**Run Applications Guide**

Stephen Blystone

Document Version 1.0

Last Updated: Apr 25, 2018

Table of Contents

[**Revision History** 3](#_Toc512462565)

[**Testing Current Pipeline** 3](#_Toc512462566)

[**Stopping Current Pipeline** 6](#_Toc512462567)

# **Revision History**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 1.0 | 4/25/2018 | Initial Version |

# **Testing Current Pipeline**

1. Start ONOS (if not already started) in **ONOS VM**.

onos-buck run onos-local

1. Open a new terminal and launch StreamSets Web GUI (if not already running) in **APP VM**:

~/streamsets-datacollector-3.1.2.0/bin/streamsets dc

Go to <http://streamsetApp:18630/>

**NOTE:** streamsetApp is my VM hostname. You can find your hostname by running “hostname” in a terminal.

Default username/password is “admin” and “admin”

1. Open a new terminal and launch Spark Master and Slave servers:

cd ~/spark-2.3.0-bin-hadoop2.7/

sbin/start-master.sh

sbin/start-slave.sh spark://streamsetApp:7077

**NOTE:** streamsetApp is my VM hostname. You can find your hostname by running “hostname” in a terminal.

1. Open a new terminal and launch zookeeper server:

cd ~/kafka\_2.11-1.0.0/

bin/zookeeper-server-start.sh config/zookeeper.properties

1. Open a new terminal and launch Kafka server:

cd ~/kafka\_2.11-1.0.0/

bin/kafka-server-start.sh config/server.properties

1. Open a new terminal and create a Kafka Topic for NETFLOW:

cd ~/kafka\_2.11-1.0.0/

bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1   
--partitions 1 --topic NETFLOW

1. List Kafka Topics to verify NETFLOW is there

cd ~/kafka\_2.11-1.0.0/

bin/kafka-topics.sh --list --zookeeper localhost:2181

You should see NETFLOW in the output.

1. Run the StreamSets Pipeline. Open StreamSets and open your pipeline. Click 
2. Launch a Kafka Consumer Console to view the NETFLOW messages being received by Kafka.

cd ~/kafka\_2.11-1.0.0/

bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic NETFLOW

1. Open a new terminal and launch Elasticsearch

cd ~/elasticsearch-6.2.3/

bin/elasticsearch

1. Open a new terminal and launch Kibana

cd ~/kibana-6.2.3-linux-x86\_64/

bin/kibana

1. Open a new terminal and launch Logstash

cd ~/logstash-6.2.4

bin/logstash --modules netflow

1. Open a new terminal and launch your Python code for Spark Streaming

cd ~/spark-2.3.0-bin-hadoop2.7/

FOR CAPTURING NORMAL TRAFFIC TO STORE IN ELASTICSEARCH:

bin/spark-submit --packages org.apache.spark:spark-streaming-kafka-0-8\_2.11:2.3.0 python/sparkKafka.py | grep -v INFO | grep -v WARN

FOR READING STORED DATA IN ELASTICSEARCH AND CHECKING FOR ANOMALIES:

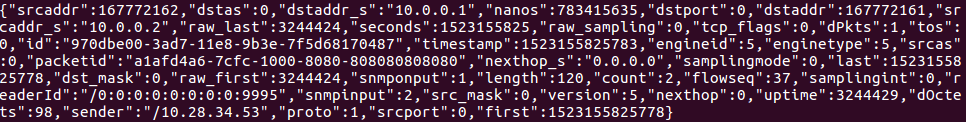
bin/spark-submit --packages org.apache.spark:spark-streaming-kafka-0-8\_2.11:2.3.0 python/sparkMachineLearning.py | grep -v INFO | grep -v WARN

NOTE: These commands will install any missing packages and necessary JAR files for spark and kafka.

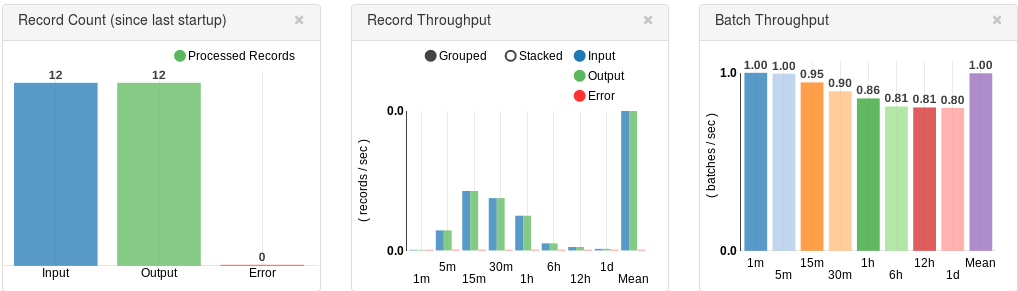
1. Start Mininet (if not already started) in **Mininet VM**.

sudo python setup\_topo.py

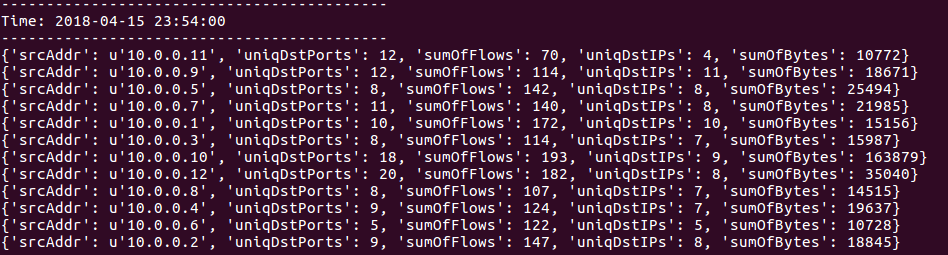
1. You should now see Netflow messages in the kafka-console-consumer window. Example:



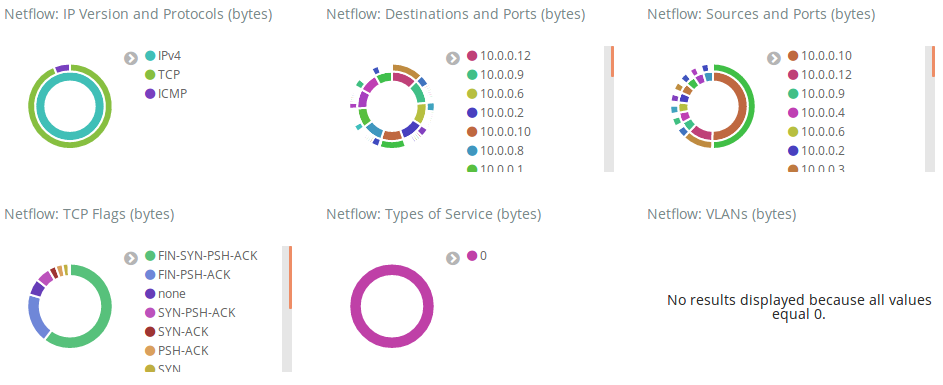
1. In StreamSets web GUI you should also see data showing up in the Summary window. Example:



1. You should see Output in your Spark Streaming terminal window.   
   NOTE: Due to buffering, it may take a few minutes to display the messages on the screen.  
     
   Example:



1. In Kibana Web Gui (<http://localhost:5601/app/kibana>) go to Dashboard.
   1. Click “Netflow: Overview” to see a visualization of the Netflow traffic being received from Mininet.



1. In **Mininet VM**, perform an ICMP Ping Flood:
   1. Type “xterm h11” at the Mininet prompt.
   2. Type “h11 ping h12” to begin a continuous ping.
   3. In the XTerm window that opens, run the following command:

sudo hping3 -V -c 100000 -d 9000 -S -w 64 --flood 10.0.0.2

* 1. This command sends a flood of 100000 pings to h12.

1. In the **App VM** within around 60 seconds (since Spark is configured to capture in 60-second windows), you should see a message in the Spark Streaming Terminal that indicates a REST API call was sent to ONOS to block all traffic from h11.
2. If you check your continuous ping in the **Mininet VM**, you should see that it has stopped receiving responses. You can also run the “pingall” command at the Mininet prompt to see that h11 cannot send packets anywhere, and no other host is able to ping h11.
3. In **Mininet VM**, perform a UDP port scan:
   1. Type “xterm h21” at the Mininet prompt.
   2. Type “h21 ping h22” to begin a continuous ping.
   3. In the XTerm window that opens, run the following command:

python scapyPortScan.py --ip 10.0.0.5

* 1. This Python script will run a port scan on h22 from port 1 to port 65535.

1. In the **App VM** within around 60 seconds (since Spark is configured to capture in 60-second windows), you should see a message in the Spark Streaming Terminal that indicates a REST API call was sent to ONOS to block all traffic from h21.
2. If you check your continuous ping in the **Mininet VM**, you should see that it has stopped receiving responses. You can also run the “pingall” command at the Mininet prompt to see that h11 and h21 cannot send packets anywhere, and no other host is able to ping h11 or h21.

# **Stopping Current Pipeline**

1. **NOTE: It is important you run these in order. Otherwise some items will refuse to shut down.**
2. Ctrl-c to stop the Spark Streaming Python code.
3. Ctrl-c to stop Logstash
4. Ctrl-c to stop Elasticsearch
5. Ctrl-c to stop Kibana
6. Stop the Spark Master and Slave servers

cd ~/spark-2.3.0-bin-hadoop2.7/

sbin/stop-slave.sh

sbin/stop-master.sh

1. Ctrl-c to stop the kafka-console-consumer.
2. Stop the Kafka Server

cd ~/kafka\_2.11-1.0.0/

bin/kafka-server-stop.sh

1. Stop the Zookeeper Server

cd ~/kafka\_2.11-1.0.0/

bin/zookeeper-server-stop.sh

1. Stop the StreamSets Pipeline by clicking  and answering “Yes” to confirm stopping.