

Laboratory Exercise: Session Data

Goals

1. To understand its basic components and architecture of a session based network monitoring system.
2. To be able to install and configure one such system, based on SiLK, and to understand how these components operate in that context.
3. To be able to use the data collected to perform common network auditing tasks.
4. To understand the tradeoffs between full packet and session data capture with respect to network security monitoring.

Prerequisites

Before you can begin with this particularly chapter, you will need to set up a virtual network environment with several different virtual machines with which a number of different software packages will be installed.

First thing you need to do is ensure that everything is updated on your SecurityOnion system¹. To do this run the following command:

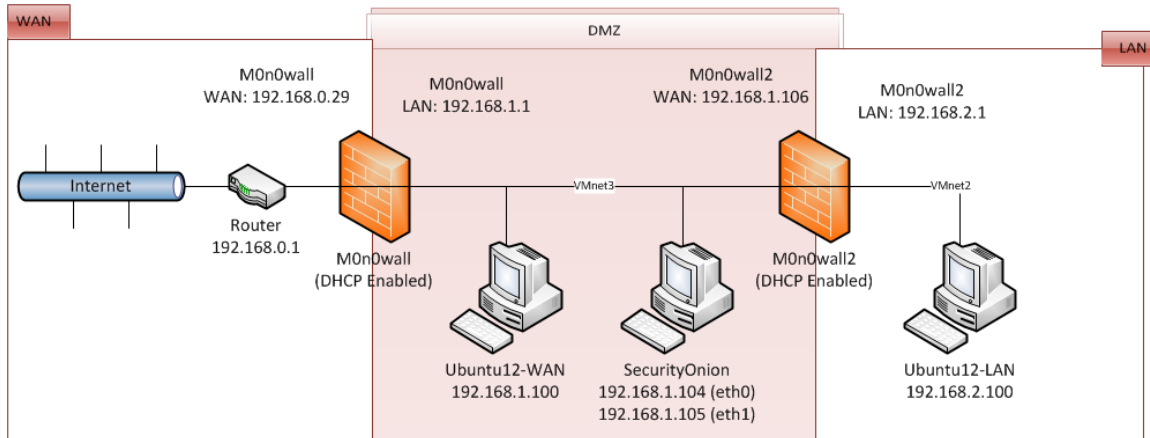
```
$ sudo soup
```

Do not worry if it takes a long time to update, this is to be expected and should not cause concern. Once it has been updated close down the VM.

You can now go about setting up the network environment that you will be using for this chapter. See below for a diagram of the machines and setup required:

¹<https://github.com/Security-Onion-Solutions/security-onion/wiki/Upgrade>

Lab 3 – Session Data by Bilal Khan









We will refer to vbox0 as the DMZ, and vbox1 as the LAN. We will refer to the bridged network as the WAN, note that this implies your native host is in the WAN.

Let's take it one step at a time. Clone the M0n0wall VM that you used in the previous exercise and call it M0n0wall2. Change the settings of the M0n0wall VM so that one network adapter is set to bridge mode and the other is set to host only (vmbx0). Go ahead and boot up the M0n0wall VM. Check that the interfaces are set up correctly by swapping them so that the LAN is em1 and the WAN is em0. Change the LAN IP address on the M0n0wall machine to 192.168.1.1 **[EXPAND ON THIS POINT]**.

The next step is to change your Ubuntu machine and the SecurityOnion so they are set to be on the vmbox0 interface. Load up the Ubuntu machine and browse to the IP address of the new M0n0wall LAN (192.168.1.1). Set up a new DHCP reservation with the Ubuntu MAC address to 192.168.1.100 and save the settings.

Go to VirtualBox and ensure that your SecurityOnion machine has two network adapters that are both set to be host-only and connected to the (vmbx0) interface too. Once you have done this, load up SecurityOnion. Run the ifconfig command and write down the two MAC addresses. Now go back to your Ubuntu machine and, using the M0n0wall web interface, add two more DHCP reservations for the SecurityOnion Ethernet adapters. A diagram below shows you what it should now look like. Go ahead and reboot both the Ubuntu and SecurityOnion VM's. Once they are loaded back up you should see the new IP addresses.

Lab 3 – Session Data by Bilal Khan

Reservations			
MAC address	IP address	Description	
08:00:27:4a:50:72	192.168.1.100	Ubuntu	 
08:00:27:10:95:5e	192.168.1.104	SecOnion-0	 
08:00:27:9a:ea:0d	192.168.1.105	SecOnion-1	 

Load up the M0n0wall2 machine and change its LAN IP address to 192.168.2.1. Determine which interface needs to be on the LAN and which needs to be on the WAN.

Change the Ubuntu-LAN machine so it is running on the vmbox1 interface then go ahead and boot that machine up too. Log into the M0n0wall2 machine and set its WAN IP address to be set by DHCP server (the M0n0wall machine will do set it). Restart both M0n0wall 2 and the Ubuntu-LAN machine in that order. At this point everything should be booted up and running. You have now set up the environment that you will need for the rest of this chapter.

To understand its basic components and architecture of a session based network monitoring system.

Approach

-

Execution

The

Cautions (Optional)

-

To be able to install and configure one such system, based on SiLK, and to understand how these components operate in that context.

Approach

-

Execution

What you need to do now is download and install all of the tools that are necessary for SiLK to run. It is important that you finish each step in order before you move onto the next step otherwise it can cause problems later on with the installations.

SecurityOnion Setup

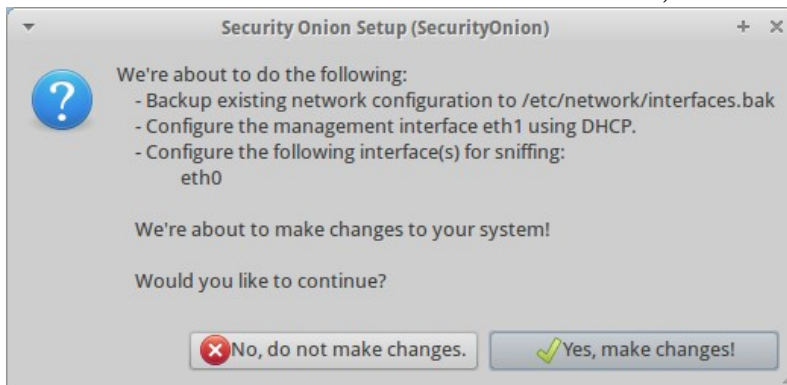
Before you can process with any further steps you need to run the SecurityOnion setup on your machine first. On the desktop there is an icon called Setup. Double click that to begin.

The first section of the setup is to configure the network interfaces, so when prompted, select “Yes, *configure /etc/network/interfaces!*”. Your management interface should be set to eth1.

You should use DHCP addressing (remember that you have previously setup a reserved IP address on the M0n0wall so it will not change).

It will then ask if you would like to configure the monitor interfaces, once again select No and check the eth0 box.

You will then be presented with a summary screen like that shown below. Check that it matches and then click “Yes, make changes!”.

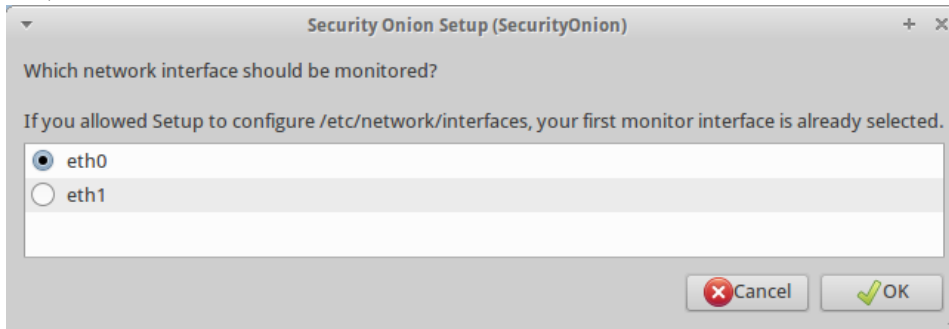


Once this is completed it will ask you to reboot before continuing to the second phase of the Setup. Go ahead and reboot the machine. After it has rebooted you will need to run the setup again so you can continue with the second phase.

Lab 3 – Session Data by Bilal Khan

It will ask you to confirm that you want to skip the network configuration setup because it appears that it has already been completed. You should skip the step and then select the “*Quick Setup*” on the next page.

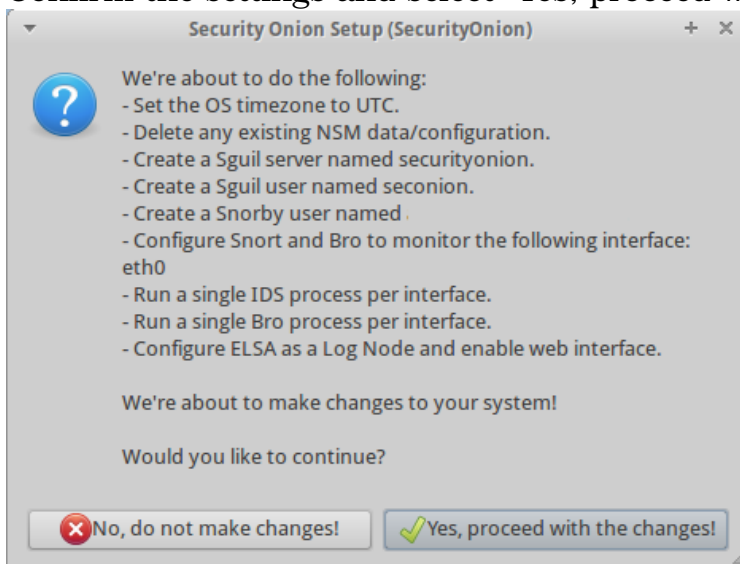
The next part is to determine which interface you want to monitor traffic on, select eth0 like shown below:



It will then ask you to provide an email address for Snorby. Whilst we will not be using it right now, it may still be useful in future so go ahead and input your email address here.

It will then ask you to create your Sguil username then continue to make your Sguil password.

The next step is it will ask you whether you want to enable ELSA, select yes. Finally it will again give you a summary, like that shown below. Confirm the settings and select “Yes, proceed with the changes!”.



You are now ready to proceed with the next stages.

Downloading

On your SecurityOnion machine, create a new directory where you are going to download all the packages. In this instance it will be referred to as 'Installs'. Change directory to there and then issue the following commands:

```
$ wget http://tools.netsa.cert.org/releases/silk-3.7.2.tar.gz
$ wget http://tools.netsa.cert.org/releases/yaf-2.4.0.tar.gz
$ wget http://tools.netsa.cert.org/releases/libfixbuf-1.3.0.tar.gz
```

Once you have issued all three commands, your directory should look like that shown below:

```
seconion@SecurityOnion:~/Documents/SessionData/Installs$ ls -lt
total 6912
-rw-rw-r-- 1 seconion seconion 5035334 Aug 15  2013 silk-3.7.2.tar.gz
-rw-rw-r-- 1 seconion seconion 1388811 May  3  2013 yaf-2.4.0.tar.gz
-rw-rw-r-- 1 seconion seconion  645616 Mar  8  2013 libfixbuf-1.3.0.tar.gz
seconion@SecurityOnion:~/Documents/SessionData/Installs$ pwd
/home/seconion/Documents/SessionData/Installs
seconion@SecurityOnion:~/Documents/SessionData/Installs$
```

Installation

The analysis of flow data requires a flow generator and a collector. So, before you can begin collecting and analyzing session data with SiLK you need to ensure that you have data to collect. In this case, you will be installing the YAF flow generation utility. YAF generates IPFIX flow data, which is quite flexible. Collection will be handled by the rflowpack component of SiLK, and analysis will be provided through the SiLK rtool suite.

To install these tools, you will need a couple of prerequisites. You can install these in one fell swoop by running this command:

```
$ sudo apt-get install glib2.0 libglib2.0-dev libpcap-dev g++ python-dev
```

As part of the installation process it will inform you that additional disk space will be used and request you confirm it. Select 'Y'. It will then begin the installation of the necessary packages.

Once this process is complete there will be a number of packages that are no longer required and can be removed from the system. To do this, run the following command:

Lab 3 – Session Data by Bilal Khan

\$ sudo apt-get autoremove

```
seconion@SecurityOnion:~/Documents/SessionData/Installs$ sudo apt-get autoremove
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  gir1.2-gstreamer-0.10 gir1.2-json-1.0 gir1.2-timezonemap-1.0
  gir1.2-xkl-1.0 libjson-glib-1.0-0 libtimezonemap1
0 upgraded, 0 newly installed, 6 to remove and 0 not upgraded.
After this operation, 2,104 kB disk space will be freed.
Do you want to continue [Y/n]? Y
(Reading database ... 193818 files and directories currently installed.)
Removing gir1.2-gstreamer-0.10 ...
Removing gir1.2-timezonemap-1.0 ...
Removing gir1.2-json-1.0 ...
Removing gir1.2-xkl-1.0 ...
Removing libtimezonemap1 ...
Removing libjson-glib-1.0-0 ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
seconion@SecurityOnion:~/Documents/SessionData/Installs$
```

With this done, you can install fixbuf using these steps:

1. Extract the archive and go to the newly extracted folder

```
$ tar xvzf libfixbuf-1.3.0.tar.gz
$ cd libfixbuf-1.3.0/
```

2. Configure, make, and install the package

```
$ ./configure
$ make
$ sudo make install
```

Before you can install YAF you need to remember to go back into the previous Installs directory. To do so issue:

```
$ cd ../
```

Now you can install YAF with these steps:

1. Extract the archive and go to the newly extracted folder

```
$ tar xvzf yaf-2.4.0.tar.gz
$ cd yaf-2.4.0/
```

2. Export the PKG configuration path

```
$ export PKG_CONFIG_PATH=/usr/local/lib/pkgconfig
```

3. Configure with applabel enabled

Lab 3 – Session Data by Bilal Khan

```
$ ./configure --enable-applabel
```

4. Make and install the package

```
$ make
```

```
$ sudo make install
```

If you try to run YAF right now, you'll notice an error. You need to continue the installation process before it will run properly. This process continues by installing SiLK with these steps:

1. Change directory to the previous Installs directory

```
$ cd ../
```

2. Extract the archive and go to the newly extracted folder

```
$ tar xvzf silk-3.7.2.tar.gz
```

```
$ cd silk-3.7.2/
```

3. Configure with a specified fixbuf path and python enabled

```
$ ./configure --with-libfixbuf=/usr/local/lib/pkgconfig/ --with-python
```

4. Make and install the package

```
$ make
```

```
$ sudo make install
```

You will know that SiLK has installed properly if you see a SiLK summary commands, like that shown below:

Lab 3 – Session Data by Bilal Khan

```
config.status: executing silk_summary commands

* Configured package:      SiLK 3.7.2
* Host type:               x86_64-unknown-linux-gnu
* Source files ($top_srcdir): .
* Install directory:      /usr/local
* Root of packed data tree: /data
* Packing logic:           via run-time plugin
* Timezone support:        UTC
* Default compression method: SK_COMPMETHOD_NONE
* IPv6 network connections: YES
* IPv6 flow record support: NO
* IPFIX collection support: YES (-pthread -L/usr/local/lib -lfixbuf -lgthread-2.0 -lrt -lglib-2.0)
* NetFlow9 collection support: YES
* ASA 0-packet work-around: NO
* Transport encryption support: NO (gnutls not found)
* IPA support:             NO
* LIBPCAP support:         YES (-lpcap)
* C-ARES support:         NO
* ADNS support:           NO
* Python support:         YES (-Wl,-Bsymbolic-functions -Wl,-z,relro -Xlinker -export-dynamic -Wl,-O1 -Wl,-Bsymbolic-f
unctions -lssl -lcrypto -lssl -lcrypto -L/usr/lib -lz -ldl -lutil -lm -Wl,-Bsymbolic-functions -Wl,-z,relro -L/usr/lib -lpython2
.7 -pthread)
* Python package destination: /usr/lib/python2.7/dist-packages
* Build analysis tools:    YES
* Build packing tools:     YES
* Compiler (CC):           gcc
* Compiler flags (CFLAGS): -I$(srcdir) -I$(top_builddir)/src/include -I$(top_srcdir)/src/include -DNDEBUG -D_ALL_SOURCE
-D_GNU_SOURCE=1 -O3 -fno-strict-aliasing -Wall -W -Wmissing-prototypes -Wformat=2 -Wdeclaration-after-statement -Wpointer-ari
th
* Linker flags (LDFLAGS):  -lz -ldl -lm
* Libraries (LIBS):        -lz -ldl -lm

seconion@SecurityOnion:~/Documents/SessionData/Installs/silk-3.7.2$
```

With everything installed, you need to make sure that all of the libraries we need are linked properly so that the `LD_LIBRARY_PATH` variable doesn't have to be exported each time you use SiLK. This can be done by creating a file named `silk.conf` in the `/etc/ld.so.conf.d/` directory with the following contents:

```
/usr/local/lib
/usr/local/lib/silk
```

To apply this change, run:
`$ sudo ldconfig`

Configuring

Now you have to configure SiLK to use `rwflowpack` to collect the flow data you generate. You need three files to make this happen: `silk.conf`, `sensors.conf`, and `rwflowpack.conf`.

NOTE: In what follows, whenever you see `<$SENSOR-$INTERFACE>` it stands for **SecurityOnion-eth0**.

Silk.conf

You will start by creating the `silk.conf` site configuration file. This file controls how SiLK parses data, and contains a list of sensors. It can be found in the previously unzipped SiLK installation tarball at `silk-3.7.2/site/twoway/silk.conf`. You need to copy it to a directory that Security Onion uses to store several other configuration files:

```
$ sudo cp silk.conf /etc/nsm/<$SENSOR-$INTERFACE>/
```

Sensors.conf

The sensor configuration file `sensors.conf` is used to define the sensors that will be generating session data, and their characteristics. This file should be created at `/etc/nsm/<$SENSOR-$INTERFACE>/sensors.conf`. For this example, your `sensors.conf` will look like this:

```
probe S0 ipfix
  listen-on-port 18001
  protocol tcp
  listen-as-host 192.168.1.104
end probe
group my-network
  ipblocks 192.168.1.0/24
  ipblocks 192.168.2.0/24
end group
sensor S0
  ipfix-probes S0
  internal-ipblocks @my-network
  external-ipblocks remainder
end sensor
```

This `sensors.conf` has three different sections: `probe`, `group`, and `sensor`.

The `probe` section tells SiLK where to expect to receive data from for the identified sensor. Here, you've identified sensor `S0`, and told SiLK to expect to receive `ipfix` data from this sensor via the TCP protocol over port 18001. You've also defined the IP address of the sensor as the IPv4 address of the collector, 192.168.1.104.

The `group` section allows you to create a variable containing IP Blocks. Because of the way SiLK bins flow data, it is important to define internal and external network ranges on a per sensor basis so that your queries that are based upon flow direction (inbound, outbound, inbound web traffic, outbound web traffic, etc.) are accurate. In your configuration, you've defined a group called `my-network` that has two `ipblocks`, 192.168.1.0/24 and 192.168.2.0/24.

The last section is the `sensor` section, which you use to define the characteristics of the `S0` sensor. Here you have specified that the sensor

Lab 3 – Session Data by Bilal Khan

will be generating IPFIX data, and that my-network group defines the internal IP ranges for the sensor, with the remainder being considered external.

When it is all completed, your file should look like the below in the correct location:

A terminal window titled "Terminal - seconion@SecurityOnion: /etc/nsm/SecurityOnion-eth0" showing the configuration of the sensors.conf file. The user runs 'pwd' to confirm the directory, then 'more sensors.conf' to view the file's contents. The configuration includes a probe for S0 using IPFIX on port 18001, a group named 'my-network' with two IP blocks (192.168.1.0/24 and 192.168.2.0/24), and a sensor S0 that uses IPFIX probes and the defined IP blocks for internal and external traffic.

```
seconion@SecurityOnion:/etc/nsm/SecurityOnion-eth0$ pwd
/etc/nsm/SecurityOnion-eth0
seconion@SecurityOnion:/etc/nsm/SecurityOnion-eth0$ more sensors.conf
probe S0 ipfix
    listen-on-port 18001
    protocol tcp
    listen-as-host 192.168.1.104
end probe
group my-network
    ipblocks 192.168.1.0/24
    ipblocks 192.168.2.0/24
end group
sensor S0
    ipfix-probes S0
    internal-ipblocks @my-network
    external-ipblocks remainder
end sensor
seconion@SecurityOnion:/etc/nsm/SecurityOnion-eth0$
```

Rwflowpack.conf

The last configuration step is to modify rwflowpack.conf, which is the configuration file for the rwflowpack process that listens for and collects flow records.

This file can be found at /usr/local/share/silk/etc/rwflowpack.conf.

First, you need to copy this file to /etc/nsm/<\$SENSOR-\$INTERFACE>/
\$ sudo cp /usr/local/share/silk/etc/rwflowpack.conf /etc/nsm/<\$SENSOR-\$INTERFACE>/

Now you need to change seven values in the newly copied file:

ENABLED=yes

This will enable rwflowpack

statedirectory=/nsm/sensor_data/<\$SENSOR-\$INTERFACE>/silk

Lab 3 – Session Data by Bilal Khan

A convenience variable used for setting the location of other various SiLK files and folders

`CREATE_DIRECTORIES=yes`

This will allow for the creation of specified data subdirectories

`SENSOR_CONFIG=/etc/nsm/<$SENSOR-$INTERFACE>/sensors.conf`

The path to the sensor configuration file

`DATA_ROOTDIR=/nsm/sensor_data/<$SENSOR-$INTERFACE>/silk/`

The base directory for SiLK data storage

`SITE_CONFIG=/etc/nsm/<$SENSOR-$INTERFACE>/silk.conf`

The path to the site configuration file

`LOG_TYPE=legacy`

Sets the logging format to legacy

`LOG_DIR=/var/log/`

The path for log storage

Finally, you need to copy `rwflowpack` startup script into `init.d` so that you can start it like a normal service. This command will do that:

```
$ sudo cp /usr/local/share/silk/etc/init.d/rwflowpack /etc/init.d
```

Once you've copied this file, you need to change one path in it. Open the file, and replace the code where `SCRIPT_CONFIG` is set to read:

```
SCRIPT_CONFIG="/etc/nsm//<$SENSOR-$INTERFACE>/rwflowpack.conf"
```

Starting Everything Up

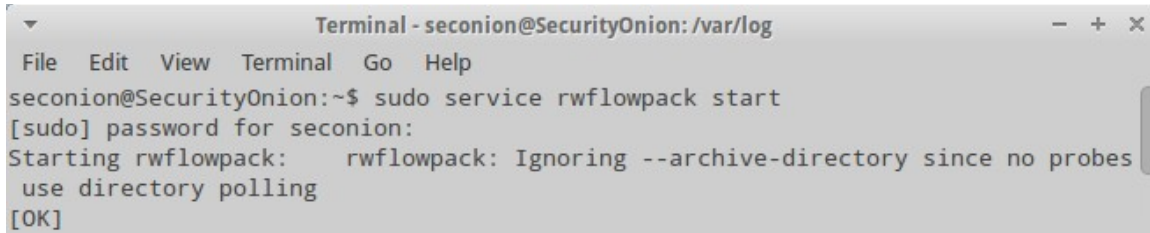
Now that everything is configured, you should be able to start `rwflowpack` and `YAF` and begin collecting data.

First, you can start `rwflowpack` by simply typing the following:

```
$ sudo service rwflowpack start
```

If everything went well, you should see a success message OK like that shown on the following page.

Lab 3 – Session Data by Bilal Khan

A terminal window titled "Terminal - seconion@SecurityOnion: /var/log" with a menu bar (File, Edit, View, Terminal, Go, Help). The terminal shows the command "sudo service rwflowpack start" being executed. It prompts for a password, then displays the output: "Starting rwflowpack: rwflowpack: Ignoring --archive-directory since no probes use directory polling" followed by "[OK]".

```
Terminal - seconion@SecurityOnion: /var/log
File Edit View Terminal Go Help
seconion@SecurityOnion:~$ sudo service rwflowpack start
[sudo] password for seconion:
Starting rwflowpack: rwflowpack: Ignoring --archive-directory since no probes
use directory polling
[OK]
```

Cautions

- When creating the silk.conf file, make sure you are in the right directory.
- To edit the contents of the silk.conf file you will need to ensure that you elevate your privileges using sudo otherwise it will only open as read only given the directory you are in.
- Be cautious when you are walking through the setup. During this time it might be possible that your network interfaces change certain settings that are required. After the reboot be sure that your network interfaces are set to Promiscuous mode: "Allow All".
- Additionally, if your rwflowpack does not start for some reason the first place you want to look at for errors is in the log file which is located at: /var/log/rwflowpack-[DATE].log.
- You might also check that the eth0 still has the IP address 192.168.1.104.

To be able to use the data collected to perform common network auditing tasks.

Approach

-

Execution

Now that your collector is waiting for data, you can start YAF to begin generating flow data. Begin by installing YAF, libfixbuf and Silk on the Ubuntu-LAN machine, like you did with SecurityOnion.

Once you have installed these, the same way in which you did for your Ubuntu machine, you need to make some additional alterations to your SecurityOnion machine. That machine also has an inbuilt firewall called

Lab 3 – Session Data by Bilal Khan

ufw. **[EXPAND ON UFW HERE]**. Go back to your SecurityOnion machine and add the following rules:

```
$ ufw allow 18001/tcp
```

```
$ ufw allow 1234
```

These two rules will allow tcp connections on port 18001 through, which will be used by your sensors, and allow connections on port 1234 that is going to be part of your testing. Once you have done so it should look like the below:

```
seconion@SecurityOnion:/var/log$ service ufw status
ufw start/running
seconion@SecurityOnion:/var/log$ sudo ufw allow 18001/tcp
[sudo] password for seconion:
Rule added
Rule added (v6)
seconion@SecurityOnion:/var/log$ sudo ufw allow 1234
Rule added
Rule added (v6)
seconion@SecurityOnion:/var/log$
```

Now that is all complete you can now go ahead and run the command needed for YAF to work. On the Ubuntu-LAN machine, issue the following command:

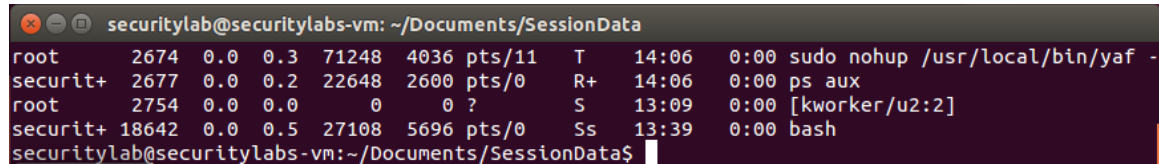
```
$ sudo nohup /usr/local/bin/yaf --silk --ipfix=tcp
--live=pcap --output=192.168.1.105 --ipfix-port=18001
--in=eth0 --applabel --max-payload=384 &
```

```
securitylab@securitylabs-vm:~/Documents/SessionData$ sudo nohup /usr/local/bin/yaf --silk --ipfix=tcp --live=pcap --output=192.168.1.105 --ipfix-port=18001 --in-eth0 --applabel --max-payload=384 &
[1] 18618
securitylab@securitylabs-vm:~/Documents/SessionData$ nohup: ignoring input and appending output to 'nohup.out'
```

You'll notice that several of the arguments you are calling in this YAF execution string match values you've configured in your SiLK configuration files.

You can verify that everything started up correctly by running `ps aux` to make sure that the process is running, as shown below:

Lab 3 – Session Data by Bilal Khan



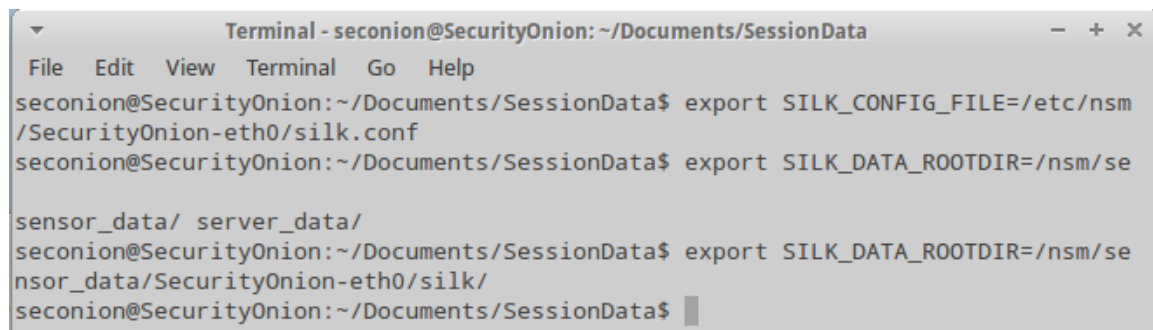
```
securitylab@securitylabs-vm: ~/Documents/SessionData
root      2674  0.0  0.3  71248  4036 pts/11   T   14:06   0:00 sudo nohup /usr/local/bin/yaf -
securit+  2677  0.0  0.2  22648  2600 pts/0    R+  14:06   0:00 ps aux
root      2754  0.0  0.0      0      0 ?        S   13:09   0:00 [kworker/u2:2]
securit+ 18642  0.0  0.5  27108  5696 pts/0    Ss  13:39   0:00 bash
securitylab@securitylabs-vm:~/Documents/SessionData$
```

If YAF doesn't appear to be running, you can check the `nohup.out` file for any error messages that might have been generated.

That's it! If your sensor interface is seeing traffic, then YAF should begin generating IPFIX flow data and sending it to `rwflowpack` for collection.

You have to tell the SiLK `rwtools` where the site configuration file is on your SecurityOnion machine. This can be done by exporting the `SILK_CONFIG_FILE` variable.

```
$ export SILK_CONFIG_FILE=/etc/nsm/<$SENSOR-$INTERFACE>/silk.conf
$ export SILK_DATA_ROOTDIR=/nsm/sensor_data/<$SENSOR-$INTERFACE>/silk/
```



```
Terminal - seconion@SecurityOnion: ~/Documents/SessionData
File Edit View Terminal Go Help
seconion@SecurityOnion:~/Documents/SessionData$ export SILK_CONFIG_FILE=/etc/nsm
/SecurityOnion-eth0/silk.conf
seconion@SecurityOnion:~/Documents/SessionData$ export SILK_DATA_ROOTDIR=/nsm/se
sensor_data/ server_data/
seconion@SecurityOnion:~/Documents/SessionData$ export SILK_DATA_ROOTDIR=/nsm/se
nsor_data/SecurityOnion-eth0/silk/
seconion@SecurityOnion:~/Documents/SessionData$
```

If you don't want to have to do the above every time you log into the system, you can place these lines in your `~/ .bashrc` file.

You should be able to use `rwfilter` now. If everything is setup correctly and you are capturing data, you should see some output from this command:

```
$ rwfilter --sensor=S0 --proto=0-255 --type=all --pass=stdout | rwcut
```

The next step is to make several different web connections from the LAN machine to various Internet websites and also use `nc` to make several different connection on a range of ports. This will show you how it can be successfully parse lots of different type of data to show you relevant information.

On your SecurityOnion machine issue the various commands below in order to collect three different types of information:

Use SiLK to analyse the top web site addresses accessed.

Go back onto your Ubuntu-LAN machine and start browsing a number of different websites. Be sure to visit your popular websites more than once for the purposes of this part of the exercise. Once you've done that after 5-10 minutes of activity go back to your SecurityOnion machine and issue the following command:

```
$ rfilter --sensor=S0 --application=80,443 --proto=0-255  
--type=all --pass=stdout | rwstats --top --count=20  
--fields=sip,dip,sPort,dPort,duration,application  
--value=records > TopWebsites
```

There are two different types of switches for the rfilter command; selection and partitioning switches. Select switches can determine the type of collection on which sensor to process along with the flow direction of the traffic. Partitioning switches on the other hand allows you to set what type of traffic behaviour is in the pass file and the rest in the fail file, depending on if the traffic flow stream matches the requirements or not².

The command above tells SiLK to use the sensor S0, the one that you had set up earlier on the Ubuntu-LAN machine, in order to inspect the packets and if it matches a traffic signature of either HTTP or HTTPS to process it as part of the pass file. This would be sent to standard out (the screen). It then gets piped through rwstats to commute a top list from the data in the pass file³. It does this by counting the top 20 records with the fields source IP, destination IP, source port, destination port, duration of the flow and the application that was used.

Finally all of the aggregated data got piped out to a file called TopWebsites which you can look at to see your data.

The biggest issue with this command is the fact that are often so many other connections that are pulled when a user connects to a website that there is just a massive amount of different flows. The problem is one

²<http://tools.netsa.cert.org/silk/rwfilter.html>

³<http://tools.netsa.cert.org/silk/rwstats.html>

would not necessarily want to limit it to outgoing only because then you wouldn't see what was coming in, which is arguably more important.

Look at the `rwstats` command and see what interesting statistics you can find out with the data that you have accumulated.

Use SiLK to determine the total volume of data that was observed, minute by minute.

```
rwfilter --sensor=S0 --application=80,443 --proto=0-255  
--type=all --pass=stdout | rwcount --bin-size=60 > WebVolume
```

Similar to the first command you can use `rwfilter` in the same manner as before to process certain data based on the web application. This time you should use `rwcount` as your piped filter because this allows flow records to be summarized across time⁴ such as observing data “*minute by minute*”. The bin-size is set to 60 seconds because the default was 30 seconds and the output is sent to a file called `WebVolume`.

Additionally, `rwcount` should be used in this instance because it automatically provides the records, bytes and packets as separate columns thus giving a good visual representation of the total volume across different collection methods.

This command is the easiest to read from a user point of view. You could quickly see the traffic by minute using a number of different methods. It told you the amount of packets, records and bytes in total for that minute. This would be a quick easy way to monitor your network traffic levels if there was a need.

Use SiLK to analyse the time and volume of non-web traffic.

Before you issue your next command, go back to your Ubuntu-LAN machine and setup an `nc` connection to the SecurityOnion machine. Set up the listener on the SecurityOnion machine and then start writing some example text between the two machines using `nc`. This should be enough to generate some data for the next example.

⁴<https://tools.netsa.cert.org/silk/rwcount.html>

Lab 3 – Session Data by Bilal Khan

```
rwfilter --sensor=S0 --sport=0-79,81-442,444-65535
--dport=0-79,81-442,444-65535 --proto=0-255 --type=all
--pass=stdout | rwuniq --fields=
sip,dip,sPort,dPort,sTime,eTime,duration,application
--values=bytes,packets,records > Non-WebTraffic
```

Trying to analyse the time and volume of non-web traffic is more challenging because you are endeavouring to find a way in which you can analyze both time and volume as well as discard any web traffic. In the example on the previous page it uses source and destination ports to exclude anything from or to ports 80 or 443. This should leave you with traffic relating to the nc commands that you sent using port 1234 and any other none related web traffic, such as DNS requests perhaps.

The rwuniq command was implemented because this gives you the ability to use time and volume data in within the specified fields. In the example above the source IP, destination IP, source Port, destination Port, start Time of the flow, end Time of the flow, Duration of the flow and what application was used. Using all three different value types should hopefully allow you to get as much volume data as possible by using bytes, packets and records. All of the outputted data to Non-WebTraffic.

Cautions (Optional)

- Need to set up your WAN interface so that it uses DHCP on M0n0wall2.
- Ensure that the DNS Forwarder is on with the M0n0wall2 machine.

To understand the tradeoffs between full packet and session data capture with respect to network security monitoring.

Approach

-

Execution

You

Lab 3 – Session Data by Bilal Khan

Cautions (Optional)

-

Reflections