Setting Up PCAP

Date of Publish: 2019-3-18

https://docs.hortonworks.com

Contents

Setting up pcap Overview

Because the pcap data source creates an Apache Storm topology that can rapidly ingest raw data directly into HDFS from Apache Kafka, you can store all of your cybersecurity data in its raw form in HDFS and review or query it at a later date.

HCP supports two pcap components:

- The pycapa tool, for low-volume packet capture
- The Fastcapa tool, or high-volume packet capture

  Fastcapa is a probe that performs fast network packet capture by leveraging Linux kernel-bypass and user space networking technology. The probe will bind to a network interface, capture network packets, and send the raw packet data to Kafka. This provides a scalable mechanism for ingesting high-volumes of network packet data into a Hadoop cluster.

  Fastcapa leverages the Data Plane Development Kit (DPDK). DPDK is a set of libraries and drivers to perform fast packet processing in Linux user space.

Set up pycapa

You can use the pycapa tool to capture low-volume data flow.

Before you begin

- Ensure you have installed Python 2.7
- Older distributions, such as CentOS 6, that come with Python 2.6 installed, should install Python 2.7 within a virtual environment and then run Pycapa from within the virtual environment.
- This installation assumes the following environment variables:

```
PYCAPA_HOME=/opt/pycapa
PYTHON27_HOME =/opt/rh/python27/root
```

Procedure

1. Install system dependencies including the core development tools, Python libraries and header files, and Libpcap libraries and header files.

   On CentOS 7+, you can install these requirements using the following command:

   ```
   yum -y install "@Development tools" python-devel libpcap-devel
   ```

   *** In a previous version of these installation instructions, we installed the following packages: epel-release, centos-release-scl, "@Development tools", python 27, python27-scldevel, python27-python-virtualenv libpcap-devel, and libselinux-python. The current installation lists much fewer packages. Are the rest of the packages no longer necessary? ***
2. Install Librdkafka at your chosen $PREFIX:

   ```
   export PREFIX=/usr
   wget https://github.com/edenhill/librdkafka/archive/v0.11.5.tar.gz   -O -
    | tar -xz
   cd librdkafka-0.11.5/
   ./configure --prefix=$PREFIX
   make
   make install
   ```

3. Add Librdkafka to the dynamic library load path.

```
echo "$PREFIX/lib" >> /etc/ld.so.conf.d/pycapa.conf
ldconfig -v
```

4. Install Pycapa.

   This step assumes that you already have the HCP source code installed on the host.

```
cd metron/metron-sensors/pycapa
pip install -r requirements.txt
python setup.py install
```

5. Start the pycapa packet capture producer:

```
cd ${PYCAPA_HOME}/pycapa-venv/bin
pycapa --producer --topic pcap -i $ETH_INTERFACE -k $KAFKA_HOST:6667
```

Start pcap

To start pcap, HCP provides a utility script. This script takes no arguments and is very simple to run.

Procedure

1. Log in to the host on which you are running Metron.
2. If you are running HCP on an Ambari-managed cluster, perform the following steps; otherwise proceed with Step 3:
   a) You can retrieve the appropriate server information from Ambari in **Kafka service** > **Configs** > **Kafka Broker** > **zookeeper.connect**.
   b) On the HDFS host, create /apps/metron/pcap, change its ownership to metron:hadoop, and change its permissions to 775:

```
hdfs dfs -mkdir /apps/metron/pcap
hdfs dfs -chown metron:hadoop /apps/metron/pcap
hdfs dfs -chmod 755 /apps/metron/pcap
```

   c) Create a Metron user's home directory on HDFS and change its ownership to the Metron user:

```
hdfs dfs -mkdir /user/metron
hdfs dfs -chown metron:hadoop /user/metron
hdfs dfs -chmod 755 /user/metron
```

   d) Create a pcap topic in Kafka:
      - Switch to **metron** user:

```
su - metron
```

      - Create a Kafka topic named pcap:

```
/usr/hdp/current/kafka-broker/bin/kafka-topics.sh \
--zookeeper $ZOOKEEPER_HOST:2181 \
--create \
--topic pcap \
--partitions 1 \
--replication-factor 1
```

- List all of the Kafka topics, to ensure that the new pcap topic exists:

```
/usr/hdp/current/kafka-broker/bin/kafka-topics.sh --zookeeper
 $ZOOKEEPER_HOST:2181 --list
```

3. If HCP is installed on an Ambari-managed cluster, use the following command to tart the pcap topology:

```
su - metron $METRON_HOME/bin/start_pcap_topology.sh
```

4. If HCP is installed by CLI, use the following command to start the pcap topology.

```
$METRON_HOME/bin/start_pcap_topology.sh
```

5. Check the Storm topology to ensure that packets are being captured.

   After Storm has captured a sufficient number of packets, you can check to ensure it is creating files on HDFS:

```
hadoop fs -ls /apps/metron/pcap
```

Installing Fastcapa

You can install Fastcapa either automatically or manually. The automated installation is the simplest but it requires CentOS 7.1. If you are not running CentOS 7.1 or would like more visibility into the installation process, you can manually install Fastcapa.

Requirements for Installing Fastcapa

The Fastcapa probe requires specific system requirements.

The following system requirements must be met to run the Fastcapa probe:

- Linux kernel 2.6.34 or later
- A DPDK supported ethernet device; NIC
- Ports on the Ethernet device that can be dedicated for exclusive use by Fastcapa

Install Fastcapa Automatically

Installing Fastcapa has several steps and involves building Data Plan Development Kit (DPDK), loading specific kernel modules, enabling huge page memory, and binding compatible network interface cards.

The best documentation for installing the Fastcapa probe is code that actually does this for you. You can use an Ansible role that performs the entire installation: metron-deployment/roles/fastcapa.

Install Fastcapa Manually

As an alternative to automatically installing Fastcapa, you can install the probe manually.

Before you begin

The following manual installation steps assume that they are executed on CentOS 7.1. Some minor differences might result if you use a different Linux distribution.

Enable Transparent Huge Pages

The Fastcapa probe performs its own memory management by leveraging Transparent Huge Pages (THP). In Linux, you can use the Transparent Huge Pages to enable either dynamically or automatically upon startup. Hortonworks recommends that these be allocated on boot to increase the chance that a larger, physically contiguous chunk of memory allocated.

Before you begin

For better performance, allocate 1 GB THPs if supported by your CPU.

Procedure

1. Ensure that your CPU supports 1 GB THPs. A CPU flag pdpe1gb indicates whether or not the CPU supports 1 GB THPs.

```
grep --color=always pdpe1gb /proc/cpuinfo | uniq
```

2. Edit /etc/default/grub to add the following book parameters at the line starting with GRUB_CMDLINE_LINUX:

```
GRUB_CMDLINE_LINUX=... default_hugepagesz=1G hugepagesz=1G hugepages=16
```

3. Rebuild the Grub configuration then reboot. The location of the Grub configuration file will differ across Linux distributions.

```
cp /etc/grub2-efi.cfg /etc/grub2-efi.cfg.orig
/sbin/grub2-mkconfig -o /etc/grub2-efi.cfg
```

4. After the host reboots, ensure that the THPs were successfully allocated:

```
$ grep HugePage /proc/meminfo
AnonHugePages:    933888 kB
HugePages_Total:      16
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
```

   The total number of huge pages should be distributed fairly evenly across each non-uniform memory access (NUMA) node. In the following example, a total of 16 requested THPs are distributed as 8 to each of 2 NUMA nodes:

```
$ cat /sys/devices/system/node/node*/hugepages/hugepages-1048576kB/
nr_hugepages
8
8
```

5. After the THPs are reserved, mount them to make them available to the probe:

```
cp /etc/fstab /etc/fstab.orig
mkdir -p /mnt/huge_1GB
echo "nodev /mnt/huge_1GB hugetlbfs pagesize=1GB 0 0" >> /etc/fstab
mount -fav
```

Install DPDK

After you enable transparent huge pages, you must install a data plane development kit (DPDK) and the associated network interface controller (NIC).

Procedure

1. Install the required dependencies:

```
yum -y install "@Development tools"
yum -y install pciutils net-tools glib2 glib2-devel git
yum -y install kernel kernel-devel kernel-headers
```

2. Specify where you want DPDK installed:

```
export DPDK_HOME=/usr/local/dpdk/
```

3. Download, build, and install DPDK:

```
wget http://fast.dpdk.org/rel/dpdk-16.11.1.tar.xz -O - | tar -xJ
cd dpdk-stable-16.11.1/
make config install T=x86_64-native-linuxapp-gcc DESTDIR=$DPDK_HOME
```

4. Specify the PCI address of the Ethernet device that you want to use to capture network packets:

```
$ lspci | grep "VIC Ethernet"
09:00.0 Ethernet controller: Cisco Systems Inc VIC Ethernet NIC (rev a2)
0a:00.0 Ethernet controller: Cisco Systems Inc VIC Ethernet NIC (rev a2)
```

5. Bind the device, using a device name and PCI address appropriate to your environment:

```
ifdown enp9s0f0
modprobe uio_pci_generic
$DPDK_HOME/sbin/dpdk-devbind --bind=uio_pci_generic "09:00.0"
```

6. Ensure that the device was bound:

```
$ dpdk-devbind --status
Network devices using DPDK-compatible driver
============================================
0000:09:00.0 'VIC Ethernet NIC' drv=uio_pci_generic unused=enic
Network devices using kernel driver
===================================
0000:01:00.0 'I350 Gigabit Network Connection' if=eno1 drv=igb
 unused=uio_pci_generic
```

Install Librdkafka

Install the Apache Kafka C/C++ client library (librdkafka) to assist in configuring Fastcapa.

Before you begin

The Fastcapa probe has been tested with Librdkafka 0.9.4.

Procedure

1. Specify an installation path for librdkafka:

```
export RDK_PREFIX=/usr/local
```

In the following example, the libs will actually be installed at /usr/local/lib; note that lib is appended to the prefix.

2. Download, build, and install librdkafka:

```
wget https://github.com/edenhill/librdkafka/archive/v0.9.4.tar.gz  -O - |
 tar -xz
cd librdkafka-0.9.4/
./configure --prefix=$RDK_PREFIX
make
make install
```

3. Ensure that the installation s on the search path for the runtime shared library loader:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$RDK_PREFIX/lib
```

Install Fastcapa

After you enabled transparent huge pages, and installed both DPDK and librdkafka, you can install Fastcapa.

Procedure

1. Set the required environment variables:

```
export RTE_SDK=$DPDK_HOME/share/dpdk/
export RTE_TARGET=x86_64-native-linuxapp-gcc
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$RDK_HOME
```

2. Build Fastcapa:

```
cd metron/metron-sensors/fastcapa
make
```

The resulting binary is placed at build/app/fastcapa.

Using Fastcapa

You can use the Fastcapa tool to capture high-volume data flow.

Procedure

1. Create a configuration file that, at a minimum, specifies your Kafka broker:

```
[kafka-global]
metadata.broker.list = kafka-broker1:9092
```

You can view the example configuration file conf/fastcapa.conf to learn other useful parameters.

2. If the capture device is not bound, bind it:

```
ifdown enp9s0f0
modprobe uio_pci_generic
$DPDK_HOME/sbin/dpdk-devbind --bind=uio_pci_generic "09:00.0"
```

3. Run Fastcapa:

```
fastcapa -c 0x03 --huge-dir /mnt/huge_1GB -- -p 0x01 -t pcap -c /etc/
fastcapa.conf
```

4. Terminate Fastcapa and clear the queue by usingSIGINT or by typing CTRL-C.

The probe will cleanly shut down all of the workers and allow the backlog of packets to drain.

To terminate the process without clearing the queue, use SIGKILL or enterkillall -9 fastcapa.

Fastcapa Environmental Abstraction Layer Parameters

The most commonly used DPDK Environmental Abstraction Layer (EAL) parameter involves specifying which logical CPU cores should be used for processing.

This can be specified in any of the following ways:

```
                          -c COREMASK         Hexadecimal bitmask of
cores to run on
 -l CORELIST         List of cores to run on
                     The argument format is <c1>[-c2][,c3[-c4],...]
                     where c1, c2, etc are core indexes between 0 and 128
 --lcores COREMAP    Map lcore set to physical cpu set
                     The argument format is
                           '<lcores[@cpus]>[<,lcores[@cpus]>...]'
                     lcores and cpus list are grouped by '(' and ')'
                     Within the group, '-' is used for range separator,
```

```
                        ',' is used for single number separator.
                        '( )' can be omitted for single element group,
                        '@' can be omitted if cpus and lcores have the same
value
```

For more information bout EAL parameters, run the following command:

```
fastcapa -h
```

Fastcapa-Core Parameters

The core parameters are command-line parameters that define how Fastcapa interacts with DPDK.

These parameters are separated on the command line by a double dash (--).

Name	Command	Description	Default
Port Mask	-p PORT_MASK	A bit mask identifying which ports to bind.	0x01
Burst Size	-b BURST_SIZE	Maximum number of packets to receive at one time.	32
Receive Descriptors	-r NB_RX_DESC	The number of descriptors for each receive queue. Limited by the Ethernet device in use.	1024
Transmission Ring Size	-x TX_RING_SIZE	The size of each transmission ring. This must be a power of 2.	2048
Number Receive Queues	-q NB_RX_QUEUE	Number of receive queues to use for each port. Limited by the Ethernet device in use.	2
Kafka Topic	-t KAFKA_TOPIC	The name of the Kafka topic.	pcap
Configuration File	-c KAFKA_CONF	Path to a file containing configuration values.	
Stats	-s KAFKA_STATS	Appends performance metrics in the form of JSON strings to the specified file.	

For more information about Fastcapa-specific parameters, run the following command:.

```
fastcapa -- -h
```

Fastcapa-Kafka Configuration File

The Kafka configuration file defines how Fastcapa interacts with librdkafka.

You specify the path to the configuration file with the -c command-line argument. The file can contain any global or topic-specific, producer-focused configuration values accepted by Librdkafka.

The configuration file is a .ini-like Glib configuration file. Place the global configuration values under a [kafka-global] header and place topic-specific values under [kafka-topic].

A minimally viable configuration file only needs to include the Kafka broker to connect to:

```
```

```
                                [kafka-global]
metadata.broker.list = kafka-broker1:9092, kafka-broker2:9092
```

The configuration parameters that are important for either basic functioning or performance tuning of Fastcapa include the following.

Name	Description	Default
metadata.broker.list	Initial list of brokers as a CSV list of broker host or host:port	NA
client.id	Client identifier.	
queue.buffering.max.messages	Maximum number of messages allowed on the producer queue	100000
queue.buffering.max.ms	Maximum time, in milliseconds, for buffering data on the producer queue	1000
message.copy.max.bytes	Maximum size for the message to be copied to buffer. Messages larger than this are passed by reference (zero-copy) at the expense of larger iovecs.	65535
batch.num.messages	Maximum number of messages batched in one MessageSet	10000
statistics.interval.ms	How often statistics are emitted; 0 = never	0
compression.codec	Compression codec to use for compressing message sets: none, gzip, snappy, lz4	none

Local global configuration values under the [kafka-global] header.

Locate topic configuration values under the [kafka-topic] header.

	Description	Default
compression.codec	Compression codec to use for compressing message sets: none, gzip, snappy, lz4	none
request.required.acks	How many acknowledgements the leader broker must receive from ISR brokers before responding to the request; { 0 = no ack, 1 = leader ack, -1 = all ISRs }	1
message.timeout.ms	Local message timeout. This value is only enforced locally and limits the time a produced message waits for successful delivery. A value of 0 represents infinity.	300000
queue.buffering.max.kbytes	Maximum total message size sum allowed on the producer queue	none

Fastcapa Counters Output

When running the Fastcapa probe, some basic counters are output to stdout. During normal operation these values are much larger.

```
                              ------ in ------  --- queued --- ----- out
 ----- ---- drops ----
[nic]                8              -               -               -
[rx]                 8              0               8               0
[tx]                 8              0               8               0
[kaf]                8              1               7               0
```

```
```

- [nic] + in : The Ethernet device is reporting that it has seen eight packets.
- [rx] + in : The receive workers have consumed eight packets from the device.
- [rx] + out : The receive workers have enqueued 8 packets onto the transmission rings.
- [rx] + drops : If the transmission rings become full, it prevents receive workers from enqueuing additional packets. The excess packets are dropped. This value never decreases.
- [tx] + in : The transmission workers consumed 8 packets.
- [tx] + out : The transmission workers packaged 8 packets into Kafka messages.
- [tx] + drops : If the Kafka client library accepted fewer packets than expected. This value might change as additional packets are acknowledged by the Kafka client library
- [kaf] + in : The Kafka client library received 8 packets.
- [kaf] + out : A total of 7 packets successfully reached Kafka.
- [kaf] + queued : There is 1 packet within the rdkafka queue

Use Fastcapa in a Kerberized Environment

You can use the Fastcapa probe in a Kerberized environment.

Before you begin

The following task assumes that you have configured the following values. If necessary, change these values to match your environment.

The Kafka broker is at kafka1:6667.

ZooKeeper is at zookeeper1:2181.

The Kafka security protocol is SASL_PLAINTEXT.

The keytab used is located at /etc/security/keytabs/metron.headless.keytab.

The service principal is metron@EXAMPLE.COM.

Procedure

1. Build Librdkafka with SASL support (--enable-sasl):

```
wget https://github.com/edenhill/librdkafka/archive/v0.9.4.tar.gz  -O - |
 tar -xz
cd librdkafka-0.9.4/
./configure --prefix=$RDK_PREFIX --enable-sasl
make
make install
```

2. Verify that Librdkafka supports SASL:

```
$ examples/rdkafka_example -X builtin.features
builtin.features = gzip,snappy,ssl,sasl,regex
```

3. If Librdkafka does not support SASL, install libsasl or libsasl2. Use the following command to install libsasl on your CentOS environment:

```
yum install -y cyrus-sasl cyrus-sasl-devel cyrus-sasl-gssapi
```

4. Grant access to your Kafka topic (in this example, named pcap):

```
$KAFKA_HOME/bin/kafka-acls.sh --authorizer
 kafka.security.auth.SimpleAclAuthorizer \
  --authorizer-properties zookeeper.connect=zookeeper1:2181 \
```

```
  --add --allow-principal User:metron --topic pcap
```

5. Obtain a Kerberos ticket:

```
kinit -kt /etc/security/keytabs/metron.headless.keytab metron@EXAMPLE.COM
```

6. Add the following additional configuration values to your Fastcapa configuration file:

```
security.protocol = SASL_PLAINTEXT
sasl.kerberos.keytab = /etc/security/keytabs/metron.headless.keytab
sasl.kerberos.principal = metron@EXAMPLE.COM
```

7. Run Fastcapa