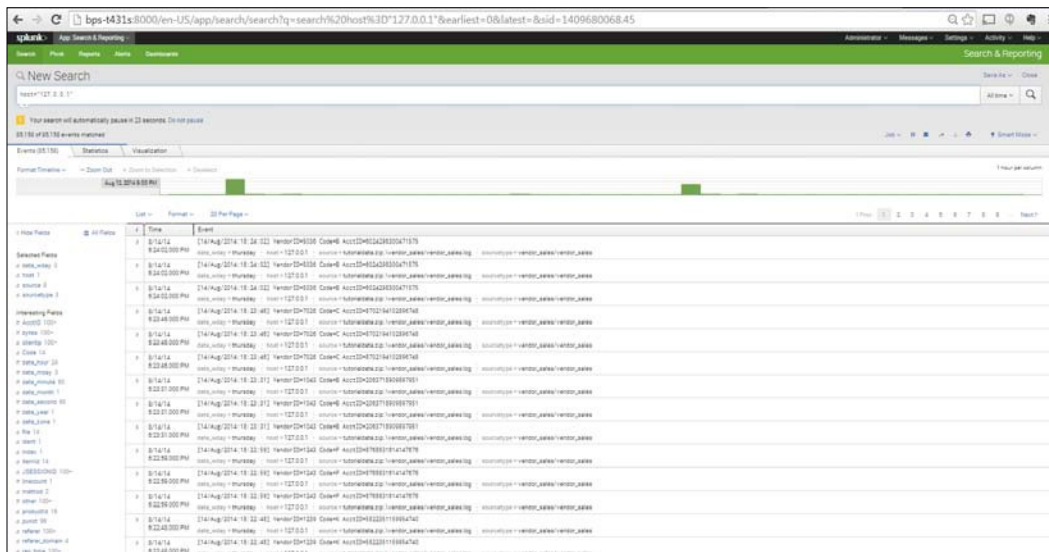
Data Summary, where you can see Hosts, Sources, and Sourcetypes

12. After clicking on the most recent data, which in this case is bps-T341s, look at the events contained there.

> Later, when we use streaming data, we can see how the events at the top of this list change rapidly.

13. Here, you will see a listing of events, similar to those shown in the following screenshot:

Events lists for the host value

14. From the preceding screenshot, you will notice the list of fields on the left-hand side. We will explore how to search for these fields in the next chapter. For now, you can click on the Splunk logo in the upper-left corner of the web page to return to the home page. Under **Administrator** at the top-right of the page, click on **Logout**.

Summary

In this chapter, we have learned about big data and how it can be stored, indexed, searched, and analyzed using Splunk. We have also followed steps to bring the data from a file into Splunk and then examine it.

In the next chapter, we'll go further with analyzing this data and learn how to conduct searches using Splunk.

2
An Introduction to Indexing and Searching

In the previous chapter, we showed you how to bring in data from different sources and index it. Data must be turned into information and made relevant before we can use it successfully, as raw data in files or streams won't help us answer the questions that arise while analyzing the data for our businesses or organizations. We need to collect the data that we are interested in before we can analyze it. And this is where Splunk's capabilities shine.

In this chapter, we will cover these important next steps for using Splunk:

- Collecting data to search
- How Splunk indexes data
- Using indexed data
- Specifying a sourcetype
- SPL and what it is
- How to perform your own simple search

Collecting data to search

In the previous chapter, we showed you how to bring data from a file into Splunk. We also discussed how data from virtually any source can be brought into Splunk. The following diagram shows the various types of data (such as Twitter, Facebook, RSS, network, and many others that are pictured) that can be easily integrated into Splunk, then searched, added to other data, monitored, and then used for creating dashboards, reports, and other kinds of analyses. Notice that the storage capabilities of Splunk are also included in the screenshot:

Many types of data can be used with Splunk

(Source: `http://www.businessinsider.com/investors-are-eating-up-these-two-enterprise-tech-ipos-heres-why-2012-4`)

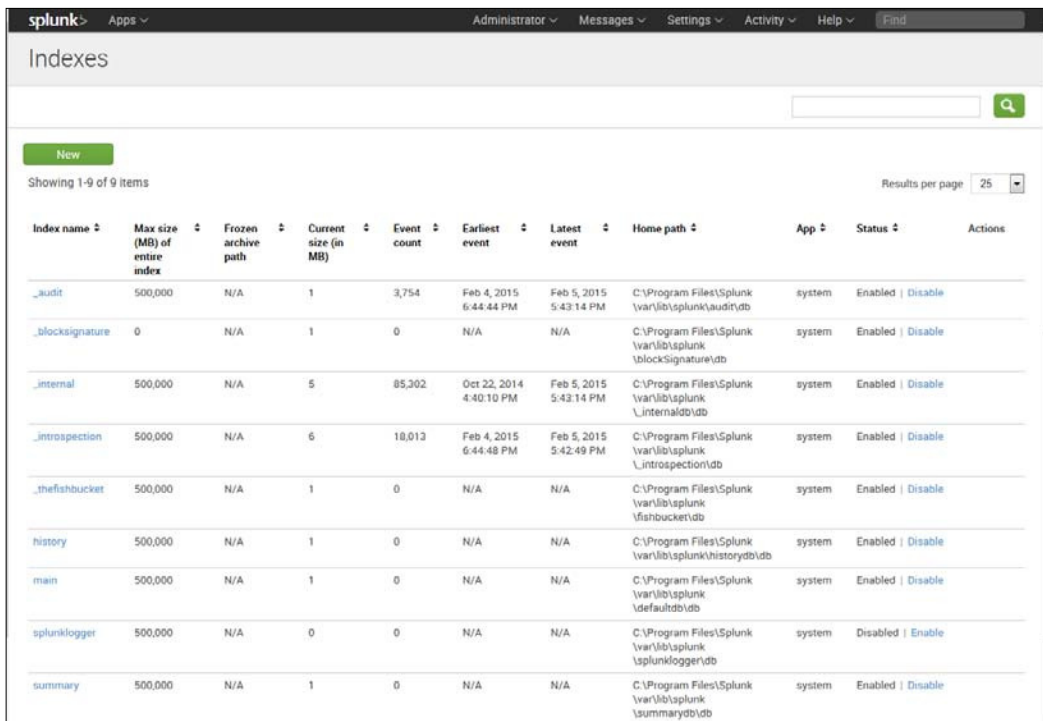Almost any kind of data can be entered into Splunk, and then stored, searched, analyzed, and reported on. Additionally, you will also see a logo labeled **Hadoop**. You may have even heard this term before, in connection to big data. Hadoop, an Apache open source software package, is a method of storing and analyzing big data that has a lot in common with Splunk. Hadoop and Splunk can work together with the application called Hunk, which we'll talk about later in *Chapter 5*, *Splunk Applications*.

Before going on, it is important to note that one of the most important capabilities of Splunk is that you can bring in large amounts of data from several different sources and easily store it and analyze it in one location.

Indexing data with Splunk

When we processed the data file in the previous chapter, we uploaded the data and Splunk processed and indexed the data. It is worthwhile to examine a bit further what happens when indexing takes place:

1. To create an index actually requires two steps: parsing and indexing. The parsing part includes the adding of metadata that always includes the host, source, and sourcetype. The indexing portion takes the events, splits them into searchable segments, and finally creates the index and raw data files.

2. After this happens, the data can then be easily searched through Splunk. The following screenshot shows how the data is brought into Splunk by forwarders. A forwarder takes data from a source, such as a web server, and then sends it to a full instance of Splunk:
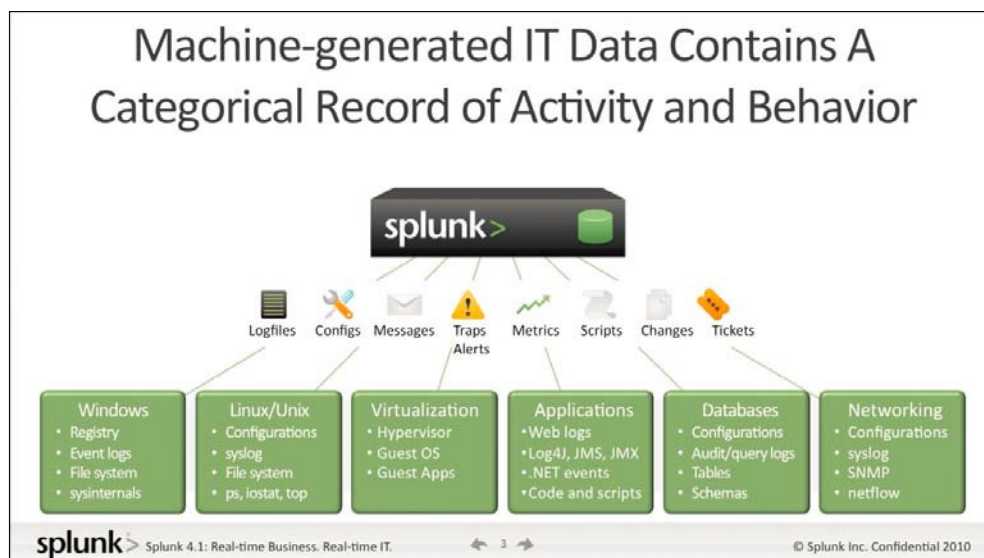
This diagram shows how Splunk uses forwarders to take data from complex IT infrastructures and then sends it to be indexed and searched.

(Source: `http://www.businessinsider.com/investors-are-eating-up-these-two-enterprise-tech-ipos-heres-why-2012-4`)

Using indexed data

Once you have indexed a file successfully, as we did in *Chapter 1*, *Introducing Splunk*, it will be listed with any other indexes that have already been created, and you can now do searches on it.

Viewing a list of indexes

To see a list of your indexes, follow the steps given next:

1. First, visit the home page (a quick way is to just click on the Splunk icon).
2. Find the **Settings** drop-down menu.
3. To finally view your indexes, under **Data**, select **Indexes**.

   You will see a screen like the one shown here:

Listing of Indexes

You will see a number of internal indexes, which are preceded by an underscore. These indexes include the logs and metrics that record Splunk's internal processing. Notice that the non-internal indexes listed here are history, main, splunklogger, summary, and Twitter. The `main` index is often selected as a default index. The `history` and `splunklogger` indexes were used for previous versions, but are not generally used now. The `summary` index stores events that have been aggregated using a transforming command to set up searches over long time periods. And the `Twitter` index is created when you use the Twitter app, as we will do in *Chapter 6, Using the Twitter App*.

Bringing in indexed data

We need to bring in the indexed data before we can search it. If we do not specify an index, index=main, which is set here to be searched by default via the indexes searched by the default setting, will be assumed. To bring in all the indexed data, we could specify `index=*`. If we want to bring in the `Twitter` index (which you will create in *Chapter 6, Using the Twitter App*), we can just specify `index=twitter`.

When we processed the data from our file in *Chapter 1, Introducing Splunk*, it was indexed by default. So we do not have to specify this index when we use it as our data source and go on to learn more about how to search in Splunk.

Specifying a sourcetype

Identifying a sourcetype for data is important because it tells Splunk how to format the data. The sourcetype is one of the default fields assigned to each event that is processed. Splunk uses it to decide how it is going to process your data. The correct sourcetype is usually assigned automatically when indexing data, for Splunk comes with many predefined sourcetypes.

One such sourcetype is `access_combined`. Using this, Splunk can analyze combined access log files, the types that are part of the massive amount of data exhaust created by web servers such as Microsoft IIS or Apache. Some common sourcetypes include the following:

Sourcetype	Used for
access_combined	A standardized format for text files used by HTTP web servers when generating server log files
cisco_syslog	Cisco standard system logs
apache_error	Errors

Sometimes the `access_combined` sourcetype specifies `_wcookie`, which indicates that each cookie set during an HTTP request is logged. The data we brought in and indexed in *Chapter 1*, *Introducing Splunk*, was specified `access_combined_wcookie`. To specify this particular sourcetype, type the following into the search bar:

sourcetype=access_combined_wcookie

This will pull up the web server logs with this sourcetype so you can then use them for analysis.

When adding custom data formats, such as logs from applications built in-house, you can specify a descriptive sourcetype for the technology as the sourcetype is what is being used to differentiate the data type. For Cisco iOS devices, you can use **sourcetype=cisco:ios**.

What is Search Processing Language (SPL)?

After we have our data indexed, we can begin to search. The default application for Splunk is the search application. It is assumed that you are doing a search unless you indicate otherwise. Searches are made using the **Search Processing Language** (**SPL**). Through search, Splunk lets the user comb through the indexed data to find what he or she needs for answering questions.

In the simplest of terms, if you only put the term failed in the search box, for instance, it knows you want to do a search and will automatically search for **failed** anywhere in the data, and will return each event that fits with failed highlighted.

Using pipes when processing data with Splunk

However, SPL can be used to do much more advanced searches and analyses as well. Pipes are a way to do this. The pipe character (|) can be used to chain together different commands in a search. In the previous simple search and in our following search, a search is implied in the first pipe, but the term **search** itself is left out. In other words, in the following search, we could say `buttercupgames` or search `buttercupgames` and it means the same thing. There are many other commands that can be used as well, and they are listed and discussed as follows. Consider the following piped command:

buttercupgames | timechart count by itemId limit=10

The command following the pipe character acts on the data after it comes from the previous pipe. Hence, a pipe can refer to either the pipe character or the command between pipes. So, as our first pipe is the term `buttercupgames`, with the search term implied, all the events containing the word `buttercupgames` will be gathered; then the second pipe's instructions about creating a timechart showing the count by `itemId` will be carried out on that gathered data. We'll cover these more advanced processes in the chapters ahead.

> **Downloading the example code**
>
> You can download the example code files from your account at `http://www.packtpub.com` for all the Packt Publishing books you have purchased. If you purchased this book elsewhere, you can visit `http://www.packtpub.com/support` and register to have the files e-mailed directly to you.

Types of SPL commands

SPL commands can be organized into groups as shown in the following table . We will now go through each of these groups:

Purpose of Command	What it Does	Actual Commands

Filter	Reduces results to a smaller set.	`search`
		`where`
		`dedup`
		`head`
		`tail`

Sort	Orders the results and can also be used to limit the number of results.	`sort`
Group	Puts those results like members together in groups to better see patterns in the data.	`transaction`

Report	Takes results of a search and summarizes them for a report.	`top / rare`
		`stats`
		`chart`
		`timechart`

Other	Included in this group are those that allow you to filter out fields, modify fields, or add fields to your results.	`fields`
		`replace`
		`eval`
		`rex`
		`lookup`

In the following tables, we discuss each type of command, what it does, and give examples of how it is used.

Filter commands

Search, of course, is included as a filter command as it results in a smaller data set. The other filter commands take the results from a search and then further reduce them based on the commands you use:

Command	What it Does
search	This is the most important command Splunk has. It is the default command as well, so there is no need for you to type it in the search box. However, if you do another search after one or more pipes, you do need to include the word search in the command. We'll learn more about search in the section *How to perform simple searches*.
where	This command takes an expression, such as **where monthly_sales > avg_mon_sales**, and evaluates it. If it is **TRUE**, it is kept in the search results.
dedup	This command only keeps the first x results for each search. **dedup source** returns only the first result for each source. Building on this, **dedup 3 source** returns only the first three results for each source.
head/tail	These commands look for a specified number of searched terms, counting from the top or bottom of the list of events. The head command returns the first x results. **head 10** returns the first ten results. The tail command returns the last x results. **Tail 10** returns the last ten results.

The sort command

This group contains just the sort command. Here are some examples of sorts and what they do:

Command	What it Does
`sort 0 anyfield`	This command sorts in ascending order by userid (A to Z, 1 to infinity, depending on whether the anyfield field is a number or name). The 0 means that all results are sorted, not just the default 10,000.
`sort 1000 fieldone –fieldtwo`	Sorts by fieldone in ascending order, then by fieldtwo in descending order, and returns up to 1,000 results.
`sort –fieldone, +fieldtwo`	Sorts by fieldone in descending order, and fieldtwo in ascending order. This command will return 10,000 results (the default).

The grouping command

There is only one grouping command that we describe in the following table, although there are others. It is an important one, as it allows you to group selected events together (note that grouping can also be done through the stats command):

Command	What it Does
`transaction`	A transaction takes selected events and groups them together. `transaction ipaddress host maxspan=60s` groups together all events with the same combination of ipaddress and host, where the first and last event are no more than 60 seconds apart.

Reporting commands

The reporting commands listed here are the most important ones. They are vital to performing analyses in Splunk and also aid in creating dashboards and reports:

Command	What it Does
`top/rare`	The **top** command returns the values that occur most often, as well as their counts and percentages. The default is 10. **top source** returns a list of the top 10 sources, including their counts and percentages. **top 15 source, host** returns a list of the 15 most frequent source-host combinations.
`stats`	The **stats** command returns the results of statistical calculations. It can return a single number, as in **stats dc(source),** which gives a distinct count that includes each different source. Or it can return a table, as in **stats max(kbps) by host**, which gives the maximum speed for each host.
`chart`	The **chart** command is used for creating tables of data. In each chart, the x-axis labels are indicated by either **over** or **by**. **chart count(fail*) over host** creates a chart showing the count of events that include the phrase "fail" plus anything after that (for example, "failed", "failure", and the like) for each value of host. For more on the **chart** command, go to `http://docs.splunk.com/Documentation/Splunk/6.1.3/SearchReference/chart`.
`timechart`	The `timechart` command produces a chart with time as the x-axis. **timechart span=1d avg(delay) by host** creates a chart showing the average delay by each host during a 1 day period.

Other commands

These commands are also commonly used for analysis in Splunk. Several of those listed help subset and modify fields for targeted analyses. The lookup command links a field to a lookup table, from which results can be identified and output:

Command	What it Does
`fields`	The `fields` command is used to remove fields from a search. Thus, the command `fields field1 field3` keeps only the fields labeled `field1` and `field3`.
`replace`	The `replace` command substitutes one value for another. In the statement `replace 0 with Check, 9 with Warning in Status`, status values of `0` are replaced with `Check` and status values of `9` are replaced with `Warning`.
`eval`	The `eval` command makes calculations and puts them into a new field. This code, `eval Depth=case(depth<=3, "Low", depth>3 AND depth<=10, "Medium", depth>10, "High")`, creates a new field, `Depth`, and uses the `case` function to assign the labels `Low`, `Medium`, or `High`, depending on the value.
`lookup`	The `lookup` command calls up a lookup table that lets you add new field values. In the statement, `lookup status_desc status OUTPUT description`, the field, `status`, is looked up in the `status_desc` lookup table and the corresponding description is output.

How to perform simple searches

Now we'll go on to do a couple of simple searches. In *Chapter 1*, *Introducing Splunk*, we brought in data from a file. This data included information on events that Splunk created for a fictional online store that sells games. It includes logs from the web server as well as MySQL, a backend database system. We'll do a simple search of these logs here, and will do more advanced searches in the chapters ahead. The steps and screenshots for this process are presented as follows:

1. First, you need to start up Splunk. Go to your start menu and activate Splunk. Notice that on the right, you should have a substantial number of events listed. These are the events that were indexed when you read in the file in *Chapter 1*, *Introducing Splunk*.

2. You are interested in looking at all the events involving Buttercup Games, one of the games you have at this fictional online store. Type **buttercupgames** into the **Search** box shown in the following screenshot:
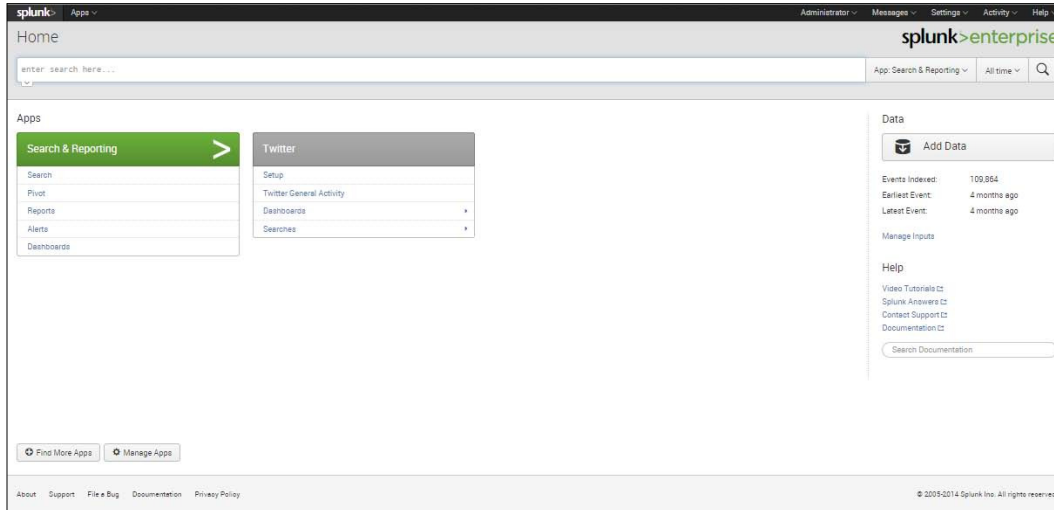
Enter buttercupgames into the search box

3. You will see something like the events listed in the following screenshot. The actual events will be different, as Splunk updates the fictional data on this site. But the events you see will have a similar structure. Incidences of the search term are highlighted in yellow. Events are listed in descending order by time, with the highest (newest) timestamp first:

Notice that when you search, the search term will be highlighted in yellow in each event and the events are listed in descending order by time, or with the highest (newest) timestamp first

4. Now add to the search itself. Next to **buttercupgames**, type **date_ wday="wednesday"**. Your results will look similar to the screenshot for just **buttercupgames**, but you'll notice that each of the events shows **date_ wday=wednesday**.

Use quotes when searching for a specific value in a specific field:

When we used the search term **buttercupgames**, we did not specify the field in which we were looking, so everywhere that **buttercupgames** occurred was picked up. When we look for **date_wday="wednesday"**, we are looking for a specific value in a specific field, so we need to specify the field we are looking for as well as the value. It is a good idea to put the search term in quotes, but this is only required if the text you are searching for contains whitespaces or special characters.

The next search will show the difference between using the implied AND and specifying OR in a search. This is important to understand as you continue to learn about searching in Splunk:

1. Suppose that you want to try to track down all instances of failed passwords that were coming into the system.

2. Click on the Splunk icon in the top left-hand corner of the screen to go back to the home page.

3. If you type in the word `fail`, you might be surprised when you get no results. The reason for this is that if you just type in `fail`, it looks only for that, and if it does not find those specific letters, followed by a space, it will not return anything. So, it will miss `failed` or any other version of `fail` that you might think it would pick up.

4. Now type in `fail*` and search and you will get a different result. This time, you'll see thousands of events that show `failed`. Since you are interested specifically in failed passwords, you decide to search on the term `failed password`. Note the number of events in the upper left-hand corner.

> There is an implied AND when you do a search in Splunk. To get results for two different terms, be sure to use OR.

5. Imagine that you want to look at the events where there was a failed password for users **myuan** and **harrison**. If you put in **failed password myuan harrison**,you will get no results because of the implied AND (you cannot have a user who is both myuan and harrison at the same time). But if you put an OR between myuan and harrison, that is, **failed password myuan OR harrison**, Splunk returns all results for failed passwords for either user.

> If, for some reason, you get no results for either of these users, it probably means that the fictional users had no events. In this case, just do a simple search on the term **user** and select two other users from the events you see and go through Step 5 given previously.

Summary

In this chapter, we have covered the way data is collected by Splunk, indexed, and prepared for searching. We've learned about the different commands that make up the **Search Processing Language** (**SPL**) and the way commands can be piped together. Lastly, we've learned how to do some simple searches, which prepares us to do more advanced analysis in the chapters ahead.

In the next chapter, you will go on to sort, filter, change, and create fields to do more advanced analysis in Splunk.

3
More on Using Search

In the previous chapter, we learned how to collect and index data to prepare it for searching, and we also did a simple search. In this chapter, we will cover more about how to use search and other commands to analyze our data. In a nutshell, we will cover the following topics:

- More on search
- Doing a count with the stats command
- Other stat functions
- Using the eval command
- Using the `timechart` command
- Visualizations
- Using the top command

More on search

We did a simple search at the end of the previous chapter. Before going on to other commands, however, let's examine how we can do other types of searches. There are several rules to be aware of when doing searches:

1. Searches are not generally case sensitive. Hence, for instance, to require the exact case of each variation of the word term, enclose it in CASE(term), CASE(Term), or CASE(TERM).

2. There is an implied `AND` when you use the search command (or the implied search command at the start of each entry in the search bar). For example, when you put `log error` in the search bar, you will only see events listed that have both `log AND error` in them.

3. If you want to search for an exact phrase, you need to put it in quotes. Inserting `log error` in the search bar (for example) will yield events with that exact phrase. The term events with log errors will not appear. Remember these points when designing searches:

    °   If you want to search only a specific field, you need to specify that field. Otherwise, you will be searching all fields. Since you are not always aware of what can appear in other fields, the results can sometimes be surprising if you do not specify the fields you want to search. So, if you want to search the text field for the terms `log` or `error` specify:

        `text=*log*  OR text=*error*`

    °   Note that the wildcard asterisks signal the search to bring in every event where the strings above appear, including incidences such as bad error or login. If you only want to search for `log` and `error` as separate words, then leave out the asterisks.

4. If you only want to consider events where the text field includes both `log AND error`, do the following:

    `text=*log* text=*error*`

5. The Boolean operators that Splunk supports, that is, `AND`, `OR`, and `NOT`, must be capitalized.

Doing a count
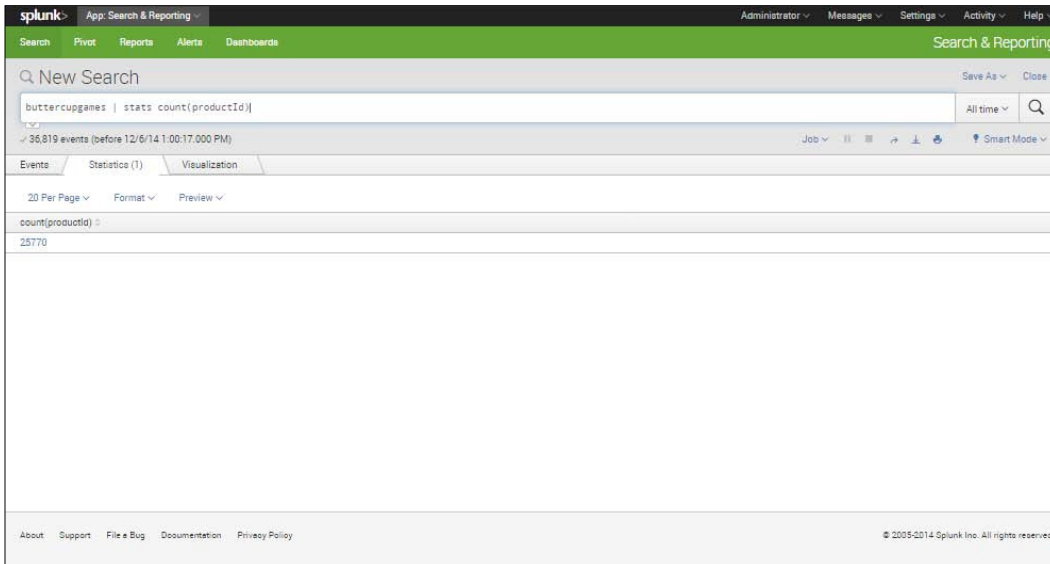
Recall that in *Chapter 2, An Introduction to Indexing and Searching*, we searched for the term, **buttercupgames** and found that every occurrence of it was highlighted. Now we want to look among the events for **buttercupgames** and get an idea of how many of each product they are selling, by doing `count` on `productId`. A count is done using `stats`, and the command is `stats count(X)`, where `X` is a field. If you are looking for the count of events, then the parentheses may be omitted, but if you're looking for the count of each instance of a field value, you'll need parentheses. If you have 100 events where the `productID` field is in `100` of them but the `customerid` field is only in 96 of them, `stats count(customerId) BY ProductId` would yield a different result than `stats count BY productID`.

> 📝 Field names are case sensitive. `HOST` is not the same as `host`, so be careful when specifying field names.

Notice that when we enter the following:

buttercupgames | stats count(productId)

(Notice that the field productId has a capital I.), we get the following:

Obtain a count of all events with a productId
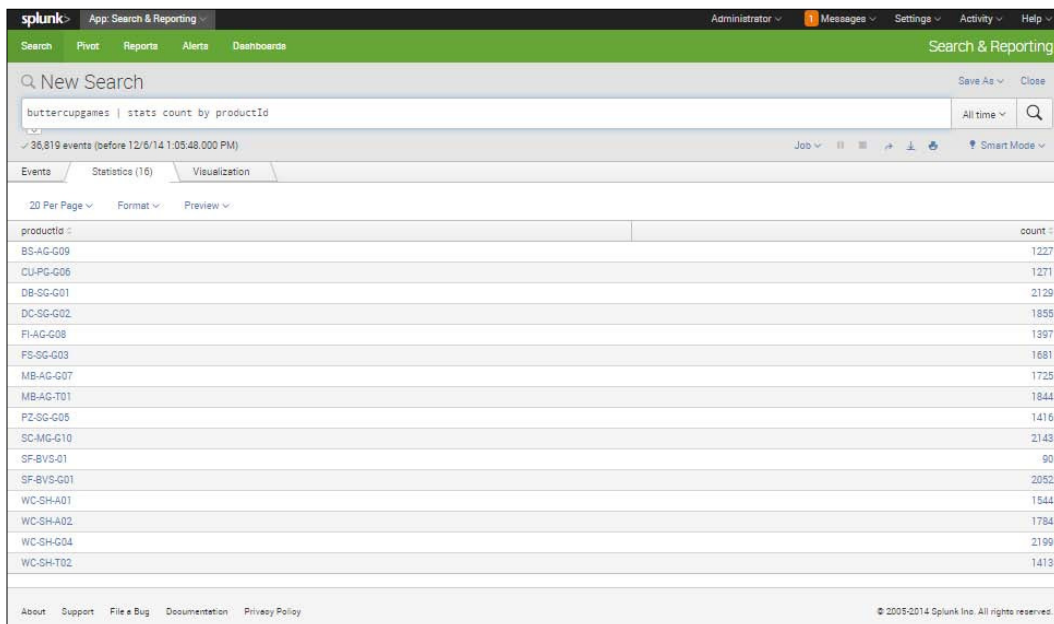
We can see that the count of all the events with `productId` is shown. However, though useful, this is not what we are looking for here.

Creating a count broken down by field values

We want to do a count for each value of the `productID` field. So this time we enter the following:

```
buttercupgames | stats count by productId
```

Now, as shown in the following screenshot, we get the individual counts for each `productId` value, so we know precisely how many were sold during the time period under consideration. We can use this information to see how well each category of `productId` did during that time period:

Obtain counts for values of productId

Other stat functions

There are numerous other stat functions available. Here are some of the most common ones:

Stats function	Description
avg(X)	Returns the average value of field X
dc(X)	Returns the distinct count of field X
earliest(X)	Returns the earliest value of field X, chronologically
last(X)	Returns the last seen value of field X
latest(X)	Returns the latest value of field X, chronologically
list(X)	Returns the list of all values of field X as a multi-value entry
max(X)	Returns the maximum value of field X
median(X)	Returns the middle value of all values of field X
min(X)	Returns the minimum value of field X
mode(X)	Returns the most frequent value of field X
perc<X>(Y)	Returns the X-th percentile value of field Y
range(X)	Returns the range (max-min) of field X
stdev(X)	Returns the standard deviation of field X
sum(X)	Returns the sum of all values of X
values(X)	Returns the list of all distinct values of field X as a multi-value entry
var(X)	Returns the sample variance of field X

Using the eval command

The eval command is one of the most useful Splunk search commands. Its usefulness is due to the fact that it can be used to calculate almost any expression you can think of. There are also numerous eval functions that can be used in conjunction with the command. A few of them will be shown to you here, but there are many more in the Splunk documentation:

Eval function	Description	Example
`case(X, "Y", . . .)`	Using pairs of arguments, X and Y, where X is TRUE, return Y.	`case(error == 404, "Not found", error == 200, "OK")`
`ceil(X)`	Gives the ceiling of a number.	`ceil(2.2)`
`if(X,Y,Z)`	If X is TRUE, result is Y. If X is FALSE, result is Z.	`if(error ==404, "Not found", "Found")`
`len(X)`	Returns number of characters in the string field.	`length(field)`
`lower(X), upper (X)`	Returns lowercase, uppercase.	`lower(username), upper(username)`
`round(X,Y)`	Rounds X to Y decimal places. If no Y is given, round to integer.	`round (3.5)`

Combining stats with eval

Now we will try an example using stats and eval commands. Here, we want to look for the counts of how a web page was accessed, whether by using GET or POST.

We enter the following into the search box:

```
sourcetype=access_* | stats count(eval(method="GET")) as GET,
count(eval(method="POST")) as POST by categoryId
```

> Be very careful here to use the exact capitalization for the field categoryId.

Here we are requesting all events that indicate a web page was accessed. Then we count up the number of results that used the GET and POST method, and then display those results based on categoryId of products, as shown in the following screenshot:
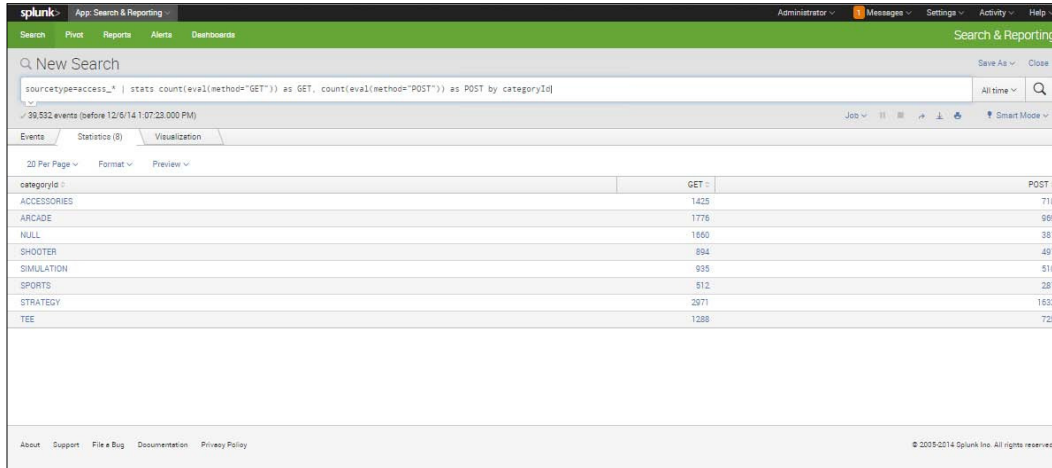
Determining Counts for "GET" and "POST" by CategoryId

Using the timechart command

We are also interested in figuring out exactly what was sold when. Are there certain days when we sell more of one product and others when we sell more of another?

To help us answer this question, we can specify the search command as shown in the following screenshot. Here we use the timechart command. This command creates a time series chart and a table of statistics. Notice that here we have set the timespan to 1 day by using the span=1d attribute. But we can use other timespans as well for analysis, with a different granularity.

Enter the following in the search bar:

```
buttercupgames  | timechart span=1d count by productId
```

After inserting this search, and looking at the **Statistics** tab, we will be able to see a breakdown of **productId** sales by date, as shown in the following screenshot:
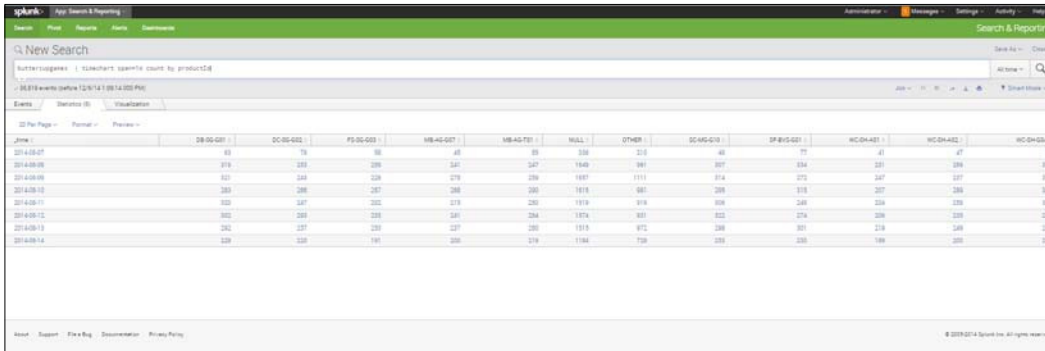
Time Chart Spanning 1 Day Showing Counts of ProductId

Visualizations

Raw counts can give us some idea, but it is often more useful to see a chart. So when we click on the **Visualizations** tab, we get a better picture:
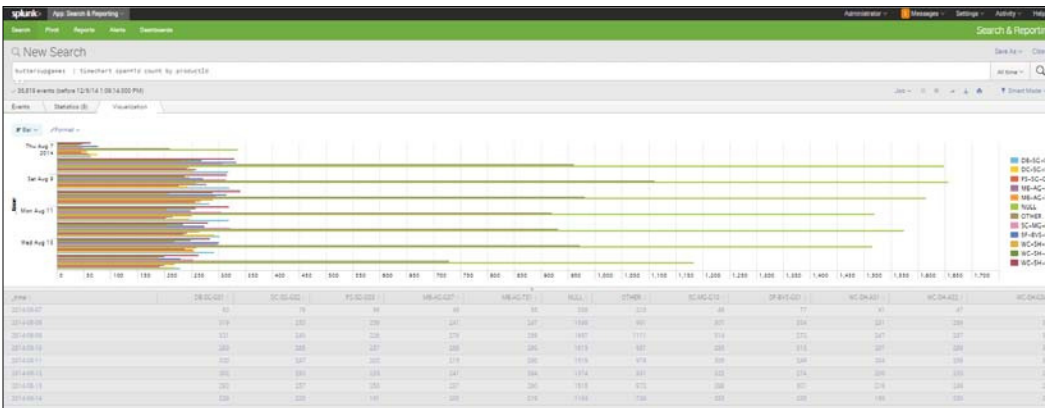
Timechart Spanning 1 Day, Showing Counts by ProductId

Changing Format to Column Chart

In the previous screenshot, the NULL category (colored green in the chart) dominates the others. We can change the format of the timechart by selecting **Column** in the drop-down menu immediately above the chart. This format makes it easier to see the distinct categories of productId. We thus get the following results:

Timechart Presented in Column Chart format

The top command

If we want to better compare our sales for the top **ProductIds** over time, we can use the top command to do so. So, in the search bar, we enter the following:

buttercupgames | top 5 productId

And the resulting screen appears, as shown in the following screenshot:

Using top Command to List Top 5 Counts of ProductId

We can then click the **Visualizations** tab and see the following chart:
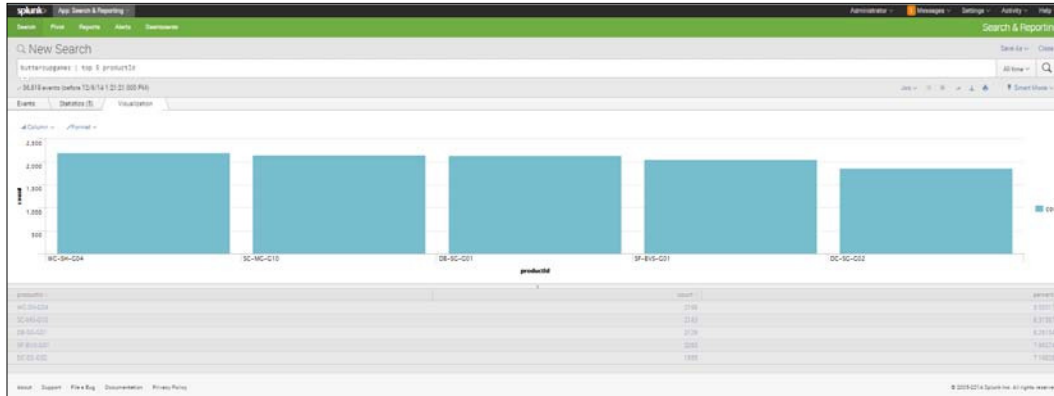
Column Chart of Top 5 ProductIds

Another way to use the top command is to pull out the top instance for a particular category. In the following screenshot, you can see the top command being used to pull out and list the top action used by each of the listed **referer_domain** values. We type in the following for this result:

```
sourcetype=access_* | top 3 action by  referer_domain
```

This code requests the events where the sourcetype is `access_*` (meaning that the web server was accessed), and then lists the top 3 actions for each referring domain. Notice that the default name count is specified at the top of the counts for each of the actions for each `referer_domain`. If you wanted to name it something else (such as **Total**), you could specify the following:

```
sourcetype=access_* | top 3 action by  referer_domain countfield=Total
```

The resulting window appears as shown in the following screenshot:

Top 3 Actions for referer_domain with Total Counts

Charting by the day of the week

You might also be interested in the top productId purchased on each of the seven weekdays. To get those results, you can enter the following:

```
buttercupgames | top 1 productId by date_wday
```

When you do this, you get the following result:

Top Product ID for Each Weekday

Putting days of the week in an alphabetical order
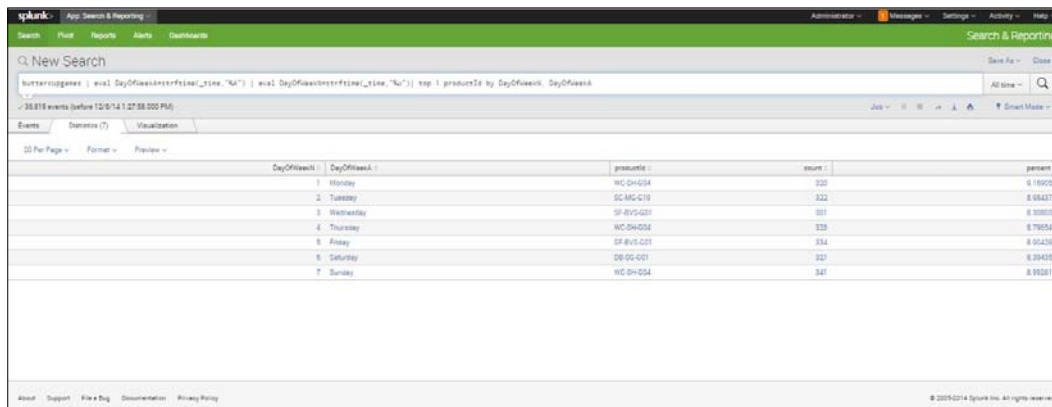
The previous screenshot is interesting, but you would probably like to format the result so that the days of the week are in the normal weekday order instead of in alphabetical order. To do this, you need to create two new fields: `DayOfWeekA`, that represents the alphabetic day of the week, and `DayOfWeekN`, that represents the numerical day of the week. (We're inventing the names of our new fields here.) We use a function, `strftime`, to evaluate the `_time` field and return the days of the week in the format we are looking for. `%A` specifies the alphabetical day of the week, and `%u` specifies the numerical day of the week; the combination here will give us our days in the proper order.

> For more information on this and other functions, refer to `http://docs.splunk.com/Documentation/Splunk/6.1.3/SearchReference/Commonevalfunctions`.

Then we ask for the events to be sorted first by **DayofWeekN** (numerically), and then followed by **DayOfWeekA** (alphabetically). The following code specifies this:

```
buttercupgames | eval DayOfWeekA=strftime(_time,"%A") | eval
DayOfWeekN=strftime(_time,"%u")| top 1 productId by DayOfWeekN,
DayOfWeekA
```

The result appears as follows:

Obtain top productId for each weekday, sorted in normal weekday order

Summary

In this chapter, we have learned more about how to search using Splunk. We have also introduced how to use the stats command and the eval command, as well as how to find top values, and how to create timecharts, tables, and visualizations.

We will continue to use what we have learned (in combination with some other commands) in *Chapter 4*, *Splunk Reports*, and we will learn how all these commands can be used to create useful reports and dashboards.

4
Reports in Splunk

In the previous chapter, we learned how to use further search techniques, use the stats, eval, and top commands, create visualizations, and also use timecharts. In this chapter, we will go on to learn more about how to further use these skills to create reports and dashboards. The topics covered in this chapter include the following:

- Getting data ready for reporting
- The Report Builder and how to use it
- Using the Report Builder to create a rare values reports
- Creating a dashboard panel with a report
- Creating a pivot
- Adding a pivot to a report

Getting data ready for reporting

Before you prepare a report, you often want to manipulate the data first to get it ready. In other words, if you know the categories you want to end up with, you should group it the way you want before processing. Splunk has two important ways to do this: tagging and setting event types.

Tagging

Tags are used to label specific values of a field. For example, many names of servers may not be immediately recognized, and using a tag format can help them be more easily recognizable or distinguishable from each other.

To tag the value of a field, use the following steps:

1. Go to **Settings | Tags**. A window will open, as shown in the following screenshot:
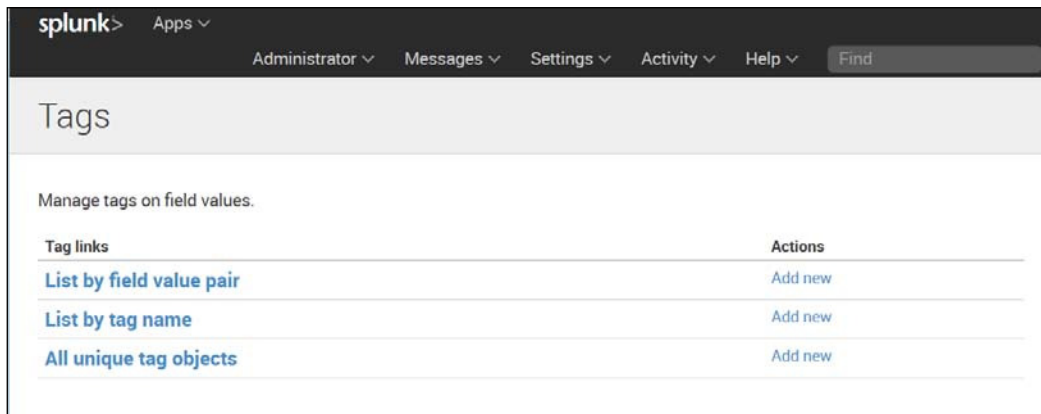
Adding Tags

2. Under **List by tag name**, click **Add new**.

3. Here we want to tag an item as **ITEM14** whenever the value of **itemId=EST-14**, as shown in the following screenshot:

Naming Tags and Specifying Field Value Pairs

4. You will now see your tag listed as shown in the following screenshot:

List by Tag Name

5. Go back to the event list and click the **>** sign next to an event. You will see details of the event open up in a way similar to that presented in the following screenshot. You can see here that **itemid=EST-14** has been tagged as **ITEM14**. Now everywhere that **EST-14** occurs, it will be tagged as **ITEM14**.
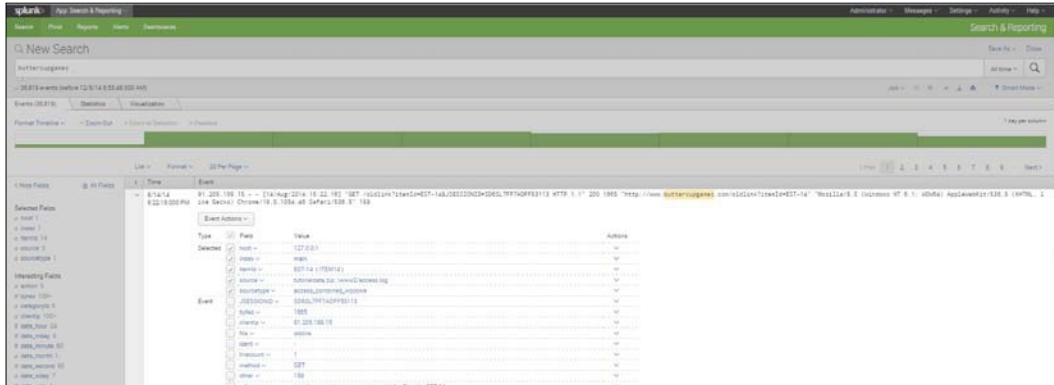
Note that itemid=EST-14 has been tagged as ITEM14

Tags enable you to search more easily and to convey meaning about the field values. When you search **tag=ITEM14**, all the cases where **itemid=EST-14** show up. By using tags in this manner, you can facilitate your analysis.

Setting event types

Another way of preparing data to be reported is to set event types, which let you put events into categories. When setting event types, you can use wildcards, field values, and Boolean expressions. This capability makes event types more versatile and powerful than tags, for which you can only use field values. As with tags, you can choose the categories you like.

When setting event types, be aware of the following:

1. You can't do a sub-search to create an **Event type**.
2. You can't use pipes in a search that create an **Event type**.

As an example of how to create an **Event type**, take the following steps using the `buttercupgames` file:

° Enter this into the search bar:

```
sourcetype="access_*" status=200 action=purchase
```

° This creates a search for events where the sourcetype is an accessed web page, the access was successful (status=200), and it ended in a purchase:

Search that will be saved as an Event Type

3. Click **Save As | Event Type** in the upper-right corner of the screen and create a name for the event type. In this case, we have used the name **success**.

4. In this screenshot, when we enter `buttercupgames | stats count by eventtype`, we get a count of each event type. In this case, we have only one event type, so we get only one count in our table, but we could easily put other event types in:

Shows Count by Eventtype

5. If you want to remove an event type, go to **Settings | Event types**, and you will get a screen similar to what is shown in the following screenshot. Just find the event type you want to remove and click on **Delete**:

Event Types (Notice that you can Delete the one you just made.)

The field extractor

In all of the examples in this book, we will use fields that have been set up automatically or previously set up. One of the primary advantages of Splunk is that it can easily recognize many types of fields. But users can also make use of the field extractor if they want to set up fields in a certain way. This can be accessed by clicking on **>** next to an event, then clicking **Event Actions** as shown in the following screenshot. If you then click **Extract Fields**, you can choose how you would like to pull out fields from the events. This gets complicated quickly though, and, for that reason, is beyond the scope of this book. For a discussion of regular expressions, go to `http://docs.splunk.com/Documentation/Splunk/6.2.1/Knowledge/AboutSplunkregularexpressions`. We'll be going on to learn how to create reports instead:
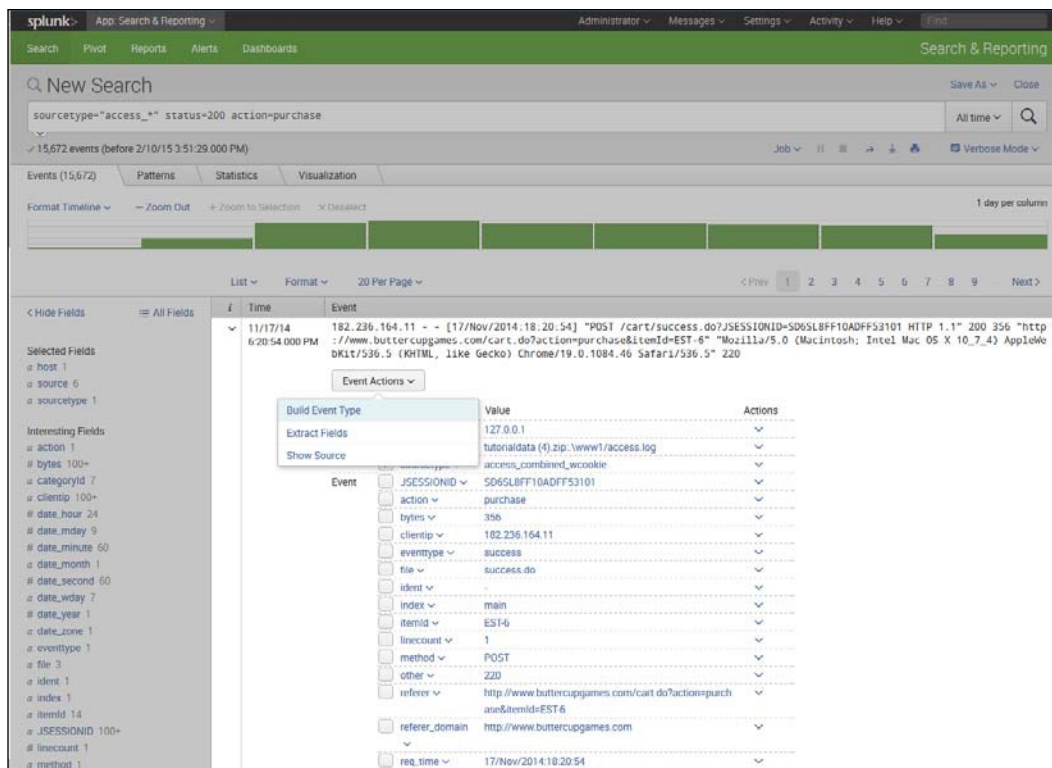
Event Actions

The Report Builder

The report builder can create reports that can be used as needed, or from which you can get regular updates. You can create these reports by running searches or pivots. Below we will show how to create a report using a search you have done on the same sample data that we have been using.

To create a simple report of the counts in each category, take the following steps:

1. In the search box, type the following:

   buttercupgames | stats count by categoryId

2. You will see a chart on the screen.

3. Click **Save As** and select **Report** as shown in the upper right-hand corner of the following screenshot:


Save a Search as Report

4. Give the report a title, such as `CategoryID Counts`.

5. Insert a description if you like.

6. Select a visualization if you wish. If not, just leave it as **None**. (We decided to choose a column chart here.)

7. Choose a time range from the time range picker if you like, by choosing **Yes** or **No**.

8. When you are done, click **Save**.

9. You will see a box that says **Your Report Has Been Created** and will give you additional options:
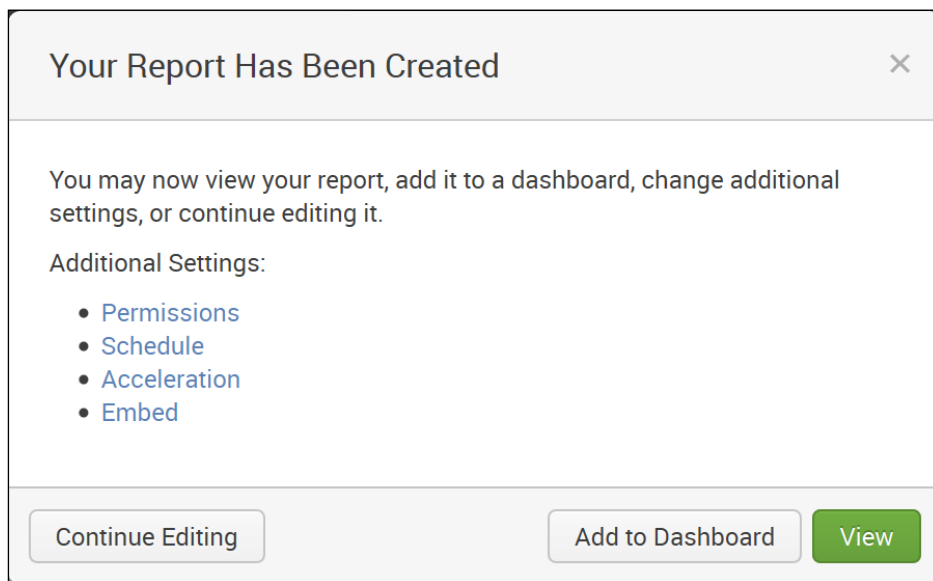
Box Showing Your Report Has Been Created

You now have the following options:

- You can set permissions to view, edit, and delete the report.

- You can schedule the report to be run (every hour, day, week, or month) at a certain time to process data for a specific range of time. You can also schedule an e-mail to alert you when the report runs or can give instructions for a script to be run.

- You can accelerate the development of the report.

- You can embed the report in a web page. (However, the report has to be scheduled to do this.)

Once you have created the report, you can click **Edit** to do one of the following things to the report:

- Change the description
- Edit permissions
- Edit the schedule
- Edit acceleration
- Clone
- Embed the report in a website
- Delete the report

You can also go to the other columns listed after **Actions** and change the following:

- The **Owner**
- The **App** used
- The properties associated with sharing the report
- Whether or not the report is embedded in a website

You will thus be able to generate a report as shown in the following screenshot:

Ways to Edit Your Report

As you can see, there are many options you can take to create useful reports that can be customized, run, and made available by different methods. Reports showcase the flexibility and capabilities that make Splunk useful.

Creating a dashboard

Dashboards are important because they enable decision-makers to have visualizations of several metrics in front of them at a time. They can also be used to drill down in terms of time or other measures.

To create a dashboard, take the following steps:

1. On the home page, under **Search and Reporting**, click **Dashboards** in the upper-left corner of the Splunk home page.

2. Click **Create New Dashboard** in the upper-right of the **Dashboards** page.

3. Fill in the **Title** (**optional**), **ID**, **Description** (**optional**), and any **Permissions** (we use the defaults here). Click on the **Create Dashboard** tab:
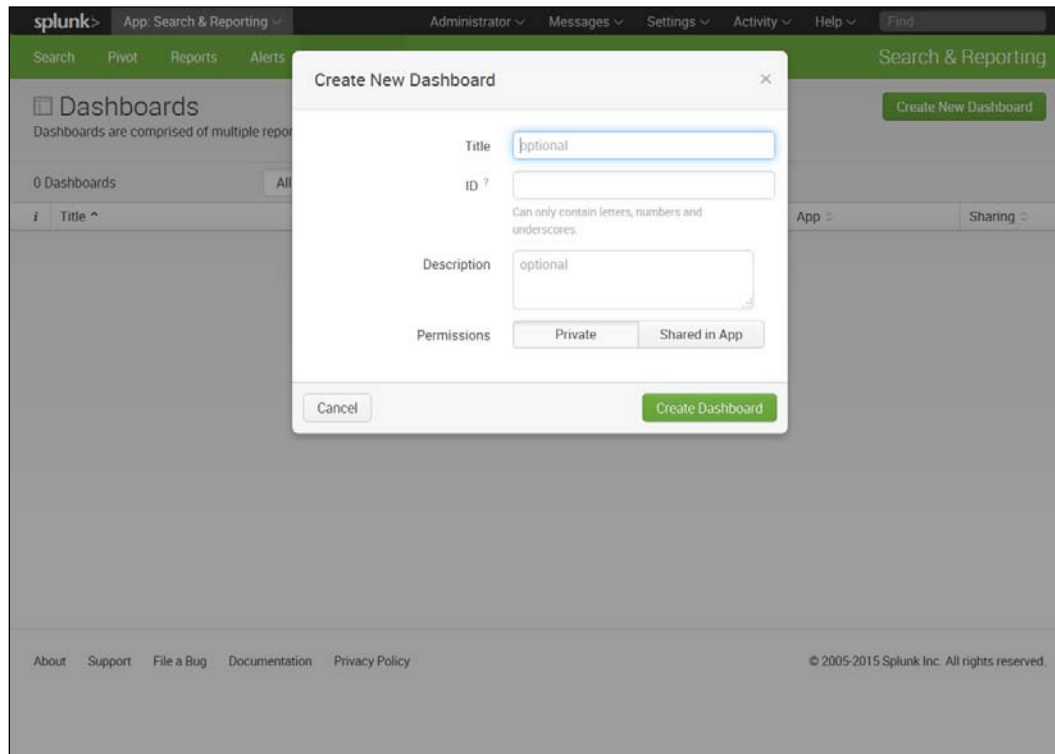
Create New Dashboard

4. We called our dashboard `Main1` in **Title**, which has defaulted to **main1** in the **ID** field.

5. Click on the **Create Dashboard** tab.

6. Click **Add Panel** in the upper right-hand corner as shown in the following screenshot:
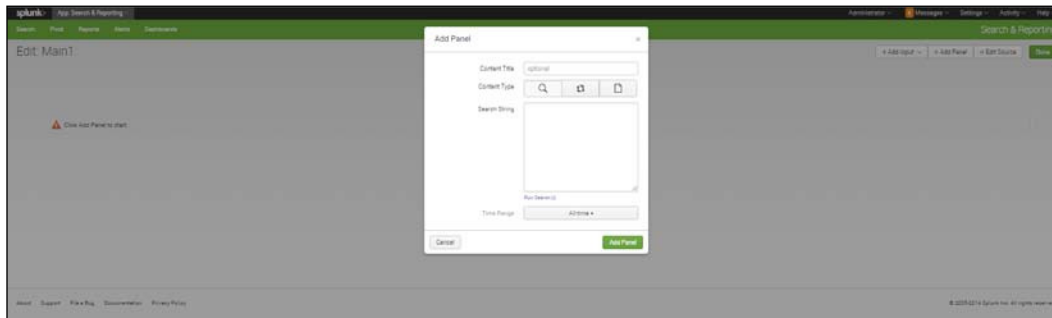
Add Panel

7. You can choose whether your panel will come from **Inline Search**, **Inline Pivot**, or **Report**. In our case, we decided to use the report we just created, that is, **CategoryID Counts**. Under **Content Type**, click on the **Report** icon (the figure that looks like a report in the previous screenshot).

8. Click on **Add Panel**. The statistics panel appears in the dashboard called Main1 as shown in the following screenshot:

Statistics Panel is Added

9. We change it to a **pie** chart by clicking the second icon in the upper right-hand corner and selecting the pie chart icon. Now our screen looks like what is shown in the following screenshot:

Pie Chart Panel

We can edit the title from this screen by going to the first icon in the upper-right corner and select **Edit Title**.

Adding a panel with a search string

It is also easy to add a panel to a dashboard just by adding it and putting in a search string. To create a dashboard and then put in a panel with a pie chart, take the following steps:

1. Under **Search and Reporting**, click **Dashboards**.

2. Enter in the information for a pie chart, as shown in the following screenshot:
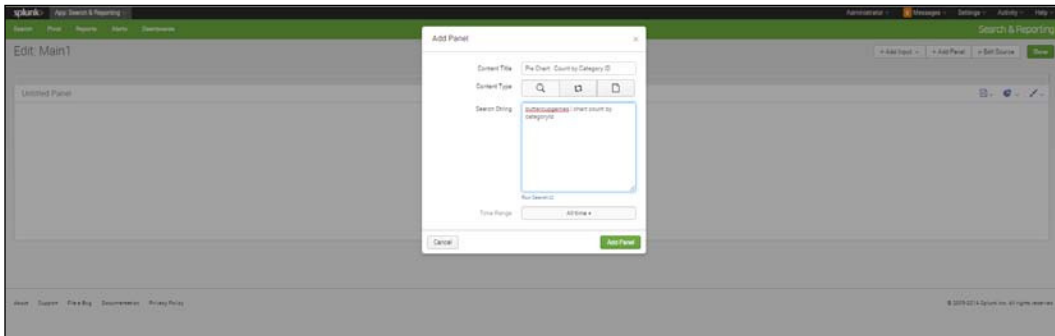
Add a Panel Using a Search String

3. Click **Add Panel**.

4. If the visualization that appears is not a pie chart, click on the **chart** icon in the upper-right corner and select **Pie**.

You should see a chart like the one shown in the following screenshot:

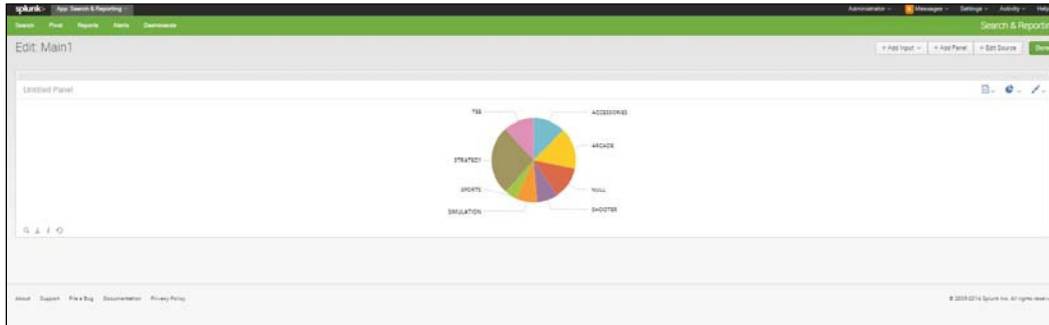Pie Chart Created Using Search String

In the following examples of more charts, you can find different visualizations that can be put in as panels in a dashboard.

Built-in search dashboards

You may not be aware that Splunk has its own built-in visualizations of search activity. If you go to **Activity** menu, then go to **System Activity**, you will see that you can choose to look at **search activity**, **server activity**, or **scheduler activity**. Screenshot a each are shown as follows:

1. First, under **Search**, click **Search activity overview** to see the various panels showing how the search is evaluated:
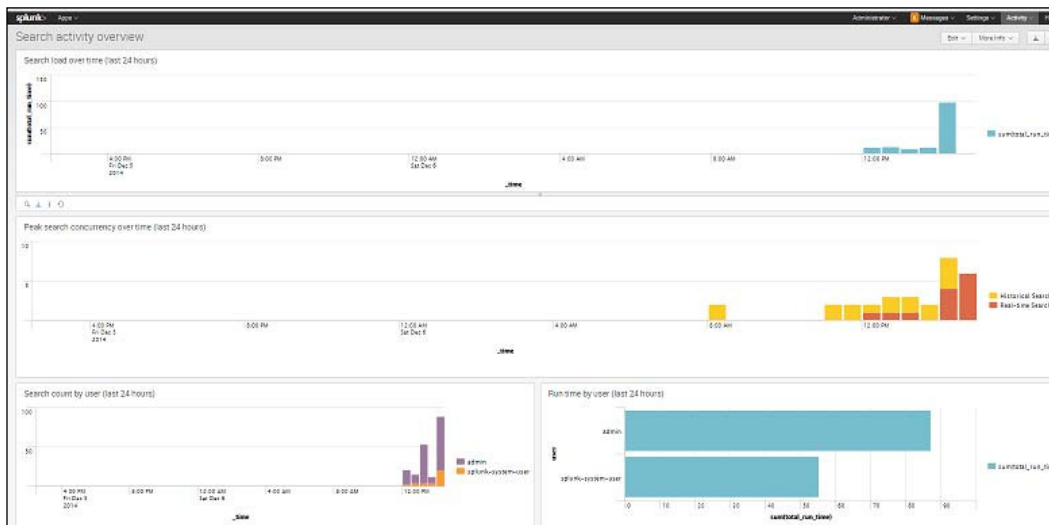
Search Activity Dashboard

2. Under **Server**, click **Splunk Browser Usage and Activity**, and you will get a window like the one shown the following screenshot:
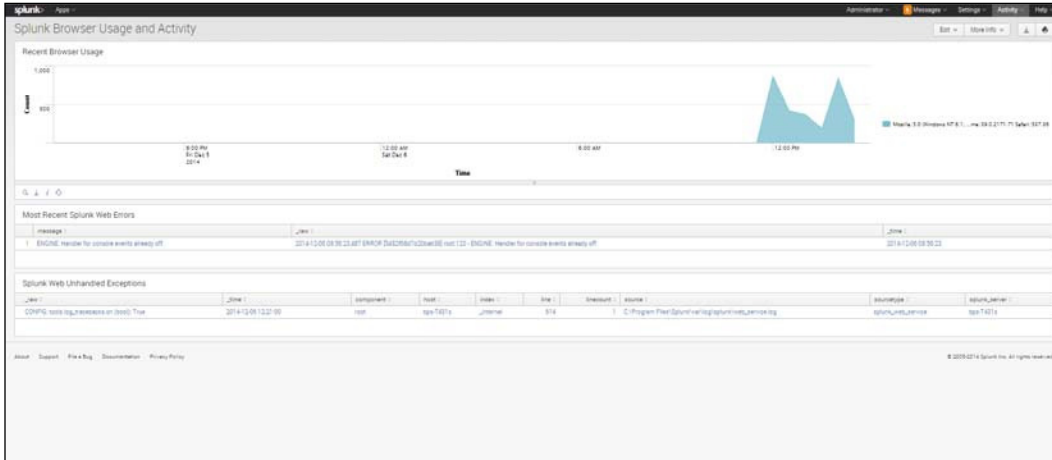
Splunk Browser Usage and Activity Dashboard for Server

3. And under **Scheduler**, click **Scheduler activity overview**. There you will see the following dashboard:
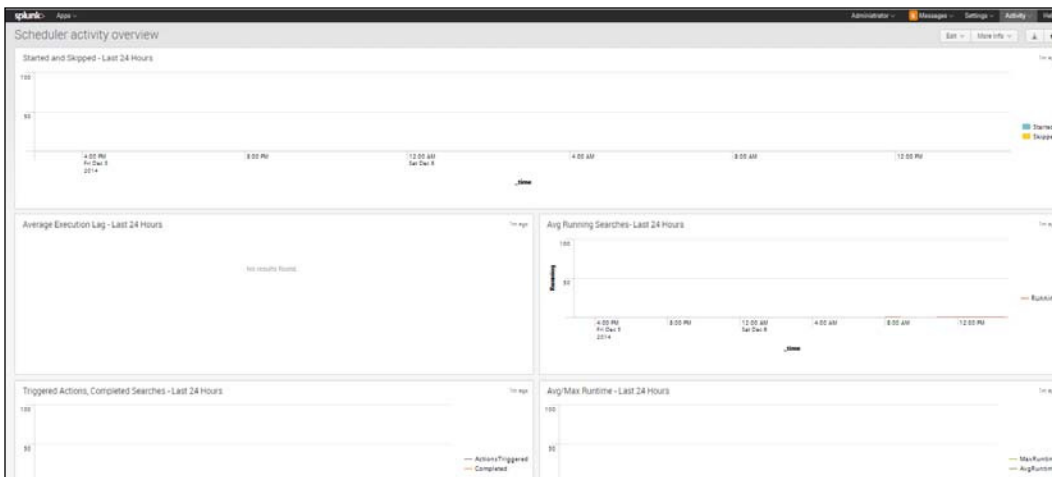
Scheduler Activity Overview Dashboard

All of these dashboards are helpful not only because they measure the internal workings of Splunk, but also because they exhibit different ways to make panels. To view the SPL behind each panel, click on the magnifying glass icon in the lower left-hand corner of each panel.

Creating a bar chart

Another common way to view data like this is to use a bar chart. For example, such a chart can be used to show the viewer the relative proportions of those who use `method=POST`, and those who make purchases.

1. To create a bar chart, you can enter the following code in the search bar:

   ```
   sourcetype=access* | timechart per_minute(eval(method="POST")) AS
   Views per_minute(eval(action="purchase")) AS Purchases
   ```

2. Let's go through this next step carefully. We begin by searching for all events with a sourcetype that begins with access are collected. Then we use the `timechart` command and the `per_minute` function to first give us a figure for the number of events per minute that use `method="POST"`, and then label it as `Views`. In addition, we use the `per_minute` function to find the number of events per minute that have `action="purchase"`, and then label the results as `Purchases`.

3. Go to the **Visualizations** tab and select **Bar**.

   You should see a chart like the one shown in the following screenshot:
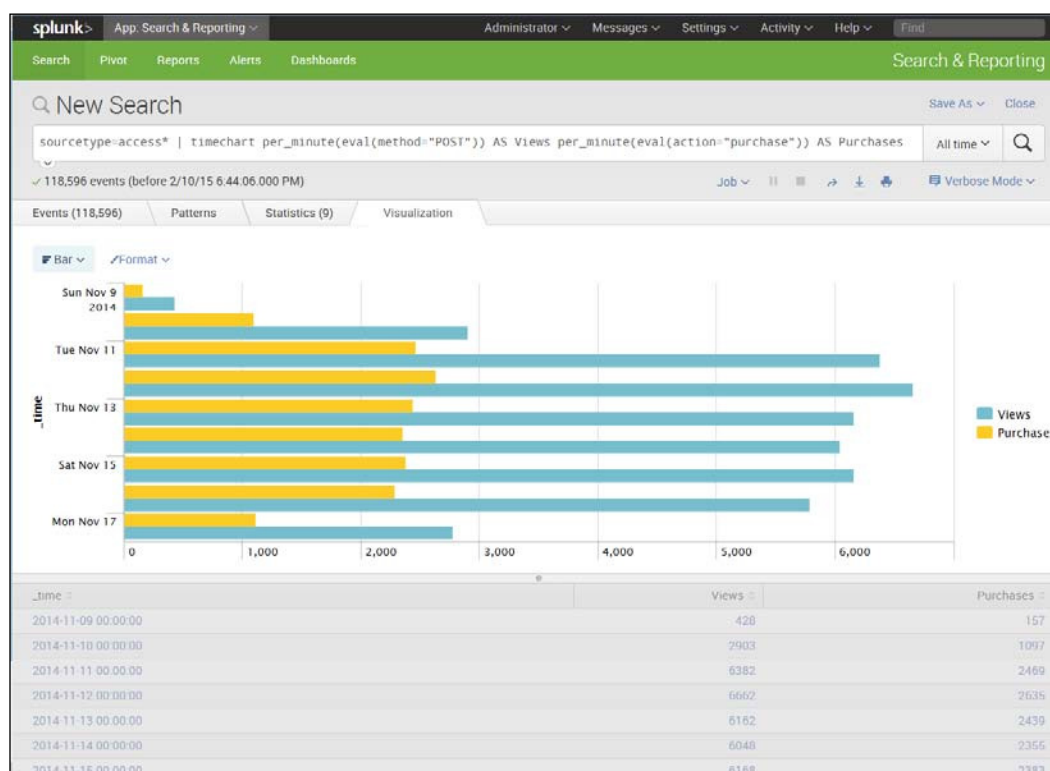
Timechart in Bar Format Showing Purchases

Creating a stacked bar chart

Sometimes, it is useful to see how the different products on a website are selling over time at the same time as you track overall sales. A stacked bar chart can be helpful here. To create a stacked bar chart, take the following steps:

1. Insert the following code into the search bar:

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

2. In the code, nothing should seem that new, except `usenull=f piece`, which indicates that you want to get rid of nulls for this analysis.

3. When you create a chart, it should look like what is shown in the following screenshot:
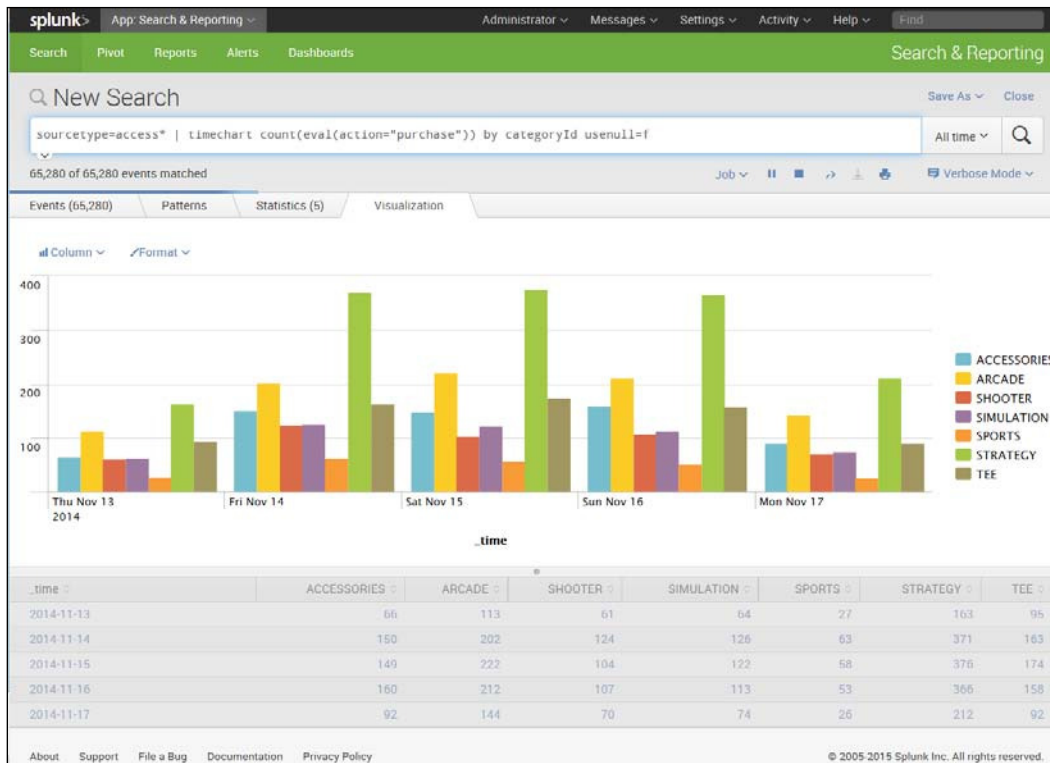
Timechart in Bar Format

4. Click on the **Format** icon in the upper-left corner of the screen.

5. You will see a window like the one shown in the following screenshot. Under **General**, select **Stack Mode**, then select **Stacked**:
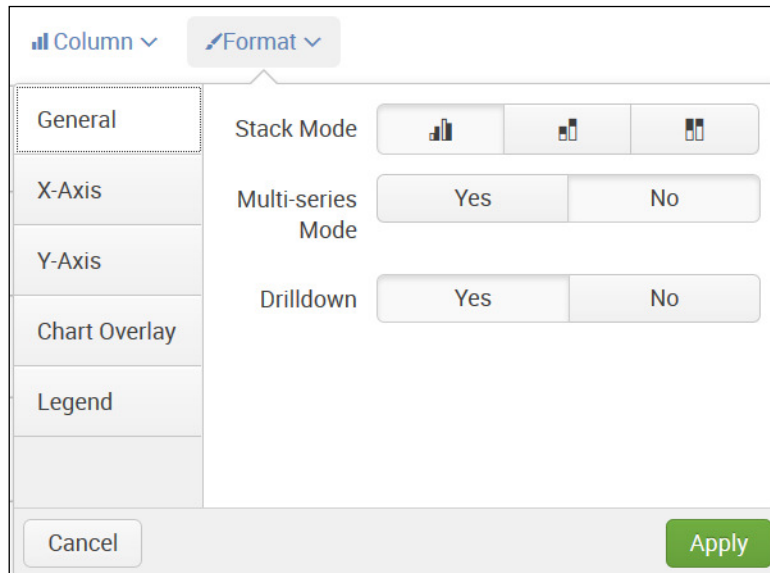
Select Stack Mode as Stacked, Multi-Series as No, Drilldown as Yes

6. Click on the **Apply** button.

Your resulting chart should look like the one shown in the following screenshot:
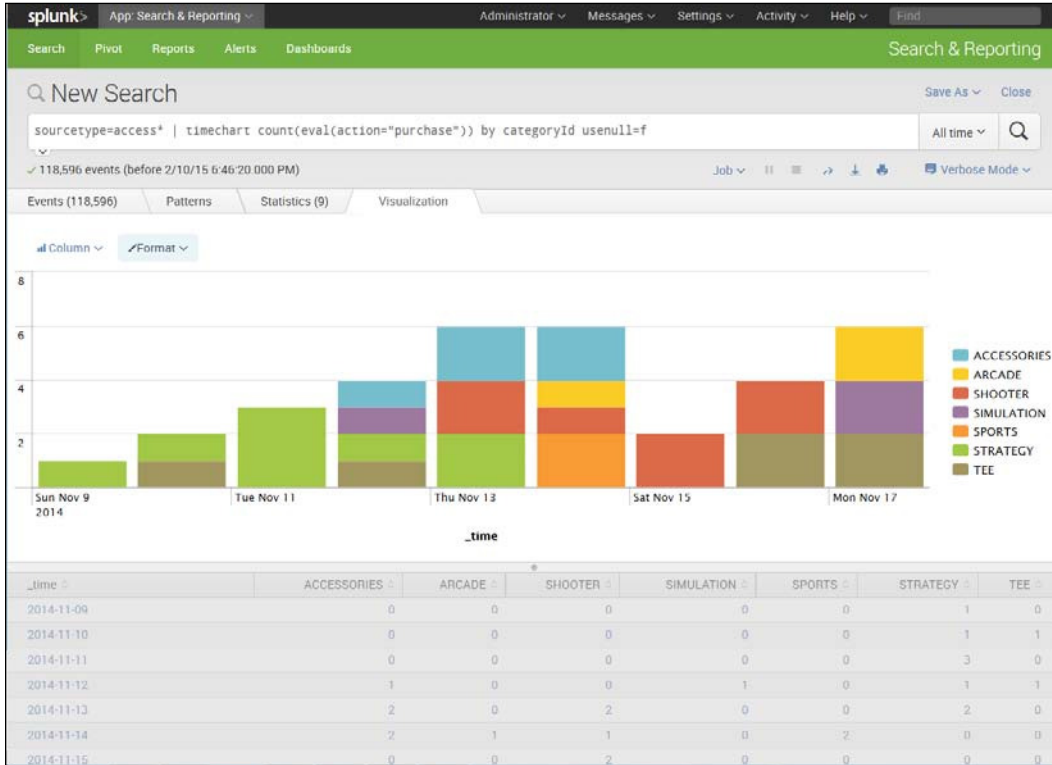
Stacked Chart

Changing the placement of a legend

In the previous stacked bar chart, the legend is on the right. If you want to change this, you can do it via the same drop-down window that we used to change the bar chart into a stacked bar chart:

1. Go to the **Format** icon in the top-left corner of the **Visualizations** tab.
2. Select the drop-down window.

3. Click on **Legend**.

4. Under **Position**, click **Bottom**.

    Your resulting chart will now look like the one shown in the following screenshot:
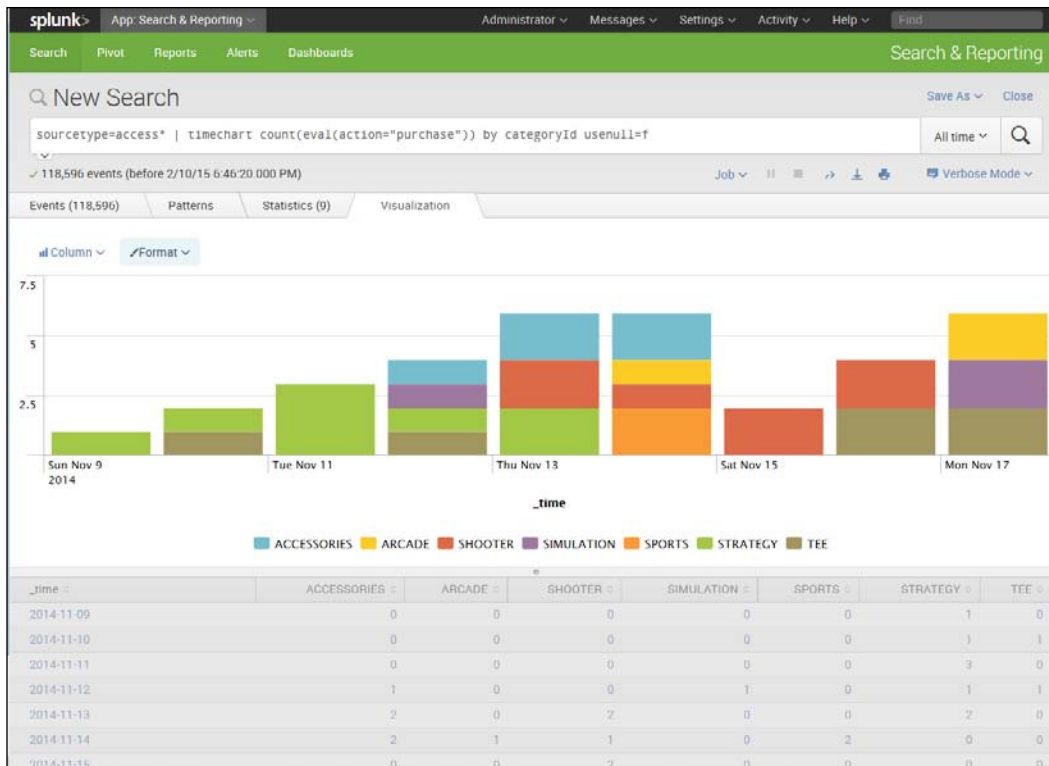
Legend is Shown at Bottom of Chart

Creating an area chart across time

The `timechart` command can be used to put together a useful chart of items bought over time. Let's look at the following code and then put it into the search bar:

```
sourcetype=access* | timechart per_minute(eval(method="POST")) AS
Views per_minute(eval(action="purchase")) AS Purchases
```

If you need to review what the code means here, go back to the bar chart shown in the following screenshot. To create an area chart from this search, take the following steps:

1. Run the search.

2. Make sure that your tabulations look reasonable and that you have **Views** and **Purchases** as column headings, and days on the side.

3. Click the **Visualizations** tab.

4. Click on the top-left icon to select **Area**.

   Your chart should look like what is shown in the following screenshot. Such as chart is useful as it shows the proportion of purchases that use `method ="POST"` and how they change over time:
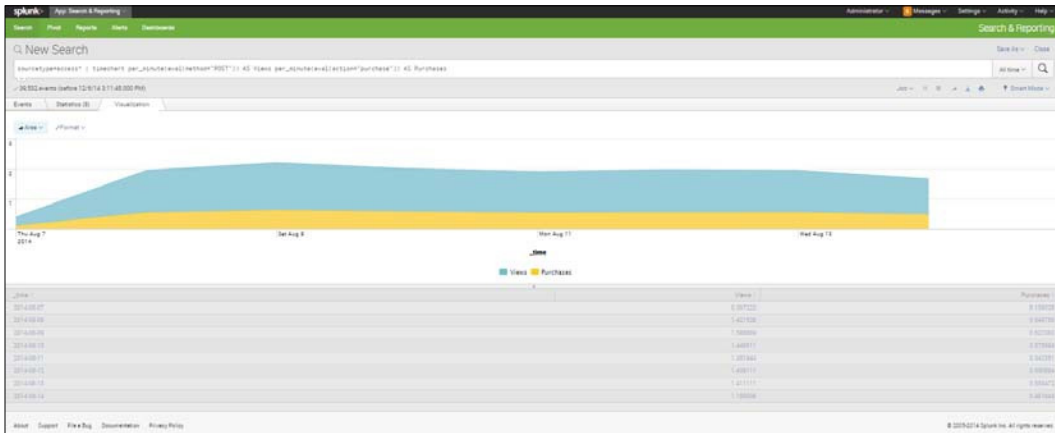

Area Chart of Percentage of Views as Purchases over Time

How to make a sparkline panel

Sometimes, it is interesting to be able to easily compare the ups and downs of various categories of an indicator field in one visualization. Sparklines allow you to do this, as they can easily track trends. They are very small line charts.

To create a sparkline panel, take the following steps:

1. Using the `buttercupgames` data, type the following code into the search bar:

   ```
   sourcetype=access* | chart sparkline(count(eval(action="purcha
   se"))) AS "Purchase Trends" count(eval(action="purchase")) AS
   Total by categoryId
   ```

   > It is very important that you spell `categoryId` exactly as it is written, with one capital I and no other capitals. Otherwise, this code will not run.

2. For each `categoryId` type, you will see a sparkline showing purchases over time that has been renamed **Purchase Trends**, and a count of the subtotal labeled as **Total**, as shown in the following screenshot:
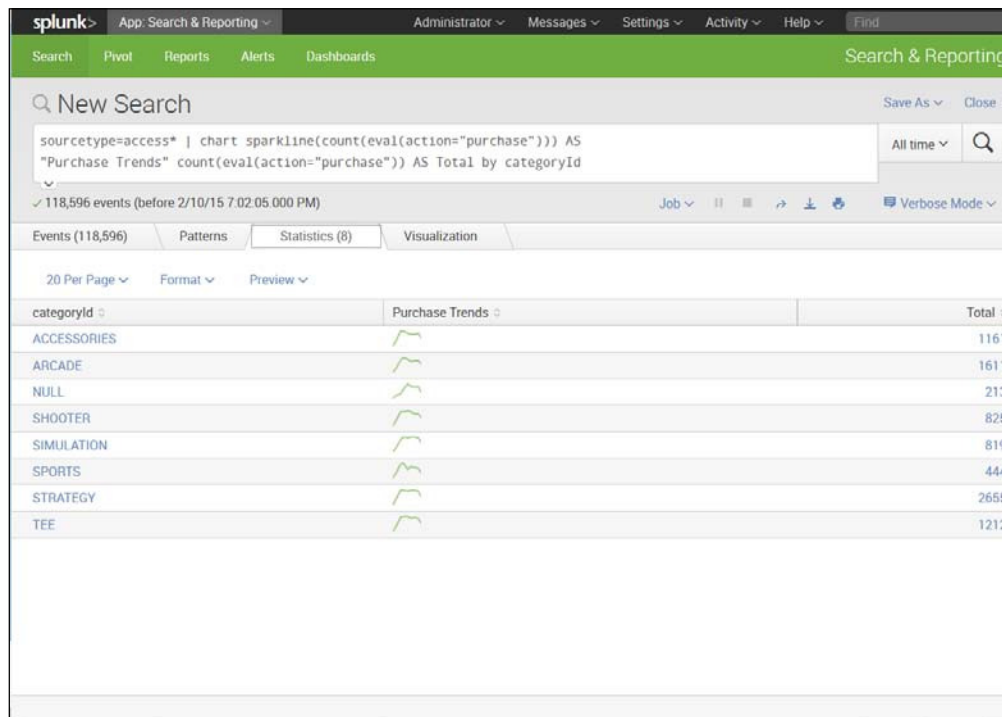

Sparkline Chart

Creating a scattergram

A scattergram is useful for comparing values for two fields. It can sometimes pick up correlations between fields. A positive correlation can be seen when a scattergram goes from the bottom left to the top right; a negative correlation from top left to bottom right. A scattergram can also show the spread of variation. If points are tightly clustered around an imaginary line in a positive direction, we can intuit a strong positive correlation. Likewise, if they are tightly clustered around an imaginary line in a negative direction, we suspect a strong negative correlation in the underlying data. In our example here, we are not looking for a correlation, but just observing a pattern in the data.

To create a scattergram, take the following steps:

1. Put the following code in the search bar:

   ```
   buttercupgames | stats count(eval(action="purchase")) as Purchase
   by date_minute
   ```

2. Look at your results on the **Statistics** tab. It is hard to see a relationship between the counts of purchases and minutes.

3. Click the **Visualizations** tab.

4. Click the icon in the upper-left corner and select the **Scattergram** chart.

5. Your chart will now show each purchase by minute over time.
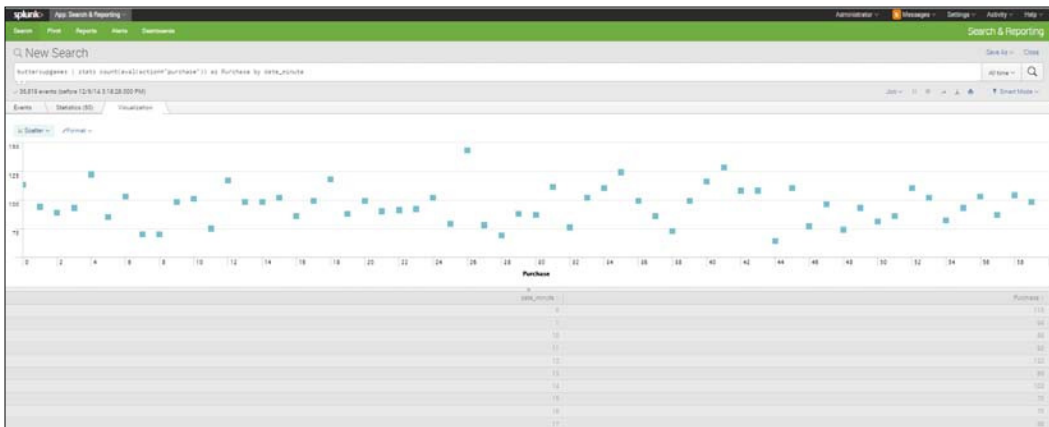
Scattergram Chart

Creating a transaction

You can group events as a transaction. The `transaction` command creates two fields:

- Duration, which is the difference between timestamps for the first and last events
- Eventcount, which is the number of events in the transaction

For example, you can use the transaction command to create a chart to show the number of transactions based on client IP address, a maximum pause of 1 hour, output evicted transactions (`keepevicted=true`), and output original events in the order they arrived (`mvlist=true`). The case function sets the name of transactions where `eventcount=1` to "`Bounced`", `2-5 pages` to "`2-5 pages`", and where it is **<=10** to "6-10 pages". It places a top limit on these transactions of 4,000 and distinguishes these bins of eventcounts as `user_type`. The steps used are shown here:

1. Insert the following code in the search bar:

    ```
    sourcetype=access* | transaction clientip maxpause=1h
    keepevicted=t mvlist=t | eval user_type=case(eventcount=1,
    "Bounced", eventcount<5, "2-5 pages", eventcount<=10, "6-10
    pages") | top limit=4000 user_type
    ```

2. Change the type to **Pie** by clicking on the icon in the upper-left corner of the **Visualizations** tab.

Your chart should look like what is shown in the following screenshot:
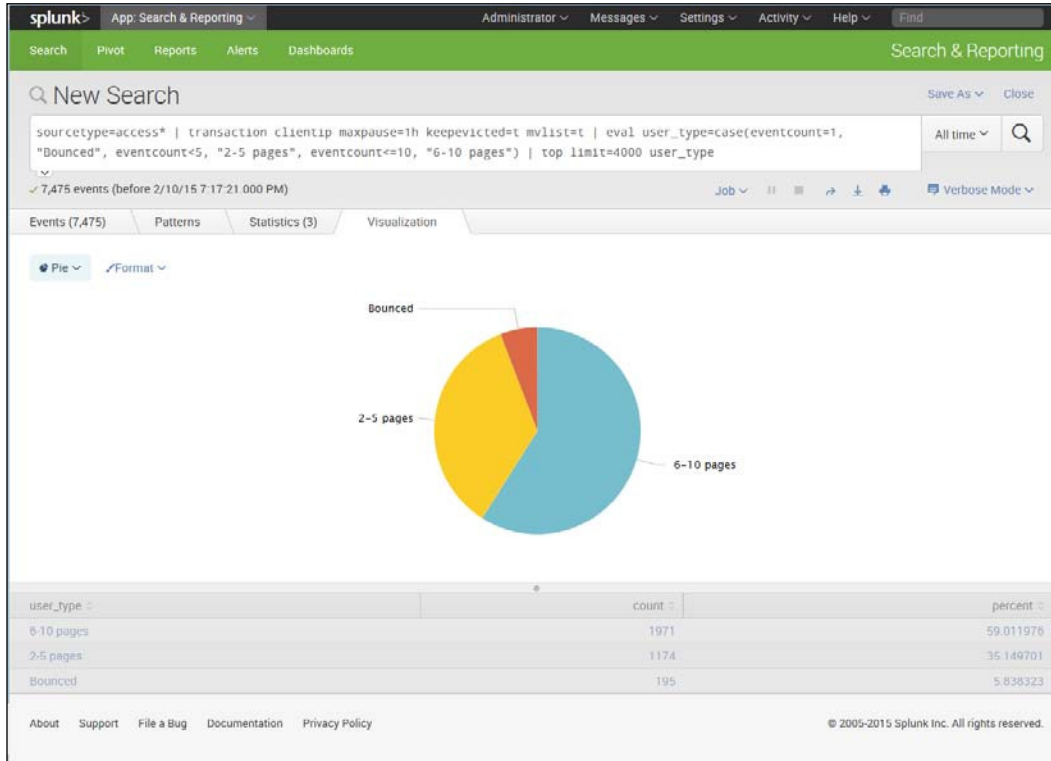
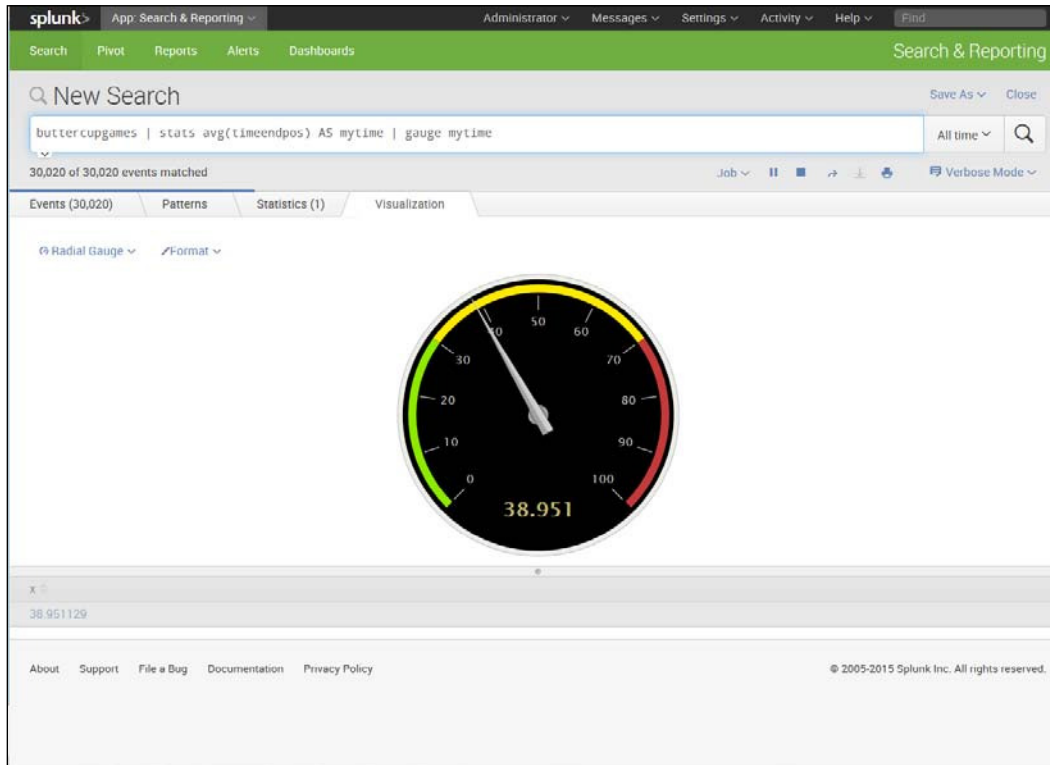Pie Chart Showing Events by user_type

Radial Gauge

A radial gauge is an effective visualization and is easy to create in Splunk. A radial gauge can be created by carrying out these steps:

1. Type in the following code:

```
buttercupgames | stats avg(timeendpos) AS mytime | gauge mytime 0
20 40 100
```

You should get a result like the one shown in the following screenshot:

Radial Gauge Chart

2. Note that you are searching the `buttercupgames` events, and wanting to measure the average end time position or length of event in seconds. Since you are interested in drawing attention whenever the average event time goes over 40 seconds, you create a gauge that marks anything over 40 as red, and one that also has two categories for 0 to 20 and 20+ to 40.

Your chart should look like what is shown in the following screenshot:
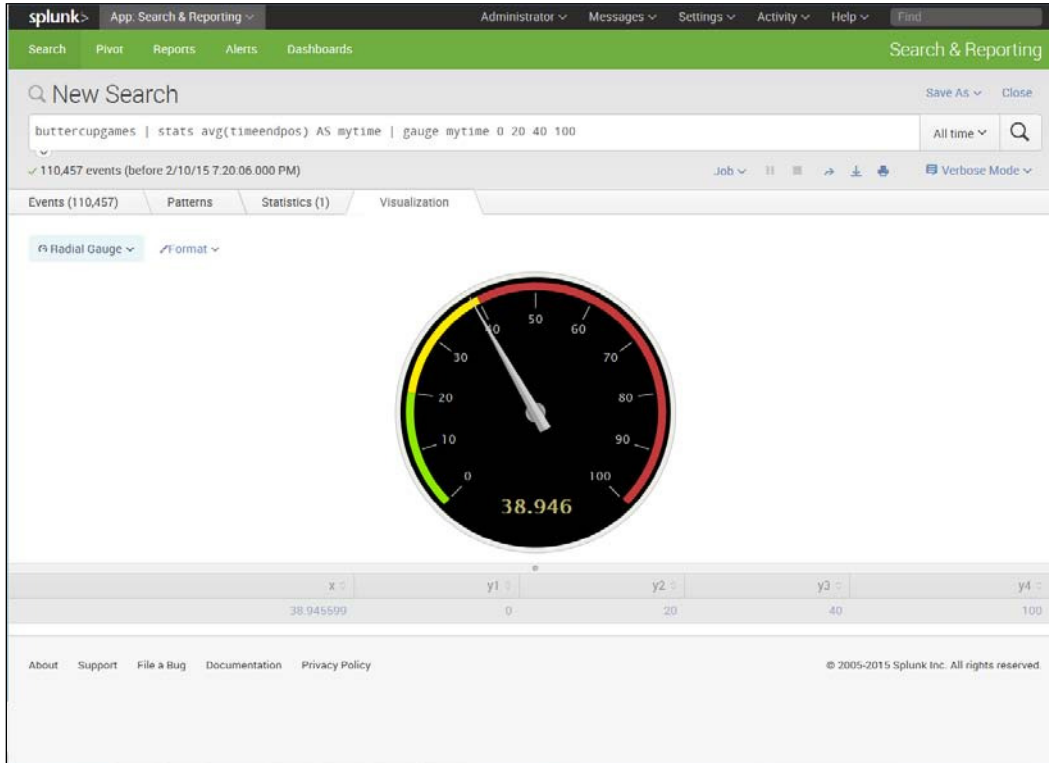
Radial Gauge Chart Showing Changed Category Ranges

Creating a Marker Gauge

You can also use a different type of measure, a Marker Gauge, for the same data as in the previous radial gauge example. To create one, simply take the following steps:
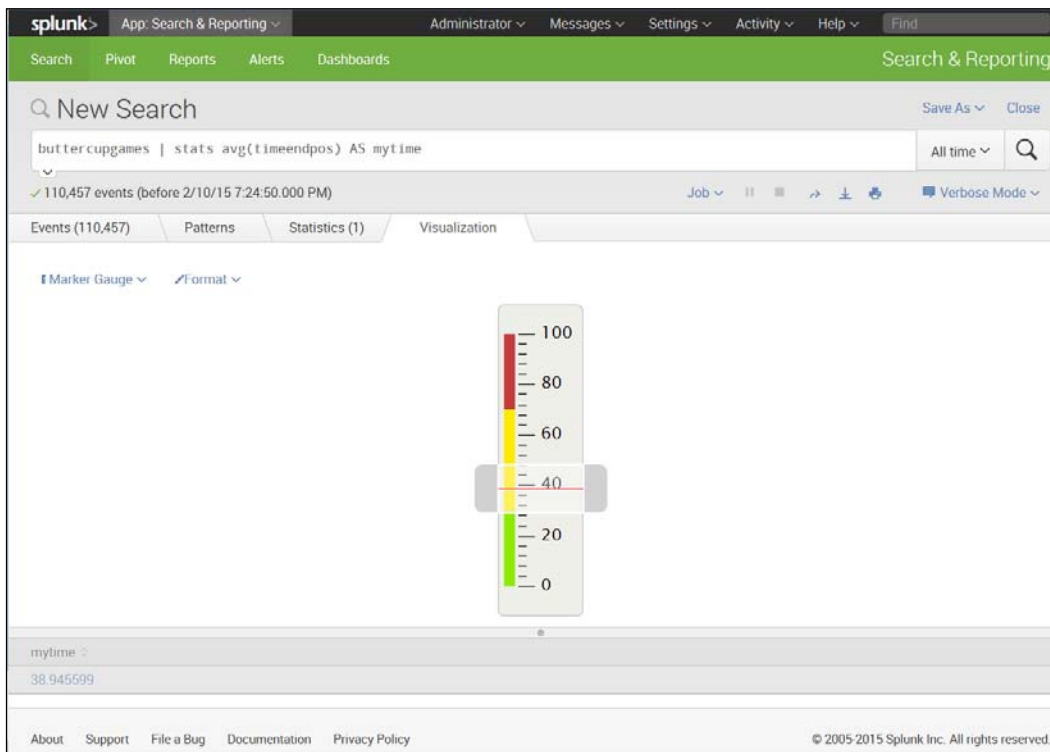
1. Type in the following code:

```
buttercupgames | stats avg(timeendpos) AS mytime
```

2. Click the **Visualizations** tab and select **Format | Marked Gauge**.

3. Click **Color Ranges**, **Manual,** as shown in the following screenshot:
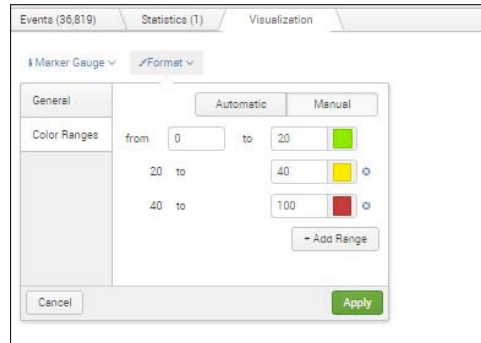
4. Type in **20**, **40**, and **100** for the three colors, as we did in the previous **Radial Gauge**.

   Your chart will look like the one shown in the following screenshot:

Marker Gauge

Creating a pivot table

Pivot tables allow you to view the data in many different ways. Splunk has many shortcuts that users can take advantage of to create pivots easily. We will create a simple pivot table here, using the following steps:

1. We want to open the pivot table interface, so we go to **Home Page**, and then, under **Search and Reporting**, select **Pivot**.

2. To create a pivot table, you use a model. Data models allow you to structure the fields in objects that are easy to pull data from. You should see a short list of models. A model is set up by someone who has detailed knowledge of the data and its properties. Here, we will use a model that is downloaded when you download Splunk. Click on **Splunk's Internal Audit Logs—SAMPLE**. After you select the model, you will see a screen that shows the objects in the model:
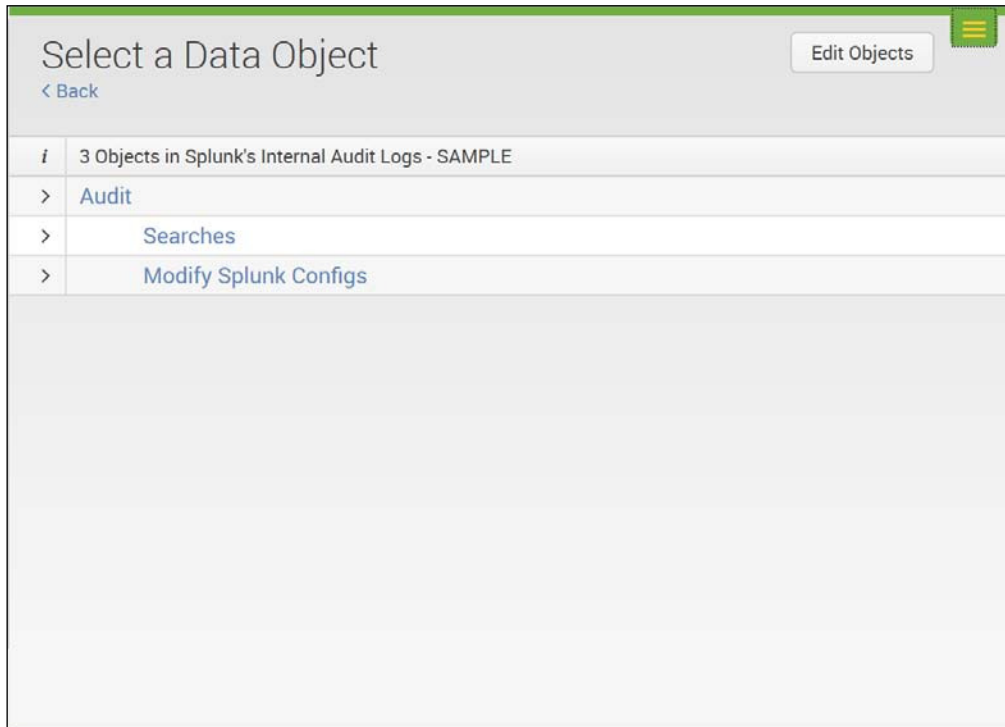

Select a Data Object

3. In the screenshot, you can see that there is one root object (**Audit**) and two child objects (**Searches** and **Modify Splunk Configs**). If you click the **>** icon by each object or child object, you can see the fields that are included in each of them, as shown in the following screenshot:
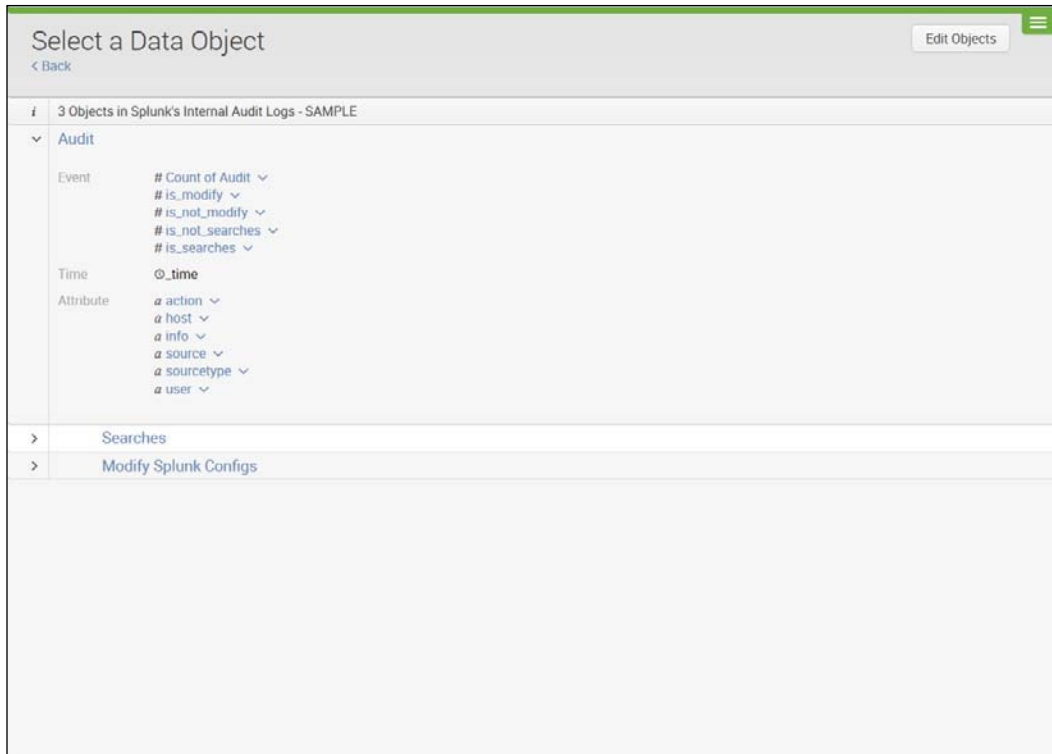
Select a Data Object

4. If you click on the downward **V** icon next to each field, you can see the options that are available to view the field in a pivot—**Top Values** or **Top Values by Time**.

5. Click the downward **V** icon next to **info** and select **Top Values by Time**:
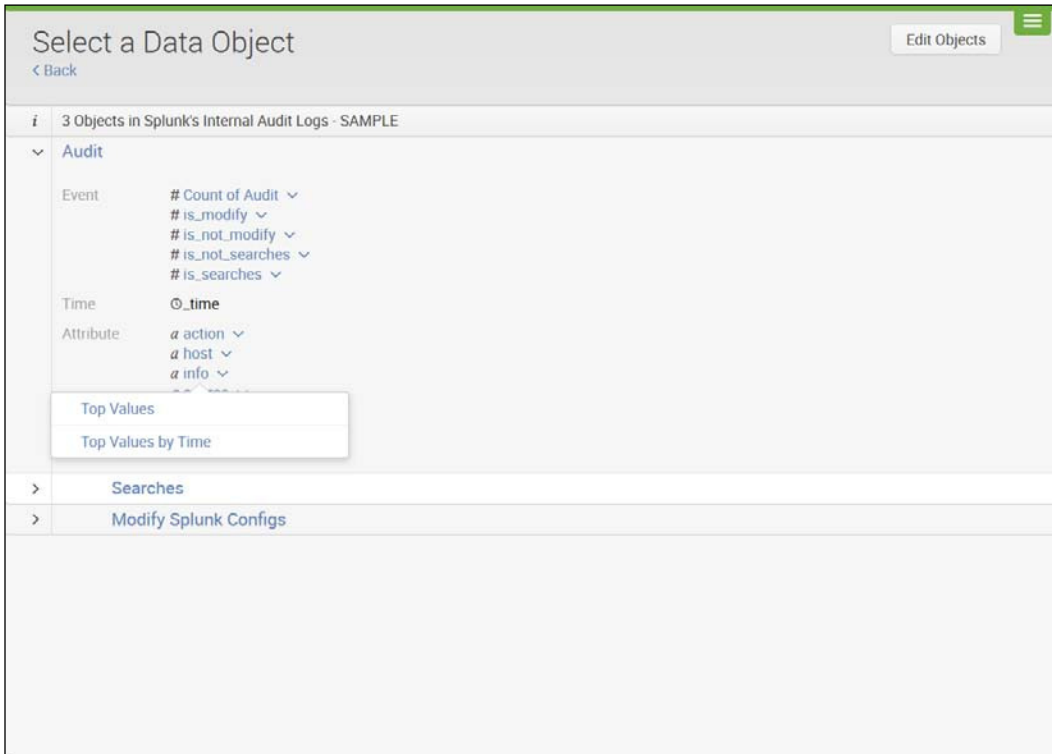
Select Top Values by Time

6. A pivot is created that shows time (in days) by categories of the field info:
   NULL, **canceled**, **completed**, **failed granted**, **n/a**, **succeeded**.

New Pivot Showing Time by Top Values of Field info

7. Save your pivot by clicking **Save As** and select **Dashboard Panel**.

8. Create a new dashboard and name it:

Save as Dashboard Panel

9. Select **New** and put in **Dashboard Title** and **Panel Title**. You now have a dashboard with one panel.

10. Create another panel of your own choosing by using some other fields in a pivot table. Add the panel to your dashboard.

Summary

In this chapter, we have learned more about how to use Splunk to create reports, dashboards, and pivot tables. We have covered various ways that Splunk's data visualization capabilities can be used to create charts and graphs for dashboard panels or reports, including bar charts, stacked bar charts, pie charts, scattergrams, sparklines, area charts, radial gauges, and marker gauges. Additionally, we have learned about transactions and pivot tables, as well as their usefulness. We will now go on to *Chapter 5*, *Splunk Applications*, and explore the many different types of applications that are available to be used with Splunk.

5
Splunk Applications

In the previous chapter, we created reports and dashboards. In this chapter, we will make a slight digression from learning how to search and produce reports in Splunk to learning about Splunk applications. We will cover the following topics:

- What are Splunk applications?
- How to find Splunk applications
- The wide range of Splunk applications
- Splunk's app environment
- How to install an app
- How to manage apps
- Splunk's Twitter application
- Installing Splunk's Twitter app

What are Splunk applications?

Splunk applications or apps are a way to extend the capabilities of Splunk. They are easy to install and use. They enable Splunk to bring in data from many sources easily and efficiently, and to quickly generate reports and dashboards using the data. The latest count from Splunk, as of late 2014, shows that there are over 630 apps available.

Exploring the different types of applications is easy and is outlined in the following sections.

How to find Splunk apps

To look for apps, take the following steps:

1. Go to the Splunk home page.

2. Click on **Apps**.

3. Select **Find More Apps**. In the resulting screen, you will see a list of all the apps. Notice that there are many pages of apps to choose from, as shown in the following screenshot:
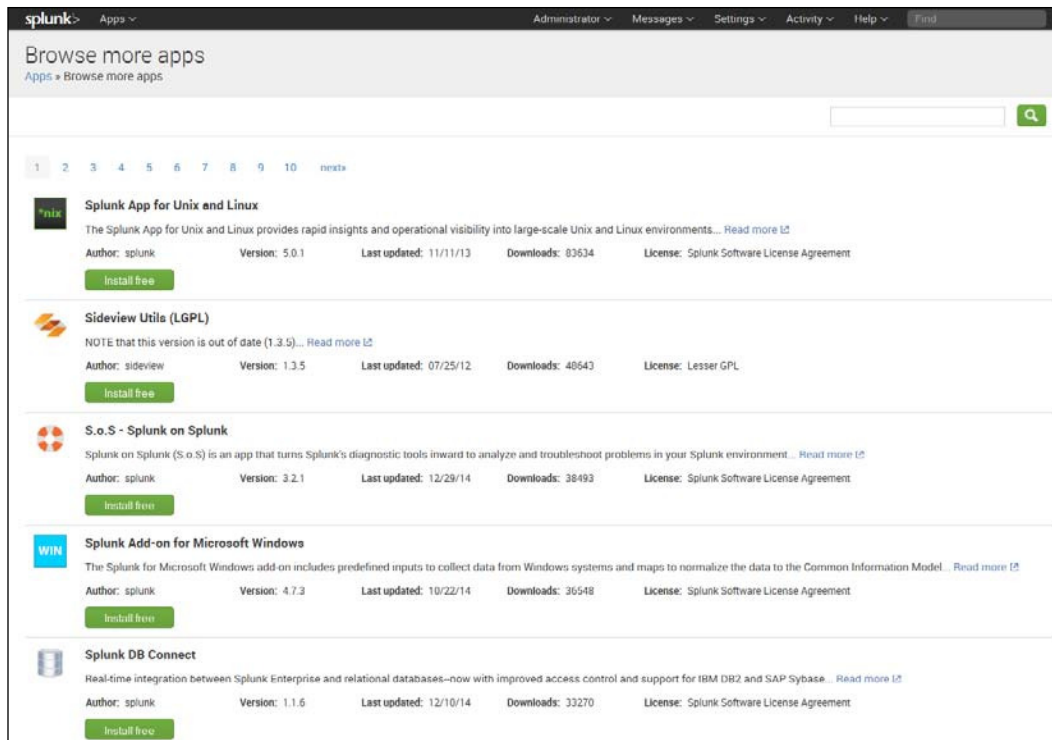
Browse for Apps

The wide range of Splunk applications

For a complete listing of all the current apps for Splunk, you can also go to `https://apps.splunk.com/`. Pay attention to the versions each app will run on, as this is very important to make sure that you will be able to access and use a particular app.

Splunk classifies apps into the categories listed in the following table. Note that some apps are classified in more than one category.

Category	Number of Apps
Application Management	112
IT Operations Management	213
Security and Compliance	210
Business Analytics	37
Utilities	192
Cool Stuff	115

Apps versus add-ons

Splunk differentiates between applications and add-ons:

- A Splunk app includes Splunk features, such as saved searches, reports, and dashboards that are built into a new graphic user interface. Many different apps (383 as of late 2014) have been developed by companies and users.

- Splunk add-ons are also numerous. Their main purpose is to provide a way to format events, including how to break data into events, how to pull out the hostname, and how to rename the sourcetypes, along with how to define field extractions. They can have several distinguishing features:

  - They are generally smaller than an app
  - They don't have their own GUI
  - They may require extra configuration to work with Splunk

There are also a few suites for Splunk that can be either apps or add-ons. These are usually larger, integrated sets of apps that are designed, supported, and installed by Splunk or a company.

The following list shows the other ways you can search apps and add-ons:

- By category (which will be discussed next)
- By support (either the Community or Splunk itself)
- By compatibility with the version of Splunk
- By Common Information Model
- By platform (Linux, Windows, FreeBSD, Solaris, AIX, OSX, HP-UX, and other platforms)

Types of apps

Splunk sorts apps into broad categories. These categories, along with some examples of apps falling into each category, are shown in the following table:

Category	Examples of Apps and Add-ons
Application Management	Splunk App for Microsoft SQL Server
	Ruby on Rails
	Splunk App for Microsoft SharePoint
	Google Voice Analytics
	Splunk App for DMV
	Hunk (for use with Hadoop)
IT Operations Management	Cisco IOS
	Splunk for SAP
	Traffic (analyzes traffic for large cities)
	Teradata Usage Monitor
	Office 365 Data Import
Security and Compliance	Splunk for Symantec
	Barracuda Web Filter
	App for McAfee Web Gateway
	Hurricane Labs App for Vulnerability Management
	Oracle Solaris SMF Manifest

Category	Examples of Apps and Add-ons
Business Analytics	Top Tweets for Twitter
	Sentiment Analysis
	Analytics for iTunes
	Dashboards for IBM Cognos
	Self-Service Analytics and Visualization for Splunk
Utilities	Splunk Web Mobile
	Shuttl (for Big Data)
	R Project
	Splunk 6.x Dashboard Examples
	Weather Alerts (from Weather Underground)
Cool Stuff	AfterGlow Visualization (for network analysis)
	Home Monitor
	Splunk for Stocks Monitoring
	Splunk for Money Exchange

Splunk's app environment

Developing and maintaining different apps in a large enterprise environment can be difficult. In today's world, computer and application architectures can be quite complicated. Different types of data come in from many different places, and these data files and streams need to be monitored and acted upon in many diverse ways – which is why Splunk is so useful. Splunk's app environment is a term that refers to the way that Splunk apps work with the rest of Splunk. Splunk's infrastructure allows developers to easily create apps that build on the usefulness of the Splunk platform as they integrate with it. Splunk's environment makes deploying their enterprise system with appropriate apps easy, which is one reason for its recent dramatic growth.

Creating a Splunk applications

The concept of creating an app is easy, although it may sound intimidating to new users. This is considered a great practice for a company or organization wanting to use Splunk's capabilities. A company's different business units may want to have their own apps that contain their distinctive domain data. A company-specific app can make it easier to integrate the different objects related to a Splunk search head. A Splunk search head is a Splunk Enterprise instance that controls the management of search functions by sending search requests to a set of what are called search peers (or indexers who index and respond to search head requests), and then compiling the results and sending them back to the user. This is useful in that any field extraction, search, report, or dashboard created in the context of an app stays in that app unless it is moved. So if multiple business units or departments are sharing a Splunk search head, this keeps their system tidy without having objects cluttered around in random apps. For this reason, more and more apps are being added to Splunk's copious collection all the time.
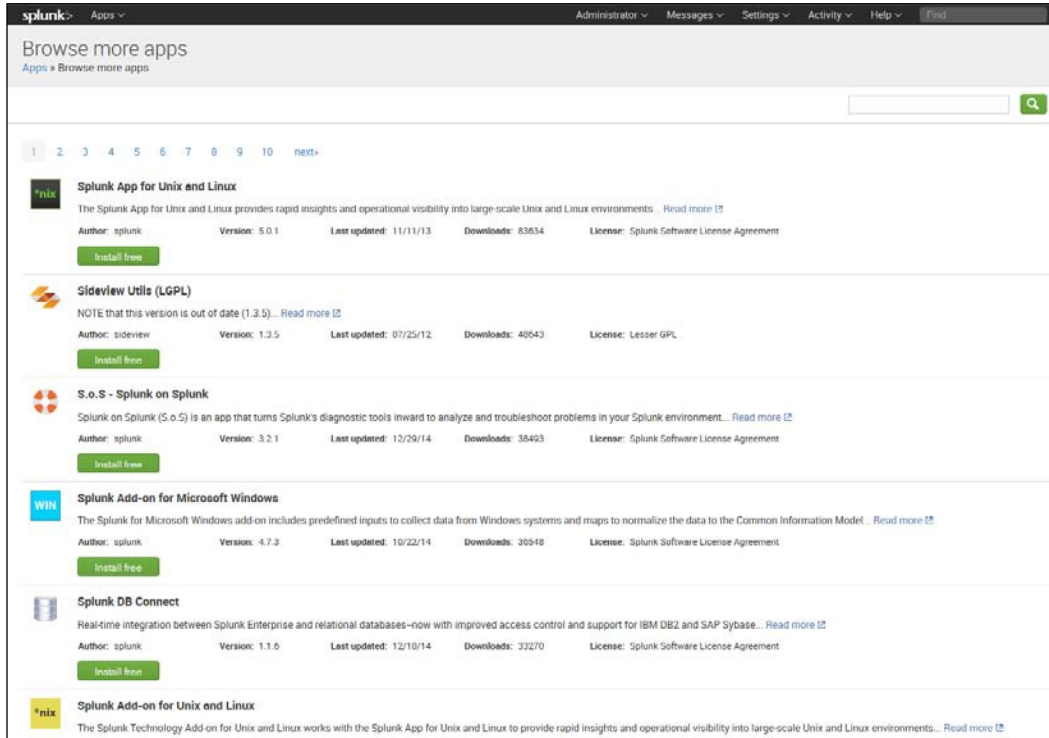
How to install an app

It is easy to install an app in Splunk. To do so, perform the following steps:

1. Go to the Splunk home page.
2. Click on **Apps**.
3. Select **Find More Apps**:
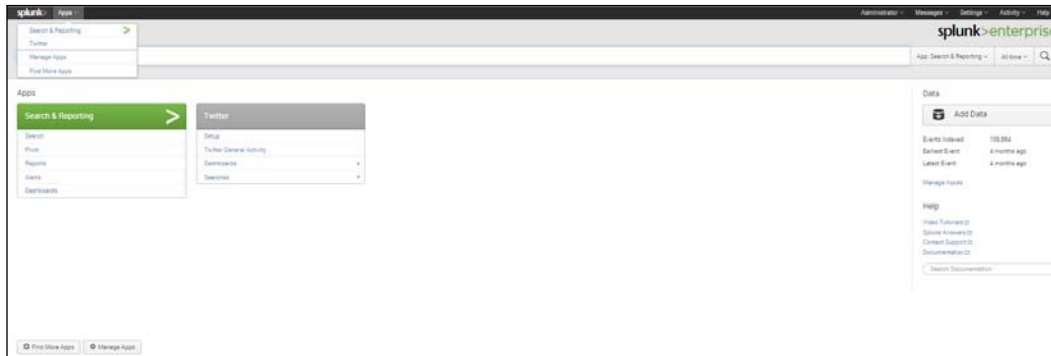
Find More Apps

4. A list of apps and add-ons will open, as shown in the following screenshot:
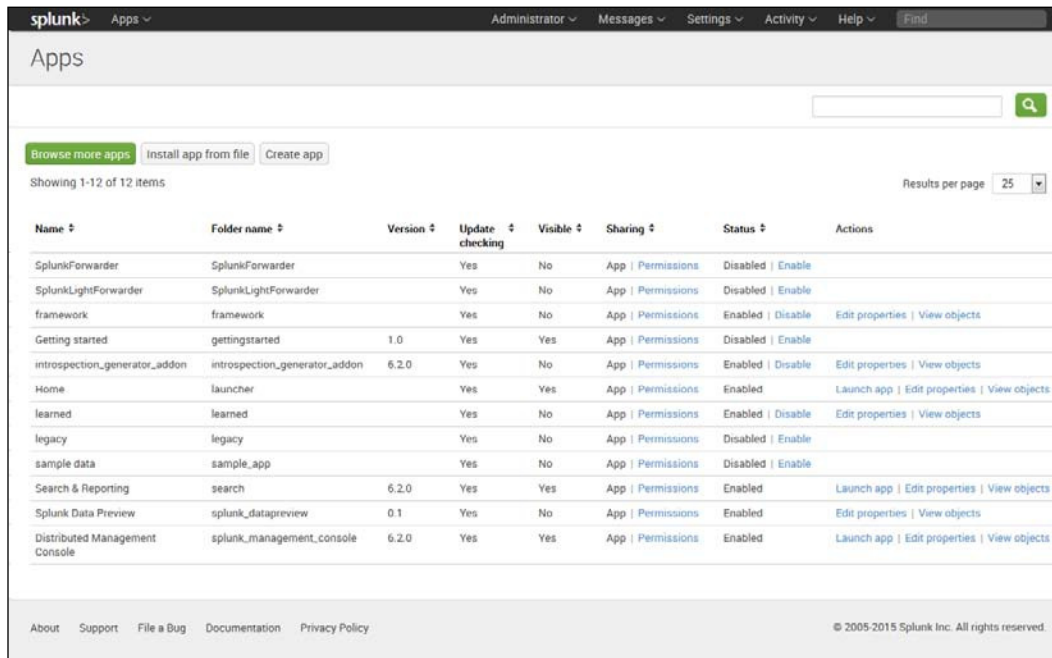

Browse more apps screen

5. We will install the App for Twitter Data at the end of this chapter, but if there is another one you want to install, you can select it and click **Install free**.

6. Follow the instructions to install the app.

7. Restart Splunk, and you should see the app installed next to your other apps on the Splunk home page.

How to manage apps

Sometimes you will need to manage your apps. To go to the page where you can do this, take the following steps:

1.   Go the Splunk home page.

2.   Click **Apps**.

3.   Select **Manage Apps**.

     A screen like the following will open up:

Apps screen, where you can manage apps

4. From this page, note that you can find more apps online, install apps, and even create apps.

5. Also notice the list of apps that you may not have realized were already installed, such as the **SplunkForwarder** and **SplunkLightForwarder** (both of which provide ways of collecting data from remote data sources).

6. Finally, notice that you are able to change permissions for the app. The following screenshot shows the **Permissions** screen for the **Search and Reporting app**:
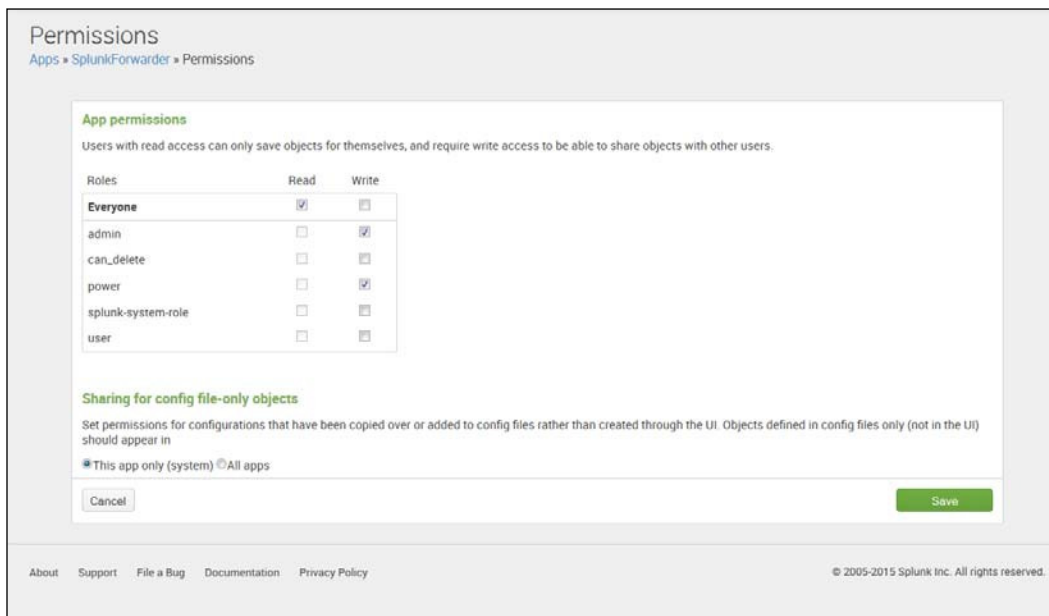
Permissions screen

7. You will see that everyone can read files associated with the app, but only those with the role of **admin** or **power** can write anything for the app.

8. Lastly, notice that you can **Enable** or **Disable** each app, and that you can also **Edit properties** and **View objects** associated with the app. The following screenshot shows the **Edit properties** screen for the **Search and Reporting app**:

The Edit properties screen for the Search and Reporting app

Splunk's Twitter Application

There is an application for Splunk called App for Twitter Data that allows easy access to the 1 percent Twitter sample stream. This stream takes just 1 percent of the tweets available from the firehose of tweets, and lets the user bring in live tweets to Splunk. We will use version 3.0 here. More information about this app can be found at `https://github.com/splunk/splunk-app-twitter`.

Installing Splunk's Twitter app

In the next chapter, we will be working with Splunk's Twitter app to bring in live streams of tweets for analysis. But let's first get it set up for now.

You must start by obtaining a Twitter account, if you do not already have one.

Obtaining a Twitter account

We need to follow these steps to obtain a Twitter account:

1. Before installing this app, you must have an active Twitter account. To obtain an account, go to `https://twitter.com/signup`.
2. Enter your **full name**, **email**, and **password**.
3. Click where it says **Sign up for Twitter**.
4. Select a **username** and **password**.
5. Click **Create My Account**.
6. You should get an e-mail where you can click on the link within to begin using your account.

Obtaining a Twitter API Key

Now you will need to create a Twitter key for the Application Programming Interface or API. This key will allow you to connect to Twitter. Follow these steps to do this:

1. Go to the **Twitter Create an application** page: `https://apps.twitter.com` and select **Create New App**. As shown in the following screenshot, insert a name for your application (it can be almost anything), a description (it can be almost anything), and a placeholder URL (it doesn't have to be real, but it must start with `http://`) for the website. These can be of your own choosing. Since you won't need a website, just put something in for now; for example, `http://www.holdthisbps.com`:

Create an application in Twitter

2. Check **Yes, I agree** in the box below the terms and conditions.

3. Click **Create your Twitter application**. You should see a screen like the one shown as follows:

Settings Information for Twitter API

4. Navigate to the **Keys and Access Tokens** tab at the top of the screen:
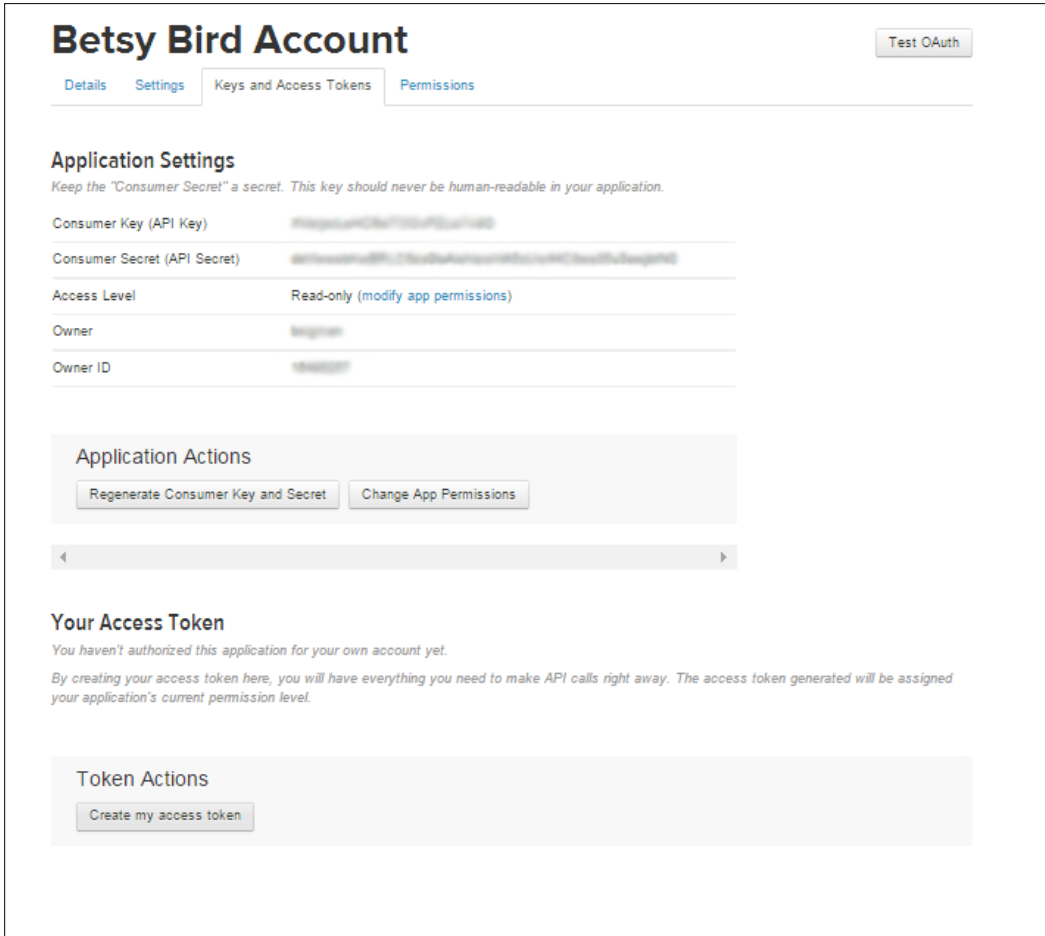
Application Settings for API

5. Click on **Create my access token** below the **Token Actions** area at the bottom of the page. You should see a page like similar to the following screenshot:

Access Token for Twitter API

6. A new section called **Your Access Token** should now appear. If it doesn't, wait another minute and then reload the page.

7. Now you have the API key information you need to install the Twitter app. Keep this page open, as you will need it to access the **API Key**, **API Secret**, **Access Token**, and **Access Token Secret** when you follow the instructions in the next section.

Installing the Twitter app

To install the Twitter app, do the following:

1. Go to the Splunk home page.

2. Go to **Apps**.

3. Select **Find More Apps**.

4. In the search bar in the upper right corner, search for Twitter.

5. Select **App for Twitter Data** as shown in the following screenshot:
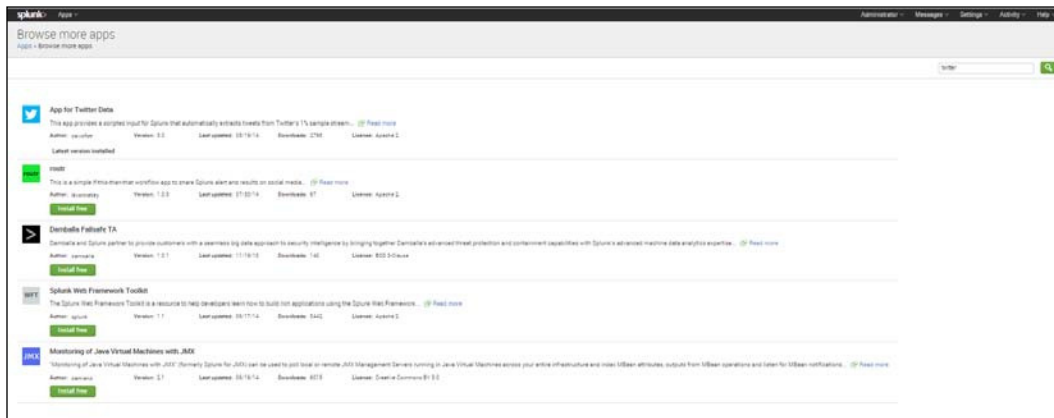
App for Twitter Data Listed with Other Apps

6. Click **Install free**. (In the previous picture, the app has already been installed, so it appears as **Latest version installed**.)

7. You will be asked to log in again with your Splunk website username (not admin, but your Splunk browser username) and password (not the one you replaced changeme with when you logged in using admin, unless you have used the same password to log in to Splunk website).

8. You will need to **Restart Splunk** to install the app. This will take a few minutes.

9. Log back in to Splunk with your admin credentials.

10. You will see **Install successful**; click **Set up now**.

11. Carefully enter your **API Key**, **API Secret**, **Access Token**, and **Access Token Secret** from the Twitter API Keys page from the previous set of instructions; check the **Enable Twitter Input** box and then click **Save**:

Fill in the needed information for the Twitter App for Splunk

> You will need to click on the **Restart Splunk** button to start seeing the data collected from Twitter. Anytime you want to turn off the Twitter input, you must uncheck the **Enable Twitter Input** box. Remember that you can only index 500,000 MB of data a day under the free license. You will need to be careful not to exceed this to avoid having your license revoked.

Now you are ready for the next chapter where we will analyze the live Twitter stream.

Summary

In this chapter, you learned what a Splunk app and add-on are, and you learned about their usefulness. We outlined the different types of applications, noted the numbers of various apps in different categories, and listed several examples of each. You learned how to find an app using Splunk's list of apps, and we discussed the ease and usefulness of developing a Splunk app for a company so that Splunk's functionalities can be used to smoothly work with the company's data. Finally, after introducing you to the Twitter app and learning about how to obtain a Twitter API key to use with it, we went through the process of installing it.

Next, we'll go on to *Chapter 6*, *Using the Twitter App*, and learn how you can use Splunk with this app to create reports and dashboards from streaming tweets.

6
Using the Twitter App

In the last chapter, we learned about the many apps available on Splunk. We also learned about how to obtain a Twitter API key, and how to install the app for Twitter data that is available for Splunk. In this chapter, we will use that app to create reports and dashboards based on streaming Twitter data. We will cover the following topics:

- Creating a Twitter index
- Searching Twitter data
- The built-in General Activity dashboard
- The built-in per-user Activity dashboard
- Creating dashboard panels with Twitter data

Creating a Twitter index

We'll start off this chapter by bringing in some Twitter data using the app we set up in *Chapter 5*, *Splunk Applications*. Open up Splunk and follow the steps given here:

1. Sign in and go to the Splunk home page.

2. If you've set up the app for Twitter data according to the instructions in the last chapter, your screen should look like the following image (if not, go back to the end of *Chapter 5*, *Splunk Applications*):
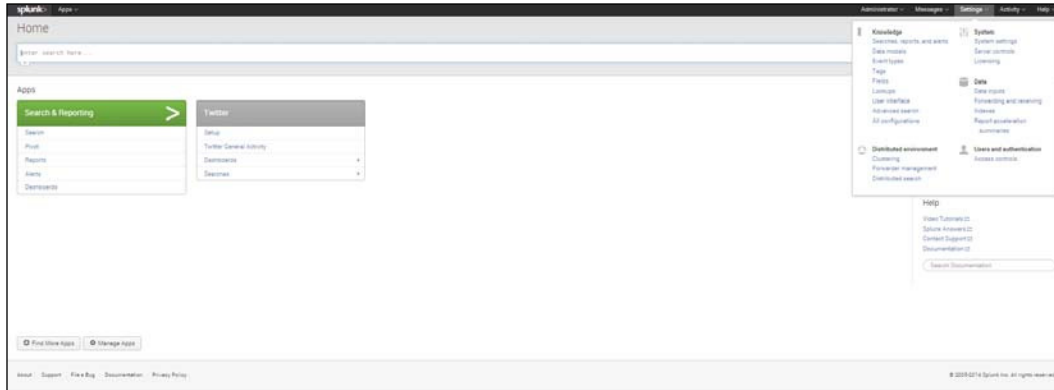
The Home Screen with the Twitter App Installed

3. Click on **Setup**, which is listed first under **Twitter** on the app.

4. You should see the API information you filled in as described in the previous chapter.

5. Check the box **Enable Twitter Input**, as shown in the following screenshot:
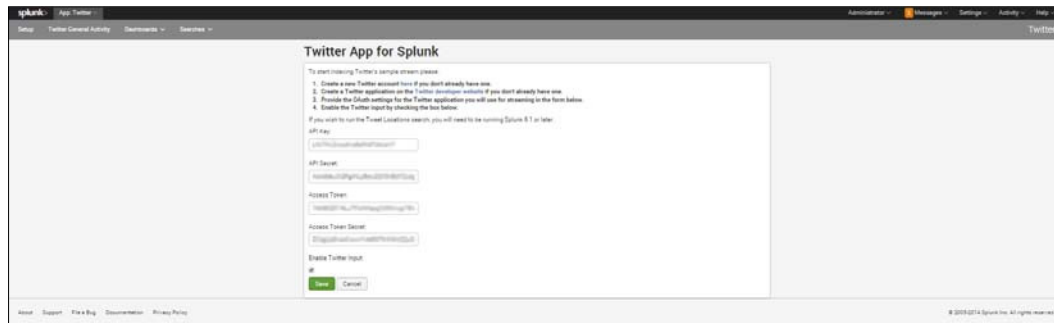
Check the Enable Twitter Input box to start the live Twitter stream

6. This will start the live Twitter stream. Remember that you will need to keep an eye on the amount of data you let in each day, as the Splunk trial license will only allow the indexing of 500 MB of data per day. Going beyond this could mean that you will lose the ability to search your data until the license has been reset or an Enterprise license is purchased. One way to keep track of the data that you are indexing is to go to the list of indexes:

   1. Go to **Settings**.

2. Under **Data**, select **Indexes**.

3. You will see a screen like the one shown here:

The Indexes screen

4. Notice that in the Twitter index (shown in the preceding screenshot), the event count is 423,465. In this case, each event is a tweet. Also notice that you can disable an index easily by clicking on **Disable** under **Status**. Additionally, you can see the path where the Twitter index is stored.

5. Remember that you can only index an additional 500 MB per day in the free version of Splunk Enterprise, but that you can index up to 500,000 MB in the Total Index. The event count is not limited here, just the amount of megabytes of data indexed each day.

6. One way to see how much you have used of the 500 MB allowed per day is to go to **Activity**, then **System Activity**, and under **Serve**, select **License Usage**. You will see a dashboard like the one shown in the following screenshot, which can tell you how much of the day's licensed indexing you have used up. This dashboard shows that I have used 241 MB of the .5 GB (500 MB) allowed per day:
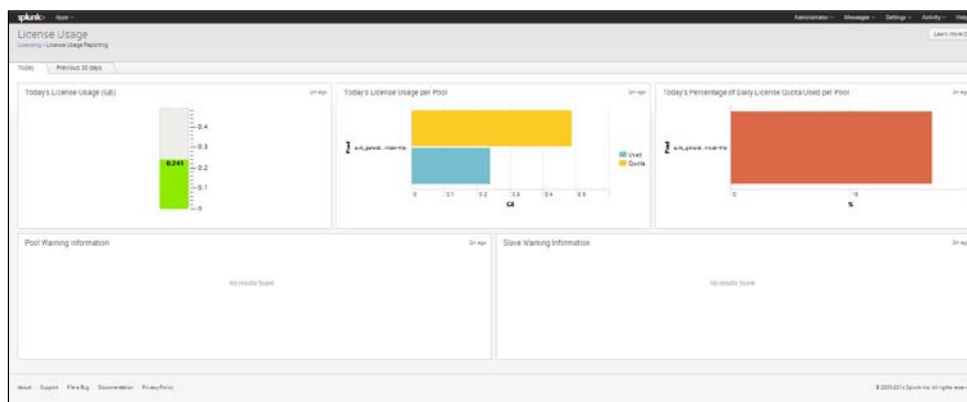
License Usage screen

Searching Twitter data

We will start here by doing a simple search of our Twitter index, which is automatically created by the app once you have enabled Twitter input (as explained previously). In our earlier searches, we used the default index (which the tutorial data was downloaded to), so we didn't have to specify the index we wanted to use. Here, we will use just the Twitter index, so we need to specify that in the search.

A simple search

Imagine that we wanted to search for tweets containing the word `coffee`. We could use the code presented here and place it in the search bar:

```
index=twitter text=*coffee*
```

The preceding code searches only your Twitter index and finds all the places where the word `coffee` is mentioned. You have to put asterisks there, otherwise you will only get the tweets with just "coffee". (Note that the text field is not case sensitive, so tweets with either "coffee" or "Coffee" will be included in the search results. There are hacks to get around this using regular expressions, but these are beyond the scope of this book.)

The asterisks are included before and after the text "coffee" because otherwise we would only get events where just "coffee" was tweeted – a rather rare occurrence, we expect. In fact, when we search our indexed Twitter data without the asterisks around coffee, we got no results.

Examining the Twitter event

Before going further, it is useful to stop and closely examine the events that are collected as part of the search. The sample tweet shown in the following screenshot shows the large number of fields that are part of each tweet. The **>** was clicked to expand the event:
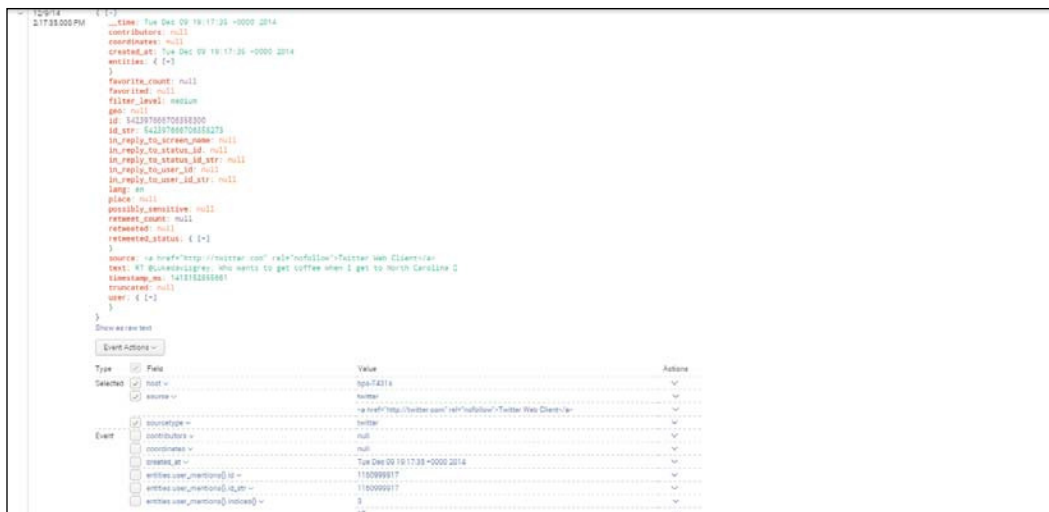
A Twitter event

There are several items to look closely at here:

1. **_time**: Splunk assigns a timestamp for every event. This is done in UTC (Coordinated Universal Time) time format.

2. **contributors**: The value for this field is null, as are the values of many Twitter fields.

3. **Retweeted_status**: Notice the {+} here; in the following event list, you will see there are a number of fields associated with this, which can be seen when the + is selected and the list is expanded. This is the case wherever you see a {+} in a list of fields:

Various retweet fields

In addition to those shown previously, there are many other fields associated with a tweet. The 140 character (maximum) text field that most people consider to be the tweet is actually a small part of the actual data collected.

The implied AND

If you want to search on more than one term, there is no need to add AND as it is already implied. If, for example, you want to search for all tweets that include both the text "coffee" and the text "morning", then use:

```
index=twitter text=*coffee* text=*morning*
```

If you don't specify `text=` for the second term and just put `*morning*`, Splunk assumes that you want to search for `*morning*` in any field. Therefore, you could get that word in another field in an event. This isn't very likely in this case, although `coffee` could conceivably be part of a user's name, such as "coffeelover". But if you were searching for other text strings, such as a computer term like `log` or `error`, such terms could be found in a number of fields. So specifying the field you are interested in would be very important.

The need to specify OR

Unlike AND, you must always specify the word OR. For example, to obtain all events that mention either coffee or morning, enter:

```
index=twitter text=*coffee* OR  text=*morning*
```

Finding other words used

Sometimes you might want to find out what other words are used in tweets about coffee. You can do that with the following search:

```
index=twitter text=*coffee* | makemv text | mvexpand text |
top 30 text
```

This search first searches for the word "coffee" in a text field, then creates a multivalued field from the tweet, and then expands it so that each word is treated as a separate piece of text. Then it takes the top 30 words that it finds.

You might be asking yourself how you would use this kind of information. This type of analysis would be of interest to a marketer, who might want to use words that appear to be associated with coffee in composing the script for an advertisement. The following screenshot shows the results that appear (1 of 2 pages). From this search, we can see that the words `love`, `good`, and `cold` might be words worth considering:

Search of top 30 text fields found with *coffee*

When you do a search like this, you will notice that there are a lot of filler words (a, to, for, and so on) that appear. You can do two things to remedy this. You can increase the limit for top words so that you can see more of the words that come up, or you can rerun the search using the following code. "Coffee" (with a capital C) is listed (on the unshown second page) separately here from "coffee". The reason for this is that while the search is not case sensitive (thus both "coffee" and "Coffee" are picked up when you search on "coffee"), the process of putting the text fields through the `makemv` and the `mvexpand` processes ends up distinguishing on the basis of case. We could rerun the search, excluding some of the filler words, using the code shown here:

```
index=twitter text=*coffee*  | makemv text | mvexpand text |
search NOT text="RT" AND NOT text="a" AND NOT text="to" AND
NOT text="the"  | top 30 text
```

Using a lookup table

Sometimes it is useful to use a lookup file to avoid having to use repetitive code. We'll present an example here that will help us with the situation presented in the preceding section. It would help us to have a list of all the small words that might be found often in a tweet just by the nature of each word's frequent use in language, so that we might eliminate them from our quest to find words that would be relevant for use in the creation of advertising. If we had a file of such small words, we could use a command indicating not to use any of these more common, irrelevant words when listing the top 30 words associated with our search topic of interest. Thus, for our search for words associated with the text "coffee", we would be interested in words like " dark", "flavorful", and "strong", but not words like "a", "the", and "then".

We can do this using a lookup command. There are three types of lookup commands, which are presented in the following table:

Command	Description
lookup	Matches a value of one field with a value of another, based on a .csv file with the two fields.
	Consider a lookup table named **lutable** that contains fields for **machine_name** and **owner**. Consider what happens when the following code snippet is used after a preceding search (indicated by . . . |):
	`. . .	lookup lutable owner`
	Splunk will use the lookup table to match the owner's name with its machine_name and add the machine_name to each event.
inputlookup	All fields in the .csv file are returned as results. If the following code snippet is used, both **machine_name** and **owner** would be searched:
	`...	inputlookup lutable`
outputlookup	This code outputs search results to a lookup table. The following code outputs results from the preceding research directly into a table it creates:
	`. . .	outputlookup newtable.csv saves`

The command we will use here is inputlookup, because we want to reference a .csv file we can create that will include words that we want to filter out as we seek to find possible advertising words associated with coffee. Let's call the .csv file `filtered_words.csv`, and give it just a single text field, containing words like "is", "the", and "then". Let's rewrite the search to look like the following code:

```
index=twitter text=*coffee*
| makemv text | mvexpand text
| search NOT [inputlookup filtered_words | fields text ]
| top 30 text
```

Using the preceding code, Splunk will search our Twitter index for `*coffee*`, and then expand the text field so that individual words are separated out. Then it will look for words that do NOT match any of the words in our `filtered_words.csv` file, and finally output the top 30 most frequently found words among those.

As you can see, the lookup table can be very useful. To learn more about Splunk lookup tables, go to `http://docs.splunk.com/Documentation/Splunk/6.1.5/SearchReference/Lookup`.

The built-in General Activity dashboard

Splunk has a built-in General Activity dashboard. To open it, perform the following steps:

1. Go to the Splunk home page.

2. On the **Twitter App** menu, click **Twitter General Activity.**

3. You will see a screen similar to the following:
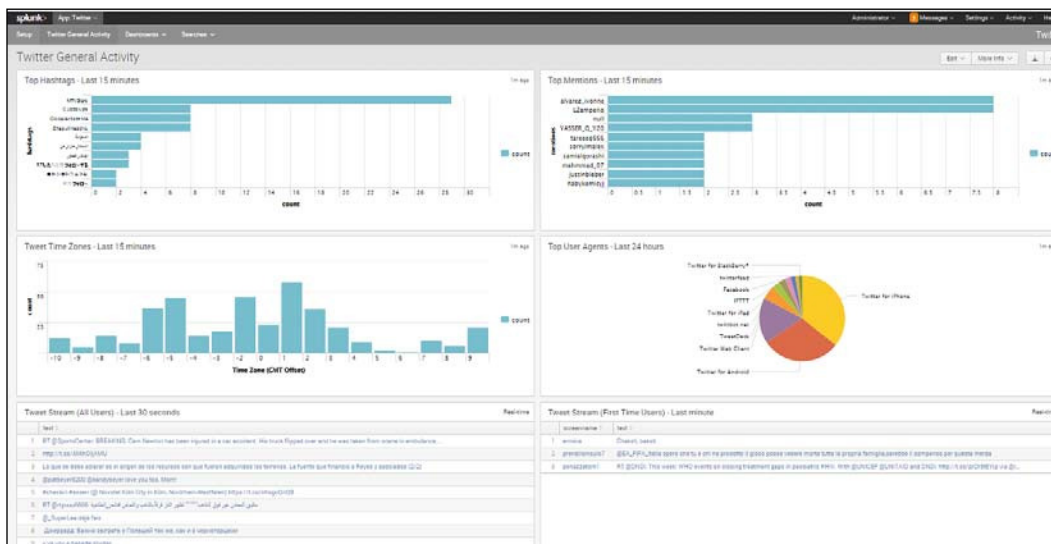
The Twitter General Activity Dashboard

You will see six dashboards, each of which displays interesting information about the 1% live Twitter stream that you have just sampled from. These dashboards are as follows:

1. Top Hashtags – last 15 minutes
2. Top Mentions – last 15 minutes
3. Tweet Time Zones – last 15 minutes
4. Top User Agents – last 24 hours
5. Tweet Stream (All Users) – last 30 seconds
6. Tweet Stream (First-Time Users) – last 30 seconds

We will examine panels 1 to 3 and 6 in detail in the following section.

The search code for the dashboard panels

Let's look at the search code for the first panel, Top Hashtags – last 15 minutes. To do this, click on the magnifying glass under the first chart, as shown in the following screenshot:
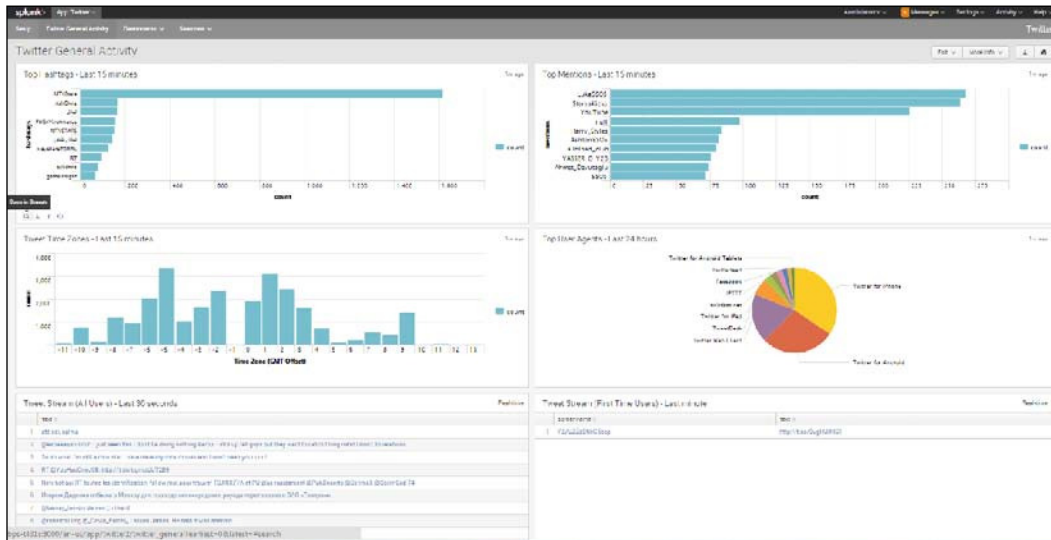
Click on the Magnifying Glass under the Top Hashtags panel

When you click on the magnifying glass, you will open the search window. If you click on the **Visualizations** tab and then click on **Bar** to show a bar chart, it looks similar to the panel in the previous dashboard. The search string used here is shown in the next section.

Top Hashtags – last 15 minutes

In the following code, we look at how the Top Hashtags panel is created in the dashboard:

```
index=twitter | rename entities.hashtags{}.text as hashtags |
fields hashtags | mvexpand hashtags | top hashtags
```

Let's break it down by the pipes shown in the code:

1. First, the Twitter index is selected.
2. The object `entities.hashtags{}.text` is renamed as `hashtags`.
3. The field `hashtags` is selected.
4. The field is expanded into multiple values.
5. The top hashtags are listed (the default is 10).

Top Mentions – last 15 minutes

In the code for the Top Mentions panel, we look at how to construct a panel of the top usernames mentioned in the last 15 minutes:

```
index=twitter | rename entities.user_mentions{}.screen_name as
mentions | fields mentions | mvexpand mentions | top mentions
```

Let's go through our construction of this code:

1. Again, use the Twitter index.
2. Rename the object `entities.user_mentions{}.screen_name` to `mentions`.
3. Select the field `mentions`.
4. Expand the `mentions` field into multiple values.
5. List the top 10 mentions.

Time Tweet Zones – 15 minutes

Here we show the code for creating the panel that shows the time zones from which most of the tweets in the last 15 minutes came:

```
index=twitter | rename user.utc_offset as z | seull | eval
z=round(z/3600) | stats count by z | sort +z
```

Here are the steps to create this code:

1. Use the Twitter index.

2. Rename the object `user.utc_offset`, which is the number of seconds of difference between the time and Greenwich Mean Time (GMT), as `z`.

3. Search all values of `z`. Treat those ending in `!` as null.

4. Evaluate `z` equal to z/3600. 3600 is the number of seconds in an hour. This gives you the number of hours plus or minus GMT.

5. Count the number of tweets occurring in each time zone.

6. Sort by the value of z in ascending order.

Tweet Stream (First-Time Users) – last 30 seconds

The Tweet Stream (First-Time Users) panel and the fifth panel show the text fields (or what we commonly think of as "tweets") in their entirety. The code for this is as follows:

```
index=twitter user.statuses_count=1 | rename user.screen_name
as screenname | table screenname text | sort -_time
```

The steps to create the code are shown here:

1. This time, when using the Twitter index, look just for those who have a `user.statuses_count` equal to 1.

2. Rename the object `user.screen_name` as `screenname`.

3. Create a table listing the `screenname` and the value of the text field associated with that `screenname`.

4. Sort with the most recent tweets at the top.

The built-in per-user Activity dashboard

There is another built-in dashboard for the Twitter app called the User Activity dashboard. To view this, perform the following steps:

1. Go to the Splunk home page.

2. On the Twitter app, click **Dashboards**, then **Per-User Activity**.

3. Examine the descriptions of each panel here.

First panel – Users Tweeting about @user (Without Direct RTs or Direct Replies)

The first panel of the dashboard will look something like the following screenshot, if you have used the popular username `@justinbieber`:

Twitter Per-User Activity for @justinbieber

The search commands used to create this panel are as follows:

```
index=twitter justinbieber NOT retweeted_status.user.screen_
name=justinbieber NOT in_reply_to_screen_name=justinbieber

| fields entities.user_mentions{}.screen_name user.followers_
count text user.screen_name

| rename entities.user_mentions{}.screen_name as mentions

| mvexpand mentions

| search mentions=justinbieber

| stats sum(user.followers_count) as Impressions max(user.
followers_count) as Followers count as Tweets values(text) as
"Tweet Text" by user.screen_name

| sort 20 -Impressions
```

We won't go through this search string in detail. But basically, the preceding commands look for a username that is not a retweet or used in a reply, and then list them. An agent for a celebrity, or a social media or PR specialist for a company, would be interested in this type of analysis.

Second panel – Users Replying to @user

The following screenshot shows counts for **Impressions**, **Followers**, **Tweets**, and **Tweet Text** for **@user**:

Counts for Impressions, Followers, Tweets, and Tweet Text for @user

The code for creating this panel is as follows:

```
index=twitter justinbieber in_reply_to_screen_name=justinbieber
| fields entities.user_mentions{}.screen_name user.followers_
count text user.screen_name
| rename entities.user_mentions{}.screen_name as mentions |
mvexpand mentions
| search mentions=justinbieber
| stats sum(user.followers_count) as Impressions max(user.
followers_count) as Followers count as Tweets values(text) as
"Tweet Text" by user.screen_name
| sort 20 -Impressions
```

This panel looks at the top 20 `user.screen_names` that tweeted `@justinbieber` during this time period, and then lists the sum of the followers who saw each tweet (renamed **Impressions**) as well as the number of followers. (Notice that user **MyDreamisDrew1** tweeted twice, so the impressions are double the size of the Followers.)

Third panel – Users Retweeting @user

In the following screenshot, we see counts for **Impressions**, **Followers**, **Tweets**, and **Tweet Text** for users retweeting `@user`:

Counts for Impressions, Followers, Tweets and Tweet Text for users retweeting @user

The code used here is as follows:

```
index=twitter retweeted_status.user.screen_name=justinbieber

| stats sum(user.followers_count) as Impressions max(user.
followers_count) as Followers count as Tweets values(text) as
"Tweet Text" by user.screen_name

| sort 20 -Impressions
```

This panel takes those who retweeted a tweet containing `@justinbieber`, then creates the **Impressions** and **Followers** fields and shows them, and then lists those tweets in descending order.

Fourth panel – Users Tweeting about #hashtag

In the fourth panel, we see **Impressions**, **Followers**, **Tweets**, and **Tweet Text** for users tweeting about `#hashtag`:

Impressions, Followers, Tweets, and Tweet Text for users tweeting about #hashtag

The code used here is as follows:

```
index=twitter justinbieber in_reply_to_screen_name=justinbieber

| fields entities.user_mentions{}.screen_name user.followers_
count text user.screen_name

| rename entities.user_mentions{}.screen_name as mentions |
mvexpand mentions |
 search mentions=justinbieber

| stats sum(user.followers_count) as Impressions max(user.
followers_count) as Followers count as Tweets values(text) as
"Tweet Text" by user.screen_name

| sort 20 -Impressions
```

This shows the `screen_names` of the people tweeting in reply to a tweet with `@justinbieber` that use a hashtag, then shows the impressions and the followers for each, the number of times they tweeted (during this time period), and the actual tweet text. The table is sorted by the number of impressions (descending).

# Creating dashboard panels with Twitter data

In the previous section, we shown and described examples of how dashboard panels can be made using Twitter data. Before ending this chapter, we present two additional examples.

Monitoring your hashtag

You might be interested in seeing what kind of traffic a hashtag was getting at a particular time. This could be to follow what people were saying about your company or a public figure. The search code to do this is presented here, for a hashtag of your choice, indicated here by `my_hashtag`:

```
index=twitter

| rename entities.hashtags{}.text as hashtags

| fields hashtags

| mvexpand hashtags

| where like(hashtags,"my_hashtag")

| timechart count span=10d
```

Let's look at this code carefully. We show each pipe on a separate line, just to be clear. We'll start with line 2, where we rename each instance of the text of the hashtag entity as hashtags. In line 3, we limit our pool of data to the field **hashtags**. In line 4, we use the `mvexpand` command to separate the hashtag field into multiple values, as we have seen before. Then we look for a specific hashtag, which is given in quotes. Here, we have used `gameinsight` in place of `my_hashtag`, which was a popular hashtag at the time this book was written. We then use the `timechart` command to find the count of hashtags during the last 10 days:
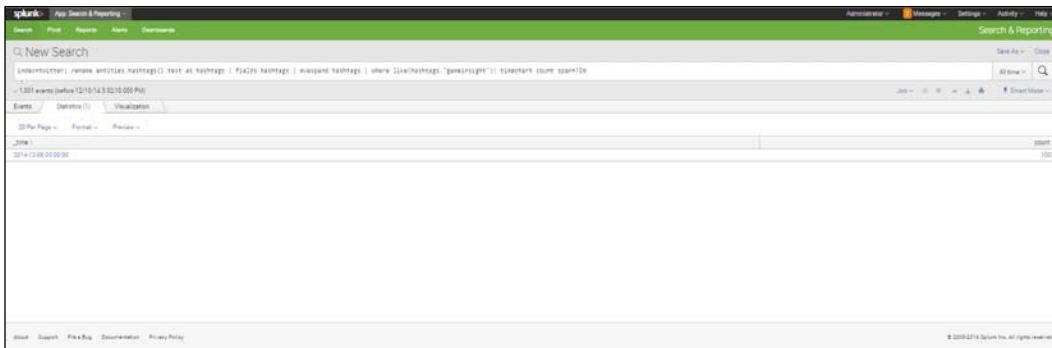
This Search string counts the number of times "gameinsight" appeared

Creating an alphabetical list of screen names for a hashtag

It might be that you are interested in looking at exactly who is tweeting a particular hashtag and precisely what they are saying. To do this, you can use the following code, replacing `gameinsight` with the hashtag of your choice:

```
index=twitter
| rename entities.hashtags{}.text as hashtags
| rename user.screen_name as screenname
| fields screenname, hashtags, text
| mvexpand hashtags
| where like(hashtags, "gameinsight")
| table screenname, text
| sort screenname
```

Going through this code, you can see that we have taken the `user.screen_name` and renamed it `screenname`, then limited our data to screenname, hashtags, and text. In the fourth line, we go on to expand the hashtags into multivalued fields. Then we limit our hashtags to those including `gameinsight`. Finally, we create a table showing the screenname and the text of the tweets, which is in alphabetical order by screenname. This way, we can see who is saying what regarding a particular hashtag. Given the increasing importance of one's image on social media, this type of analysis, as well as the others discussed in this chapter, can be extremely useful.

Summary

In this chapter, we have used the app for Twitter data to learn about how to input live data streams. We have explored in detail the built-in dashboards that come with this app, and have learned about the commands behind each panel and how they work in Splunk. We have also learned more about doing more detailed searches in Splunk. And we have additionally learned about how to use a lookup table to aid our searches.

In the next chapter, we will go on to learn the useful skill of using Splunk to create alerts.

7
Monitoring and Creating Alerts in Splunk

In the past six chapters, we have introduced you to Splunk, its apps ecosystem, and how they work with data. We have also shown you how to use Splunk to create reports and dashboards. In this chapter, we will cover how to monitor and create alerts in Splunk. We will cover the following topics:

- Monitoring your system in Splunk
- Looking at geographic data
- What an alert is

Monitoring your system in Splunk

We often want to monitor data so that we can see what is happening with it and what it indicates about the system that is creating it. In a business, sensors, logs, and other types of data are produced that you need to keep track of by using metrics. You can set up reports to monitor these metrics using Splunk. Here are some ways to answer questions that businesses might have.

Analyzing the number of system users

Imagine that you've been having problems over the last couple of days and you want to simply measure how many people are on your system during each hour. To do this, enter the following code into the search bar:

```
sourcetype=access_* earliest=-2d@h latest=now | timechart count
```

Here we see the use of two time modifiers, **earliest** and **latest**, which can be used to indicate the relative start time that you want to use as well as the end time. In this case, `earliest=-2d@h` means that you should include events that occurred within the last two days (**-2d**), and round to the nearest hour (**@h**). When you use this code, the timechart count pipe provides a count of events for each hour over the last two days.

You will see a chart like this:

Using Time Modifiers (Earliest and Latest) with timechart

Discovering client IP codes that have not been used on certain days

You might want to find out if some clients have not used your system in the last few days. You can check this out with the code shown as follows:

```
* clientip !=211* | timechart count
```

This code searches all events for those where the clientip is not equal to `211*` (`!=` means not equal and `211*` refers to all IP addresses beginning with `211`). The first part of an IP address (the first three digits) usually signifies the network. The following screenshot shows are the results, which show the IP codes beginning with `211` that engaged with the site a lot during one month, but did not during the next three months. Your data probably does not look like this though. We found this pattern because we downloaded the `tutorialdata.zip` twice, with three months between the first and the last download:

Search for Client IP Addresses that are Not Equal to 211* Using Timechart

Checking the IP status

You might wish to test and see how successful your website traffic is. This can be done by looking at status codes. Successful status can be defined in various ways, but here it is defined as being coded from greater or equal to 200 to less than 300. You can use the following code:

```
* earliest=-2d latest=-1d status>=200 status<300 | timechart count BY
status
```

The code indicates to include all events from 2 days ago to 1 day ago that have a status greater than 200 and less than 300, and create a chart showing hours by status. The timechart defaults to hours, given the setting of one day's time.

The table produced looks like the following screenshot:

Timechart of Counts of Status of Events in the Last Day, Ranging from >= 200 to <300

You can easily turn this into a column chart by clicking the **Visualizations** tab and selecting **Column**. Likewise, you could turn it into many other types of charts. If you use a column chart and don't need a legend (since there is only one color bar), you can get rid of the legend by selecting **Format**, **Legend**, **None**:
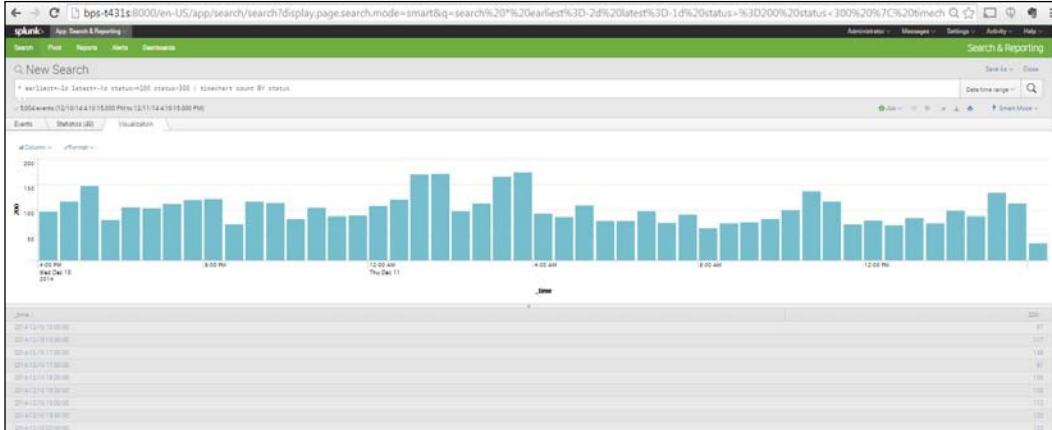
Column Chart Showing Counts of Status of Events in the Last Day, Ranging from >= 200 to <300

Looking at geographic data

Now let's look at some geographic data. Geographic data helps business analysts know where their business is coming from. Splunk has some built-in commands – iplocation and geostats – that will help us find and analyze geographic data. We will learn about these commands in the following sections.

Using the iplocation command

The iplocation command extracts geographic locations from a third-party dataset to help the Splunk user easily obtain geographic values for a client IP or Internet protocol address (the `clientip` field). The `iplocation` command, by default, returns the `Country`, `City`, `Region`, `lat` (latitude), and `lon` (longitude) fields associated with each event. In the following code snippet, we have used the buttercupgames data (used in earlier chapters) and created a table of the top 15 countries with the greatest counts:

```
buttercupgames | iplocation clientip | top limit=15 Country
```

As you can see here, Splunk gives both the counts and the percentages in its output:

Top 15 Countries in Terms of Counts of IP Addresses

We can do the same type of analysis to create a table of the five most common cities that appear in our data, using the following code:

```
buttercupgames | iplocation clientip | top limit=5 City
```

And we get the following result:

Top 15 Cities in Terms of Counts of IP Addresses

Interestingly, you will notice that in this list the top geographic entity has no name. This is because it represents a group of all the `clientip` values that were not matched with a city.

Using the geostats command

Another useful tool for analyzing data geographically is the geostats command. This command allows us to easily take the lat and lon fields created by the iplocation command (and based on the clientip), and uses these to cluster the counts geographically and map them.

The code is simple:

```
buttercupgames | iplocation clientip | geostats count
```

And our results help us to quickly visualize the locations from which our data is coming:
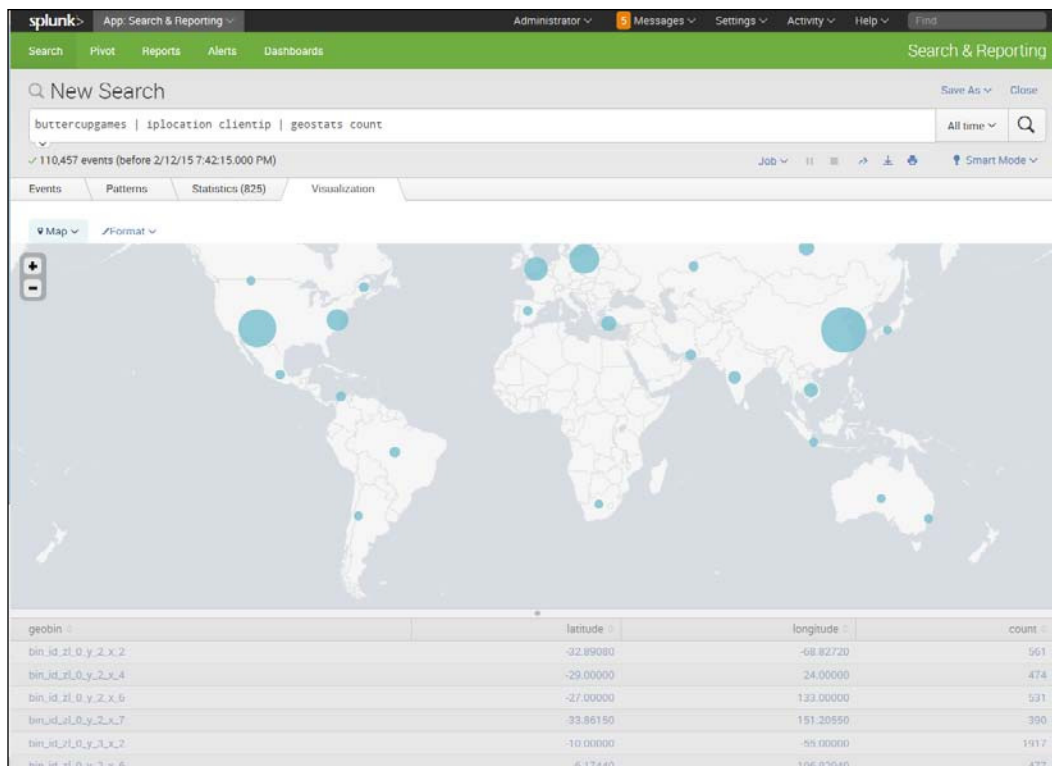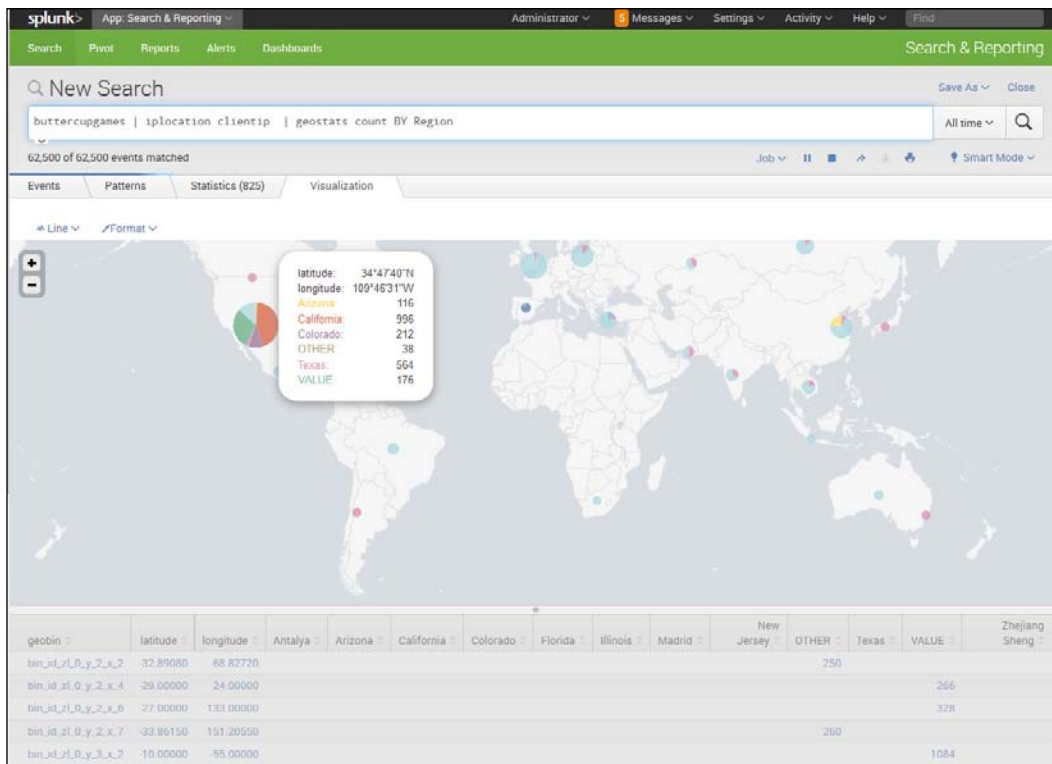

Mapped Locations for Counts of Client IP Addresses

In the map shown here, you can see that longitude and latitude data has been used to cluster the events into the geobins listed on the left. The counts and percentages falling into each of these geobins is shown, and the size of the bubbles indicate the relative counts in each geobin.

We can also search by using the field Region, using the following code:

```
buttercupgames | iplocation clientip  | geostats count BY Region
```

The result will be as shown in the following screenshot:

Mapped Counts of Client IP Addresses by Region

Notice that in this screenshot, the counts of the clientip addresses coming from different regions have been mapped. However, due to the built-in defaults for the geostats command, large numbers coming from the Southwest area of the United States have been grouped together in a colored pie chart. When you mouseover the chart, you see the counts from **California**, **Colorado**, **Other**, **Texas**, and **Value**. You can zoom in and out on the map using the **+** and **-** signs on the left. When you do this, you can see bubbles representing **California**, **Colorado**, and **Texas**.

But you may be asking, what do the **Other** and **Value** labels mean? The **Other** category represents those client IP addresses that are associated with a count that is less than the default `globallimit`, which is 10. If you add the code `globallimit=0`, you will be able to see the mapped locations of all the client IP addresses, regardless of how many share each location. **Value** is used to represent those locations for which lat (latitude) and lon (longitude) cannot be determined. So, we can get rid of the **Other** category, but still have many in the "Value" category, when we use the following code:

```
buttercupgames | iplocation clientip  | geostats count BY Region
globallimit=0
```

The resulting map shows this change when we zoom in on the region:
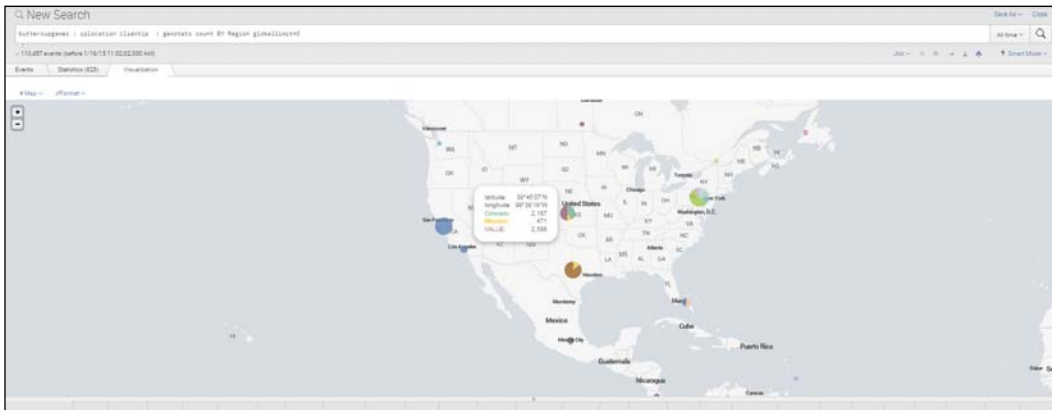
Results of Zooming In

Notice how the clustering algorithm used this time has grouped **Colorado** with **Missouri**. No **Other** category is found, but the **Value** label still applies to a large number of events.

Performing alerts in Splunk

Alerts are ways that business people, workers, managers, and others can receive notifications about something that they need to know has happened, or about something that is likely to happen soon. The usefulness of alerts in this age of machine data cannot be overstated; the amount of information out there is growing rapidly and it is important that it be monitored, and done so using automatic controls. It is beyond human capability to check large data streams, given the speed and volume at which it comes in. Furthermore, problems also need to be caught early. Fortunately, automatic alerts provide a solution.

Once an alert is set, there are various ways to convey alert information. Alerts can be set to send a message or e-mail, set off an alarm, run a script, produce an ad-hoc report, or take any number of other actions that can help to let people know something they need to become aware of.

> The alert feature will be accessible only to those with a full enterprise system.

Types of alerts

There are three basic types of alerts in Splunk. These are listed and described as follows:

1. **Per result alert**: This type of alert takes place when a trigger condition is met. So, for example, if a trigger is set to indicate when a product's sales have dropped below 70 percent of their average normal sales for a particular season, an alert like this would notify sales managers that there may be a problem.

2. **Scheduled alert**: A scheduled alert is set to occur on a schedule, set to notify according to set intervals, if a condition is met.

3. **Rolling-window alert**: This type of alert takes place if, within a rolling time window, an action or set of actions occurs. Such an alert can be particularly useful for fraud protection; for instance, actions such as large expenses charged in a short period of time can set off such alerts, allowing information about the problem to be shared quickly with those who need to know.
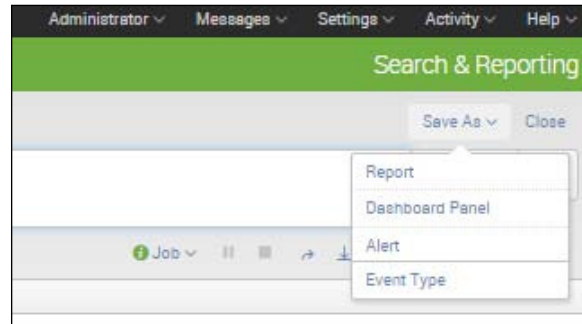
Setting an alert

Here we are going to set an alert based on a saved search. The search we will use is the number of products sold in the last week. First, let's create the search using the following code:

1. Type the following code in the search bar:

    ```
    sourcetype=access_* earliest=-7d@d latest=now action="purchase"
    | stats count(eval(action="purchase")) AS "Total Products Sold
    Last Week"
    ```

2. Now let's go on to create an alert.

3. In the right-hand corner, you will see a **Save As** icon. Click it and you will see a menu like the following screenshot:

Saving an Alert

4. Click on **Alert**.

5. In the screen that appears, select a name for your alert. Here we type `Last Week Purchases Alert`, but any descriptive name would work.

6. Choose **Scheduled** for the **Alert type**.

7. In the **Time Range** area, set your alert to **Run** every week (note that you could also choose other time periods), and then, in the boxes below, select the day and time you'd like to run it each week.

8. In the **Trigger** area, notice that you can choose a number of Trigger characteristics. Here, the ones chosen include **Number of results** (could also have chosen **Number of Hosts**, **Number of Sources**, or **Custom**), and **is less than** (could also have chosen other similar options) and a number (in this case **300**). This alert is being set up to let management know when purchases during a given week drop below 300.

9. Then click **Next**.

Save as Alert screen

10. You will see a box asking you what type of action to trigger. Here you can choose from several options, as shown in the following screenshot:


Save as Alert screen showing other options

11. Select **List in Triggered Alerts**, and, under **Severity**, **High**. Notice that you could also send an e-mail or run a script. If you choose to run a script, you could design it to take actions like shutting down the system, disallowing any more attempts at user log in, and other steps. Running a script is often used to curtail further access or to prevent problems until the system is fixed.

12. Click under **When triggered** | **execute action** | **Once**.

13. Notice that there is an option to **Throttle**. If you check the **Throttle** box, an area opens that asks you how long after executing actions to suppress alerts for. You can choose a number and a period of time (seconds, minutes, hours, or days). Throttling prevents the announcement of more alerts until a specified time after the first alert is issued. When you set a time for throttling, it needs to be based on the specifics involved. You wouldn't want to set it so that a crucial alert would be prevented, but you also don't need to see every alert go off once you know there is a problem. There is a fine balance to strike in order to set the throttling for the right amount of time. Here, however, we are not concerned about throttling and leave the box unchecked.

14. Finally, you choose whether to share the alert or not, and who to share it with. In the box you see here, select **Shared in App**.

15. Click **Save**.

Managing alerts

Alerts are managed within the Alert Manager, where you can choose to search, filter, or view the alerts according to the application (indicated by the **This App's** button), the severity of the alert, and the alert itself. You can also delete alerts. To view the results of the alert we just created, take the following steps:

1. From the **Search** Menu, click on **Alerts**.

2. If you have created an alert as indicated previously, you should see something like the following screenshot:
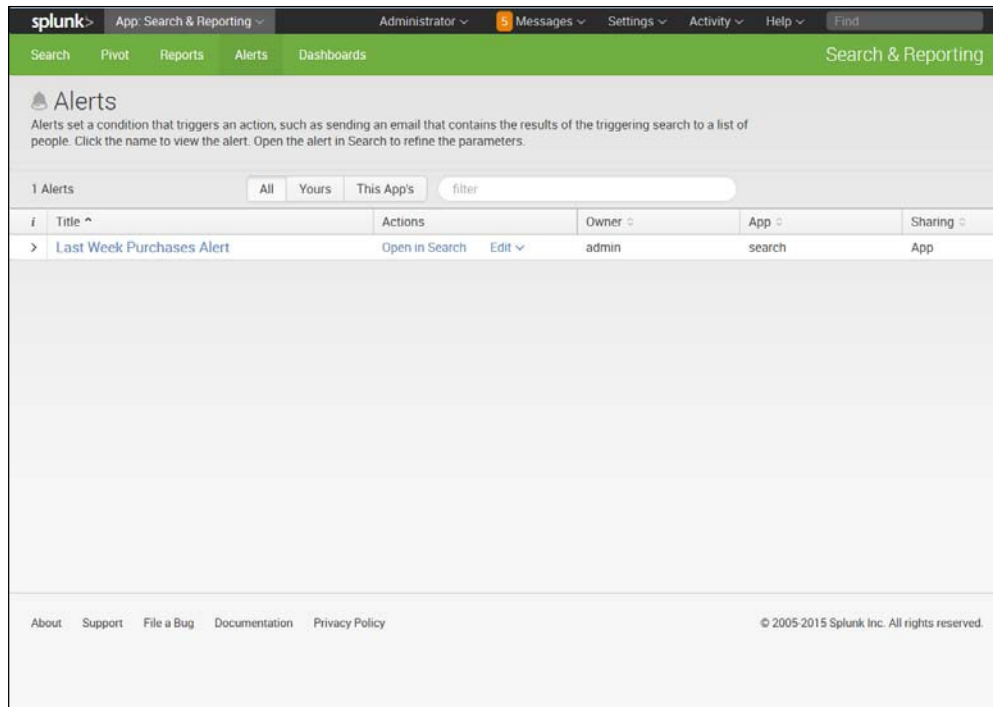
List of Alerts

3. Notice that you can select the **All**, **Yours**, or **This App's** buttons, and that you can filter the alerts by typing in a word or phrase.

4. You could edit the alert from here. Click the app you want to select, then **Edit**, and you can see the options of choosing various ways you can edit the alert, clone it, or delete it:
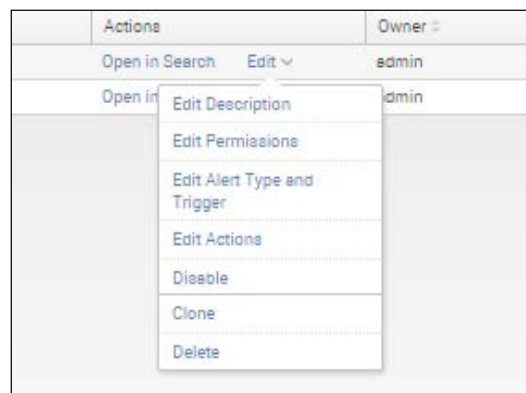
Edit the Alert

5. We have only one alert to show here, but we can go ahead and click it and see the results:
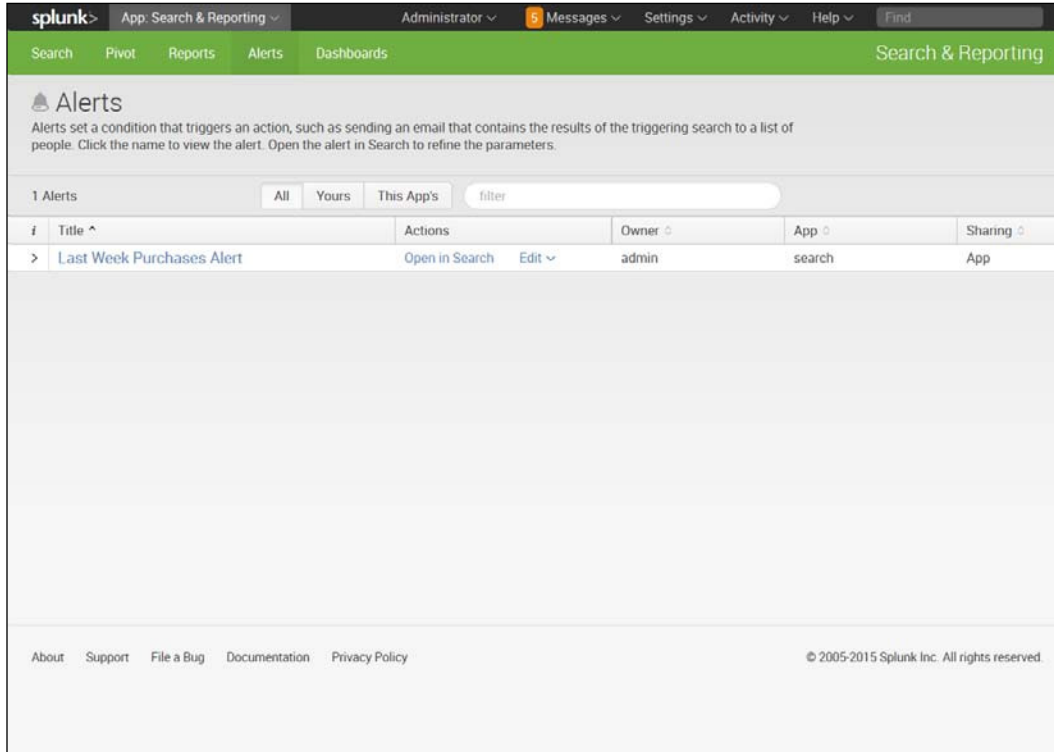
6. Notice that there have been no fired events for this alert, meaning that so far the purchases in the week have been greater than or equal to 300.

Another example of an alert

We'll do another example of an alert so that you can see what happens when an alert is triggered. This time, we will use the following search code:

```
sourcetype=access_* earliest=-3d latest=now action="purchase"
| stats count(eval(action="purchase")) AS "Total Products Sold in last
3 days"
```

The criteria we use are as follows:

- **Title**: **Total Products Sold in Last 3 Days**
- **Alert Type**: **Scheduled**
- **Time Range**: **Run every hour**
- **Schedule**: **At 0 minutes past the hour**
- **Trigger Condition**: **Number of Results**
- **Trigger if number of results**: **is Less than**, **1000**

The criteria for the alert are specified as shown in the following screenshot:

Alert for Total Products Sold in Last 3 Days

When we click on the **Alert** we created, we can see that the alert has been triggered. We can attain information on the time, type, condition, and actions of the alert, as well as the app associated with it (which, in this case, is the search app). Permissions are also shown, and private is indicated here:

Information from Triggered Alert

Summary

In this chapter, we learned how to use Splunk to monitor our data and to create alerts to let us quickly learn about any issues or foreseeable trends in the data. We have also learned about the different kinds of alerts, as well as about how to create settings so that the alerts will be useful for the different ways we can use them.

This brings our book to a conclusion, but please be aware that there is still a lot to learn about this useful software. We encourage you to delve deeper into the many ways you can use Splunk to learn more about your organizational and operational data, and to make your work more efficient and accurate. We suggest going to `www.splunk.com` and selecting **Resources** to see where you can get tutorials, videos, and information on apps, as well as learn many other ways you can build on your basic knowledge of Splunk. Happy Splunking!

Index

round(X,Y) 42
stat, combining with 42, 43
upper (X) 42
using 42
event 13
event types
about 13
reference link 13
setting 54-57

F

field extractor 58
fields 14
filter commands, Search Processing Language (SPL)
dedup 28
head 28
search 28
tail 28
where 28

G

geographic data, extracting
about 124
geostats command used 126-128
iplocation command used 124-126
geostats command
used, for extracting geographic data 126-128
grouping command, Search Processing Language (SPL)
transaction 29

H

Hadoop 22
hashtag
alphabetical list of screen names, creating for 119
monitoring 118

I

indexed data
bringing in 25
using 24

inputlookup command 110
installation, Splunk 6
iplocation command
used, for extracting geographic data 124-126
IP status
checking 123, 124

L

legend
placement, modifying of 70
list, of indexes
viewing 24, 25
lookup commands
inputlookup 110
lookup 110
outputlookup 110
lookup table
reference link 111
using 109, 110

M

Mac OS X
Splunk, setting up for 7
Marker Gauge
creating 78, 79

O

outputlookup command 110

P

per result alert 129
pipes
used, for processing data 26
pivot table
creating 80-84

Q

quotes
using 33

R

radial gauge
  about 76
  creating 76, 77
regular expressions
  reference link 58
report builder 59
reporting
  preparing for 51
reporting commands, Search Processing
       Language (SPL)
  chart 30
  rare 30
  stat 30
  timechart 30
  top 30
report, of count
  creating 59-62
rolling-window alert 129

S

scattergram
  about 74
  creating 74
scheduled alert 129
search
  about 37
  data, collecting for 22
  rules, for performing 37, 38
search code, for dashboard panels
  about 112
  Time Tweet Zones - 15 minutes 113
  Top Hashtags - last 15 minutes 113
  Top Mentions - last 15 minutes 113
  Tweet Stream (First-Time Users) -
      last 30 seconds 114
Search Processing Language (SPL) 26
Search Processing Language (SPL),
      commands
  eval 31
  fields 31
  filter 27
  group 27
  lookup 31
  replace 31

  report 27
  sort 27
simple searches
  about 106
  performing 31-34
sort commands, Search Processing
       Language (SPL)
  sort 0 anyfield 29
  sort 1000 fieldone -fieldtwo 29
  sort -fieldone, +fieldtwo 29
sourcetype
  about 14
  access_combined 14, 25
  apache_error 14, 25
  cisco_syslog 14, 25
  specifying 25
  websphere_core 14
sparkline panel
  creating 73
Splunk
  about 5
  alerts, performing in 128
  app, installing in 90, 91
  big data, aspects 10
  data, indexing with 23
  data, obtaining into 15-20
  data, processing with pipes 26
  installing 6
  setting up 8
  setting up, for Mac OS X 7
  setting up, for Windows 6, 7
  setup instructions 6
  URL 6
  URL, for documentation 9, 12
Splunk API 8
Splunk applications
  about 85
  creating 90
  environment 89
  finding 86
  managing 92, 93
  URL 87
Splunk, data sources
  about 12
  data files 12
  machine data 12

Thank you for buying
Splunk Essentials

About Packt Publishing

Packt, pronounced 'packed', published its first book, *Mastering phpMyAdmin for Effective MySQL Management*, in April 2004, and subsequently continued to specialize in publishing highly focused books on specific technologies and solutions.

Our books and publications share the experiences of your fellow IT professionals in adapting and customizing today's systems, applications, and frameworks. Our solution-based books give you the knowledge and power to customize the software and technologies you're using to get the job done. Packt books are more specific and less general than the IT books you have seen in the past. Our unique business model allows us to bring you more focused information, giving you more of what you need to know, and less of what you don't.

Packt is a modern yet unique publishing company that focuses on producing quality, cutting-edge books for communities of developers, administrators, and newbies alike. For more information, please visit our website at `www.packtpub.com`.

About Packt Enterprise

In 2010, Packt launched two new brands, Packt Enterprise and Packt Open Source, in order to continue its focus on specialization. This book is part of the Packt Enterprise brand, home to books published on enterprise software – software created by major vendors, including (but not limited to) IBM, Microsoft, and Oracle, often for use in other corporations. Its titles will offer information relevant to a range of users of this software, including administrators, developers, architects, and end users.

Writing for Packt

We welcome all inquiries from people who are interested in authoring. Book proposals should be sent to `author@packtpub.com`. If your book idea is still at an early stage and you would like to discuss it first before writing a formal book proposal, then please contact us; one of our commissioning editors will get in touch with you.

We're not just looking for published authors; if you have strong technical skills but no writing experience, our experienced editors can help you develop a writing career, or simply get some additional reward for your expertise.
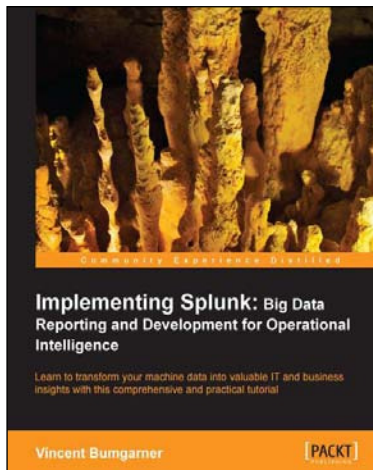
Implementing Splunk: Big Data Reporting and Development for Operational Intelligence

ISBN: 978-1-84969-328-8   Paperback: 448 pages

Learn to transform your machine data into valuable IT and business insights with this comprehensive and practical tutorial

1. Learn to search, dashboard, configure, and deploy Splunk on one machine or thousands.

2. Start working with Splunk fast, with a tested set of practical examples and useful advice.

3. Step-by-step instructions and examples with a comprehensive coverage for Splunk veterans and newbies alike.
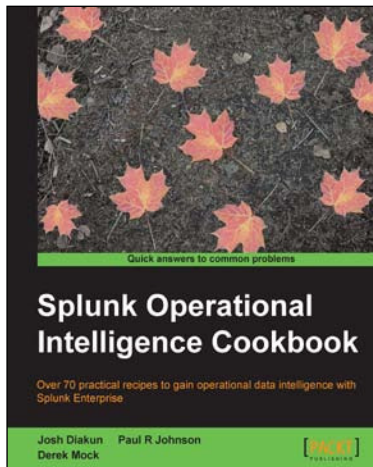
Splunk Operational Intelligence Cookbook

ISBN: 978-1-84969-784-2   Paperback: 414 pages

Over 70 practical recipes to gain operational data intelligence with Splunk Enterprise

1. Learn how to use Splunk to effectively gather, analyze, and report on the operational data across your environment.

2. Expedite your operational intelligence reporting, be empowered to present data in a meaningful way, and shorten the Splunk learning curve.

3. Easy-to-use recipes to help you create robust searches, reports, and charts using Splunk.

Mastering Splunk

ISBN: 978-1-78217-383-0   Paperback: 344 pages

Optimize your machine-generated data effectively by developing advanced analytics with Splunk

1. Develop simple applications into robust, feature-rich applications to search, monitor, and analyze machine-generated big data with ease.

2. Learn about lookups, indexing, dashboards, navigation, advances transaction with examples.

3. Understand the key features of Splunk by exploring real-world examples and apply the technology in your database.
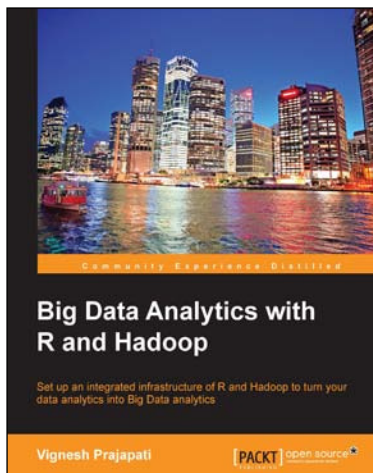
Big Data Analytics with R and Hadoop

ISBN: 978-1-78216-328-2   Paperback: 238 pages

Set up an integrated infracrcture of R and Hadoop to turn your data analytics into Big Data analytics

1. Write Hadoop MapReduce within R.

2. Learn data analytics with R and the Hadoop platform.

3. Handle HDFS data within R.

4. Understand Hadoop streaming with R.

5. Encode and enrich datasets into R.

Please check **www.PacktPub.com** for information on our titles