Report on

NetFlow

Flow Tool

Hadoop Setup

By:

Yogesh Dorbala, cs1100112

Kishore Rajendra, cs1100118

CONTENTS

NetFlow

Introduction – Who? :

A granular understanding of how bandwidth is being used is extremely important in IP networks today. Packet and byte interface counters are useful but understanding which IP addresses are the source and destination of traffic and which applications are generating the traffic is invaluable.

Monitoring IP traffic flows facilitates more accurate capacity planning and ensures that resources are used appropriately in support of organizational goals. It helps determine where to apply Quality of Service (QoS), optimize resource usage and it plays a vital role in network security to detect Denial-of-Service (DoS) attacks, network-propagated worms, and other undesirable network events.

Definition – What? :

NetFlow is a feature that was introduced on Cisco routers that give the ability to collect IP network traffic as it enters or exits an interface. By analyzing the data that is provided by NetFlow a network administrator can determine things such as the source and destination of the traffic, class of service, and the cause of congestion.

NetFlow consists of three components: flow caching, Flow Collector, and Data Analyzer. It creates an environment where administrators have the tools to understand who, what, when, where, and how network traffic is flowing

Information Gathering – How? :

Routers and switches that support NetFlow can collect IP traffic statistics on all interfaces where NetFlow is enabled, and later export those statistics as NetFlow records, toward at least one NetFlow collector - typically a server that does the actual traffic analysis

IP Packet attributes used by NetFlow:

- IP source address
- IP destination address
- Source port
- Destination port
- Layer 3 protocol type
- Class of Service
- Router or switch interface

These attributes are the IP packet identity or fingerprint of the packet and determine if the packet is unique or similar to other packets

Accessing the Data:

There are two primary methods:

1) **Command Line Interface:**

- The "Command Line Interface" (CLI) with show commands or utilizing an application reporting tool. If you are interested in an immediate view of what is happening in your network, the CLI can be used.
- NetFlow CLI is very useful for troubleshooting.

2) **NetFlow collector:**

- The other choice is to export NetFlow to a reporting server or what is called the "NetFlow collector". The NetFlow collector has the job of assembling and understanding the exported flows and combining or aggregating them to produce the valuable reports used for traffic and security analysis
- NetFlow collector – typically a server that does the actual traffic analysis, then processes the data to perform the traffic analysis and presentation in a user-friendly format.

Explaining the need – When? :

This flow information is extremely useful for understanding network behavior

- Source address allows the understanding of who is originating the traffic
- Destination address tells who is receiving the traffic
- Ports characterize the application utilizing the traffic
- Class of service examines the priority of the traffic
- The device interface tells how traffic is being utilized by the network device
- Tallied packets and bytes show the amount of traffic

Applications – Where? :

- Network Monitoring
- Network planning
- Security Analysis
- Application Monitoring
- User Monitoring
- Traffic Engineering
- Peering Agreement
- Usage-base Billing
- Destination sensitive billing

References

http://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios
netflow/prod_white_paper0900aecd80406232.pdf

https://nsrc.org/workshops/2012/apricot-nmm/materials/netflow.pdf

Flow Tool in Ubuntu

Introduction

Flow-tools is library and a collection of programs used to collect, send, process, and generate reports from Flow data. The tools can be used together on a single computer or server or distributed to multiple servers for large deployments. The flow-tools library provides an API for development of custom applications for NetFlow export versions 1, 5, 6 and the 14 currently defined version 8 subversions.

A Perl and Python interface have been contributed and are included in the distribution. Flow data is collected and stored by default in host byte order, yet the files are portable across big and little endian architectures.

Commands that utilize the network use a localip/remoteip/port designation for communication. "localip" is the IP address the host will use as a source for sending or bind to when receiving NetFlow PDU's (i.e., the destination address of the exporter).

Configuring the "localip" to 0 will force the kernel to decide what IP address to use for sending and listen on all IP addresses for receiving. "remoteip" is the destination IP address used for sending or the expected address of the source when receiving.

If the "remoteip" is 0 then the application will accept flows from any source address. The "port" is the UDP port number used for sending or receiving. When using multicast addresses the localip/remoteip/port is used to represent the source, group, and port respectively.

Flows are exported from a router in a number of different configurable versions. A flow is a collection of key fields and additional data. The flow key is {srcaddr, dstaddr, input, output, srcport, dstport, prot, ToS}. Flow-tools supports one export version per file.

Fields contained in the exported list

Export versions 1, 5, 6, and 7 all maintain {nexthop, dPkts, dOctets, First, Last, flags}, ie the next-hop IP address, number of packets, number of octets (bytes), start time, end time, and flags such as the TCP header bits. Version 5 adds the additional fields {src_as, dst_as, src_mask, dst_mask}, i.e., source AS, destination AS, source network mask, and destination network mask.

Version 7 which is specific to the Catalyst switches adds in addition to the version 5 fields {router_sc}, which is the Router IP address which populates the flow cache shortcut in the Supervisor. Version 6 which is not officially supported by Cisco adds in addition to the version 5 fields {in_encaps, out_encaps, peer_nexthop}, ie the input and output interface encapsulation size, and the IP address of the next hop within the peer.

Version 1 exports do not contain a sequence number and therefore should be avoided, although it is safe to store the data as version 1 if the additional fields are not used. Version 8 IOS NetFlow is a second level flow cache that reduces the data exported from the router.

There are currently 11 formats, all of which provide {dFlows, dOctets, dPkts, First, Last} for the key fields.

- 8.1 - Source and Destination AS, Input and Output interface
- 8.2 - Protocol and Port
- 8.3 - Source Prefix and Input interface
- 8.4 - Destination Prefix and Output interface
- 8.5 - Source/Destination Prefix and Input/Output interface
- 8.9 - 8.1 + ToS
- 8.10 - 8.2 + ToS
- 8.11 - 8.3 + ToS
- 8.12 - 8.5 + ToS
- 8.13 - 8.2 + ToS
- 8.14 - 8.3 + ports + ToS

Version 8 CatIOS NetFlow appears to be a less fine grained first level flow cache.

- 8.6 - Destination IP, ToS, Marked ToS,
- 8.7 - Source/Destination IP, Input/Output interface, ToS, Marked ToS,
- 8.8 - Source/Destination IP, Source/Destination Port,
  - Input/Output interface, ToS, Marked ToS,

Commands in Flow Tool

- **flow-capture** - Collect, compress, store, and manage disk space for exported flows from a router.
- **flow-cat** - Concatenate flow files. Typically flow files will contain a small window of 5 or 15 minutes of exports.
- **flow-fanout** - Replicate NetFlow datagrams to unicast or multicast destinations
- **flow-report** - Generate reports for NetFlow data sets. Reports include source/destination IP pairs, source/destination AS, and top talkers.
- **flow-tag** - Tag flows based on IP address or AS #. Flow-tag is used to group flows by customer network.
- **flow-filter** - Filter flows based on any of the export fields.
- **flow-import** - Import data from ASCII or cflowd format.
- **flow-export** - Export data to ASCII or cflowd format.
- **flow-send** - Send data over the network using the NetFlow protocol.
- **flow-receive** - Receive exports using the NetFlow protocol without storing to disk like flow-capture.
- **flow-gen** - Generate test data.
- **flow-dscan** - Simple tool for detecting some types of network scanning and Denial of Service attacks.
- **flow-merge** - Merge flow files in chronological order.
- **flow-xlate** - Perform translations on some flow fields.
- **flow-expire** - Expire flows using the same policy of flow-capture.
- **flow-header** - Display meta information in flow file.
- **flow-split** - Split flow files into smaller files based on size, time, or tags.

Installing Hadoop on Linux

Step 1:

Java (JDK):

Hadoop requires working Java 1.5, we will install Java 7. You can install either Oracle (Sun) Java JDK or Open JDK

Please check if your system has Java running on your system by typing **java -version** in the terminal.

Open Terminal and type these commands and execute one by one

```
sudo add-apt-repository ppa:webupd8team/java
```

```
sudo apt-get update
```

```
sudo apt-get install oracle-java7-installer
```

Java is installed in /usr/lib/jvm/jdk1.7.0_40, where jdk1.7.0_40 varies according to the version you are installing.

Step 2:

Creating a Dedicated Hadoop System user:

We are adding a group hadoop and creating an account/user named hduser in that group.

You can give any name for the user (instead of hduser).

Running these commands will create a new user account (hduser) in group Hadoop.

```
sudo addgroup hadoop
```

```
sudo adduser --ingroup hadoop hduser
```

Step 3:

SSH: Secure SHell

Hadoop requires SSH access to manage its nodes, i.e. remote machines plus your local machine if you want to use Hadoop on. For single-node setup of Hadoop, we therefore need to configure SSH access to localhost (your local machine) for the hduser (Hadoop user in this case) user we created.

Installing ssh server if not present, switch to the hadoop user, generate an SSH key for this user, and enable SSH access to your local machine with this newly created key for this user.

```
sudo addgroup hadoop sudo apt-get install openssh-server
su – hduser
ssh-keygen -t rsa -P ""  // and press enter when asked to save in a file.
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
```

Explanation:

Generating an SSH key for the hduser

This will generate public key without any passphrase. Generally, it’s not recommended to have phrase-less key but here you can’t be entering phrase every time Hadoop communicates with each other.

Step 4:

Disable IPV6 for Hadoop

Hadoop uses 0.0.0.0 for various networking-related configuration, if IPV6 is enabled then Hadoop binds to the ipv6. So it’s better to disable ipv6. You can check whether ipv6 is enabled on your machine by typing the following command in the terminal.

```
cat /proc/sys/net/ipv6/conf/all/disable_ipv6
```

If it return 1, then it’s disabled otherwise it’s enabled.

Disabling ipv6

Open sysctl.conf

```
sudo gedit /etc/sysctl.conf
```

Open sysctl.conf and add these lines at the end of this file.

```
# IPv6
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

Open sysctl.conf and add these lines at the end of this file Reload the configuration and now check if ipv6 is disabled.If returned value is 1, ipv6 is disabled and you are good to go.

```
sudo sysctl –p
```

To disable IPv6 only for Hadoop, add the following line to /home/hduser/hadoop/conf/hadoop-env.sh, after you install Hadoop.

```
export HADOOP_OPTS=-Djava.net.preferIPv4Stack=true
```

Step 5:

Installing Hadoop

Download Hadoop from the Apache Download Mirrors and extract the contents of the Hadoop to the location of your choice. For this tutorial purpose we are using /home/hduser.

Here downloaded hadoop tar file is hadoop-1.2.1.tar.gz and we are moving it to /home/hduser/hadoop

```
cd /home/hduser
sudo tar xzf hadoop-1.2.1.tar.gz
sudo mv hadoop-1.2.1 hadoop
sudo chown –R hduser:hadoop hadoop
```

Other Configurations

Updating .bashrc file

```
sudo gedit $HOME/.bashrc
```

Add following lines at the end to set Hadoop environment variable, set JAVA_HOME variable, add hadoop bin directory to PATH variable

```
export HADOOP_HOME=/home/hduser/hadoop
export JAVA_HOME=/usr/lib/jvm/jdk1.7.0_40
#adding hadoop bin directory to PATH
export PATH=$PATH:$HADOOP_HOME/bin
```

The only required environment variable we have to configure for Hadoop in this tutorial is JAVA_HOME.

Open hadoop-env.sh

```
sudo gedit /home/hduser/hadoop/conf/hadoop-env.sh
```

Add the following line at the end of hadoop-env.sh

```
export JAVA_HOME=/usr/lib/jvm/jdk1.7.0_40
```

Step 6:

Changing configurations of Hadoop files.

We will configure the data storage directory network ports etc.

Now we create temporary directory of Hadoop processing, and this is the directory where we Hadoop stores files. We will give full permissions to this folder.

```
sudo mkdir –p /home/hduser/tmp

sudo chown hduser:hadoop /home/hduser/tmp

sudo chmod 750 /home/hduser/tmp
```

Step 7:

Now add the following snippets between <configuration> … </configuration> tags in the respective configuration XML files.

In **conf/core-site.xml** file

```
<property>

<name>hadoop.tmp.dir</name>

<value>/home/hduser/tmp</value>

<description>A base for other temporary directories.</description>

</property>


<property>

<name>fs.default.name</name>

<value>hdfs://localhost:54310</value>

<description>The name of the default file system. A URI whose

scheme and authority determine the FileSystem implementation. The
```

```
uri’s scheme determines the config property (fs.SCHEME.impl) naming
the FileSystem implementation class. The uri’s authority is used to
determine the host, port, etc. for a filesystem.</description>
</property>
```

In **conf/mapred-site.xml** file:

```
<property>
 <name>mapred.job.tracker</name>
 <value>localhost:54311</value>
<description>The host and port that the MapReduce job tracker runs
 at.  If "local", then jobs are run in-process as a single map
 and reduce task.
 </description>
</property>
```

In **conf/hdfs-site.xml** file:

```
 <property>
 <name>dfs.replication</name>
 <value>1</value>
 <description>Default block replication.
 The actual number of replications can be specified when the file is created.
 The default is used if replication is not specified in create time.
 </description>
 </property>
```

Step 8:

Formatting Name Node

If you want to start using the single node cluster, the first thing we need to do is to format the Hadoop file system which is implemented on top of the local file system. You need to do this only in first you are setting up the cluster.

This simply initializes the directory we have set (/home/hduser/tmp in Step 6) using the following command in the terminal.

```
/home/hduser/hadoop/bin/hadoop namenode -format
```

Now for starting the single-node cluster

```
/home/hduser/hadoop/bin/start-all.sh
```

This will start Namenode, Datanode, Jobtracker and a Tasktracker on single machine (localhost)

You check whether all the expected Hadoop processes are running or not through jps (part of Java), type the following command.

```
jps
```

You can check if Hadoop is listening through all the configured ports or not through netstat

```
sudo netstat -plten | grep java
```

Now to stop your cluster just run the following command in a terminal

```
/home/hduser/hadoop/bin/stop-all.sh
```

References:

- http://hortonworks.com
- http://hadoop.apache.org/