Bro Network Security Monitor

This lab provides a basic understanding of Bro, a network security monitor. You will learn how to configure the Bro tool and utilize it in a real-world situation. In the days of data breaches, it is important to ensure that systems have network security monitors installed on their systems as both a detective and a preventative measure.

This lab focuses strictly on Linux distributions. We deal with three machines – Good, Bad and Ugly. As shown in the Lab Network Map, there are two internal machines (Good, Ugly) and one external machine (Bad). Bad is the attacker, sending malicious injects to the Ugly. The Bro tool is located on Good to detect the malicious activity happening within the internal network.

Learning Objectives

By the end of this lab, you should be able to do the following:

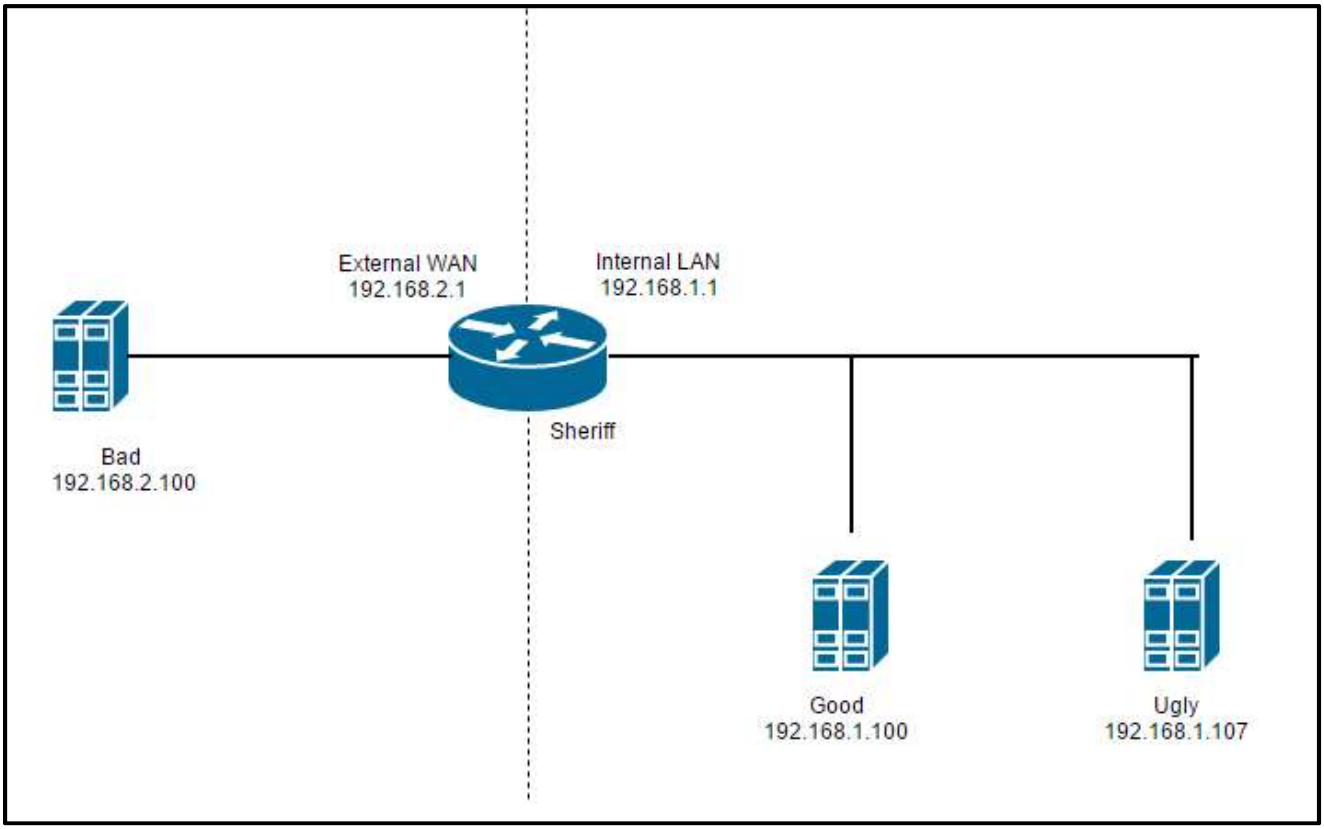
- Configure the Bro tool
- Customize Bro using scripts to suit the specific needs of the organization
- Analyze the logs generated by Bro and respond to them.

Scenario/Background

The healthcare company, GVH, has recently adopted a new online portal to host current member information. It is only a year-old portal and has successfully moved through the development and testing phases with no issues. It was developed by a highly trusted third party that GVH has been doing business with for years.

Since GVH has been gaining huge popularity in the past year, its competitor CDN is plotting a plan against them. They want to keep their reputation clean; hence they hire a security breaker – 'Kali'. Kali uses his terminal, Bad, to find a vulnerability to exploit on one of the servers of the GVH internal network. This gives them the opportunity to steal their data. They also plan to perform a Denial of Service attack on the GVH network. During this lab, the user gets to be "Bad" by sending some code injects from Kali, while simultaneously being "Good" by using Bro to detect this malicious activity.

Lab Network Diagram



The lab environment consists of three virtual computer systems.

1. **Bad:** A Kali Linux system that will send the code injects. IP address: **192.168.2.100**. Username and password: **root** | **bad@1**.
2. **Good:** An Ubuntu system where Bro will be performing network monitoring. IP address: **192.168.1.100**. Username and password: **good** | **good@1**.
3. **Ugly:** An Ubuntu system that Bad will target, and that will be the infected system. IP address: **192.168.1.107**. Account and password: **ugly** | **ugly@1**.

The lab environment also has a router/firewall server called Sheriff. You will not be expected to perform any activities on Sheriff, aside from preparing it as part of the virtual environment.

Preparing the Virtual Environment

Before you begin the lab, you must ensure that all of the virtual machines are connected to the same network. When you prepare the virtual machines, you need to create two local segments: one for LAN and one for WAN. The machines will reside on the local segments as follows:

Bad: WAN

Good: LAN

Ugly: LAN

Sheriff: LAN and WAN

Note: These instructions are for VMWare. If you are using a different software, use similar instructions to ensure the machines are prepared as above.

Bad

1. Open up VMWare. Right click on Bad and go to the Settings.

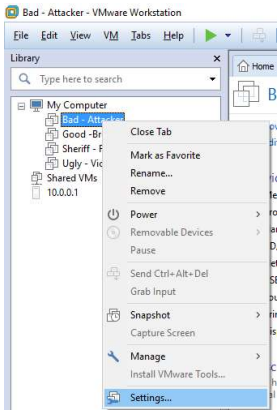


Figure 1. Opening Bad settings

2. In the Settings, navigate to Network Adapter and choose LAN segment under Network Connections.

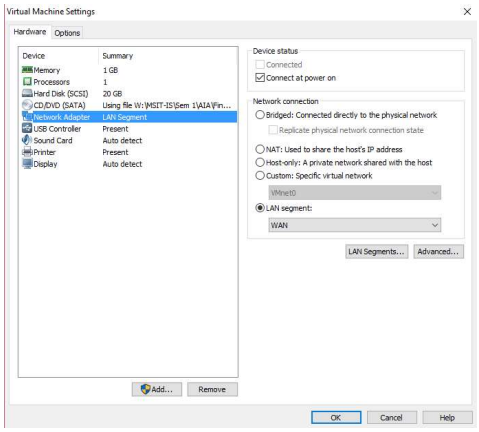


Figure 2. Bad settings

3. If WAN is not included in the drop down of LAN segments, click on the LAN Segments… box which will open the Global LAN Segments.

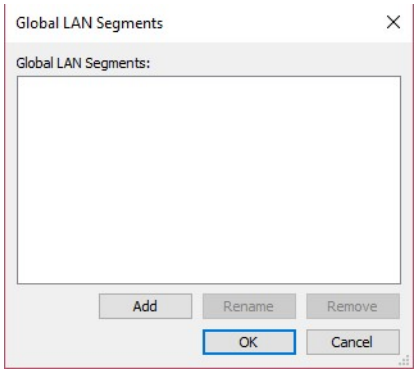


Figure 3. Global LAN Segments.

4. Click the Add button.
5. Type in WAN.
6. Click OK to add WAN as a Global LAN segment.
7. Under the drop down for LAN segment, ensure that WAN is selected

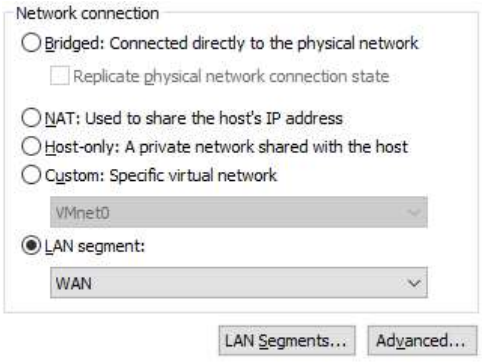


Figure 4. WAN as the LAN segment

8. Click OK to exit out of the settings for Bad.

Good

1. Open up VMWare. Right click on Good and go to the Settings.
2. In the Settings, navigate to Network Adapter and choose LAN segment under Network Connections.
3. If LAN is not included in the drop down of LAN segments, click on the LAN Segments… box which will open the Global LAN Segments.
4. Click the Add button.
5. Type in LAN.
6. Click OK to add LAN as a Global LAN segment.
7. Under the drop down for LAN segment, ensure that LAN is selected.

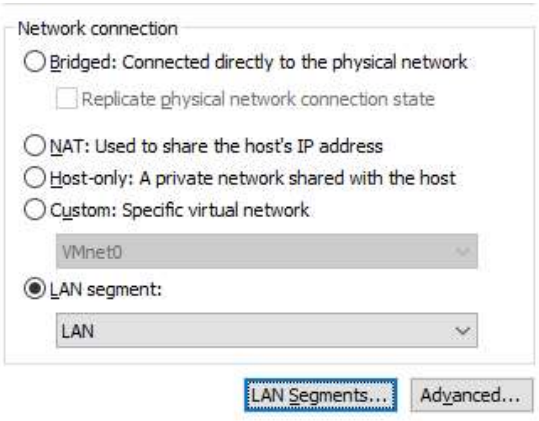


Figure 5. LAN as LAN Segment

8. Click OK to exit out of the settings for Good.

Ugly

1. Open up VMWare and right click on Ugly and go to the Settings.
2. In the Settings, navigate to Network Adapter and choose LAN segment under Network Connections.
3. Because you have already created the LAN option, choose LAN from the drop down and it should look like Figure 5.
4. Click OK to exit out of the settings for Ugly.

Sheriff

1. Open up VMWare and right click on Sheriff and go to the Settings.
2. You will now see Network Adapter and Network Adapter 2.

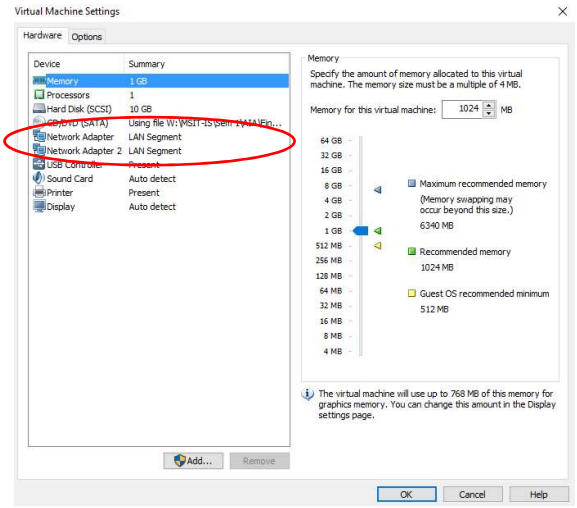


Figure 6. Sheriff Settings

3. In the Settings, navigate to Network Adapter and choose LAN segment under Network Connections.
4. Because you have already created the LAN option, choose LAN from the drop down, and it should look like **Figure 5**.
5. Now, go to Network Adapter 2, and choose LAN segment under Network Connections.
6. Because you have already created the WAN option, choose WAN from the drop down, and it should look like **Figure 4**.
7. Click OK to exit out of the settings for Sheriff.

Configuring IP Addresses

Additionally, you may need to set up the IP address for the VMs, as they may have automatically reset with the changes done to the Network Adapters.

On Bad:

1. Power on the Bad VM. Log on to Bad using:

   Username: **root** and Password: **bad@1**

2. Click on the downward arrow at the top right corner of the screen and choose Wired Connected → Wired Settings

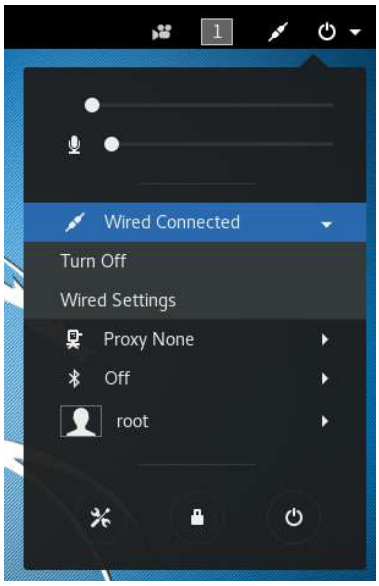


Figure 1. Wired Settings

3. Check that the IPv4 address of Bad is set to 192.168.2.100. If it is not, click on this icon 

4. Select IPv4 from the choices on the left pane, select “Manual” from the Addresses dropdown list, then enter the data noted below in Figure 2.

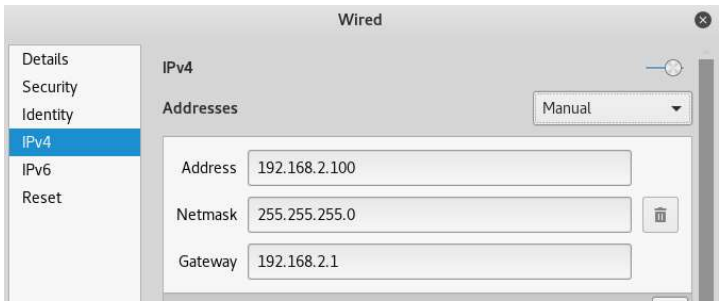


Figure 2. IPv4 Settings for Bad

5. Click Apply, then close the Network Setting window.

6. Open the terminal by clicking on the terminal icon. 

7. Check that the IP address has been properly set up by using the command `ifconfig`. Confirm that the IPv4 address is 192.168.2.100.

On Good:

1. Power on the Good VM. Log on to Good using:

   Username: **good** and Password: **good@1**

2. Click on the network icon at the top right corner of the screen and choose Edit Connections

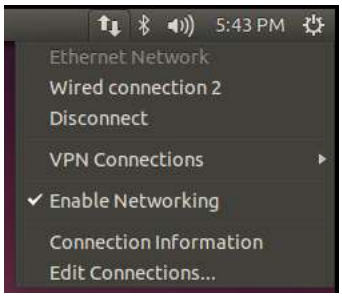


Figure 1. Network Connections

3. Select the current wired connection, then click on Edit
4. Go to the IPv4 Settings tab. The Method selected should be “Manual” and the IP address should be 192.168.1.100. Change the method to “Manual”, and click on Add in order to enter the manual IP address, Netmask and Gateway details (see Figure 2 below).

Figure 2. IPv4 Settings for Good

5. Click on Save, then Close.
6. Open the terminal with [Ctrl] + [Alt] + [T]. Check that the IP address has been properly set up by using the command `ifconfig`. Confirm that the IPv4 address is 192.168.1.100.

On Ugly:

1. Power on the Ugly VM. Log on to Ugly using:

   Username: **ugly** and Password: **ugly@1**

2. Click on the network icon at the top right corner of the screen and choose Edit Connections

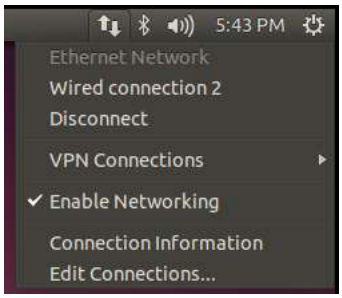


Figure 1. Network Connections

3. Select the current wired connection, then click on Edit
4. Go to the IPv4 Settings tab. The Method selected should be “Manual” and the IP address should be 192.168.1.107. Change the method to “Manual”, and click on Add in order to enter the manual IP address, Netmask and Gateway details (see Figure 2 below).

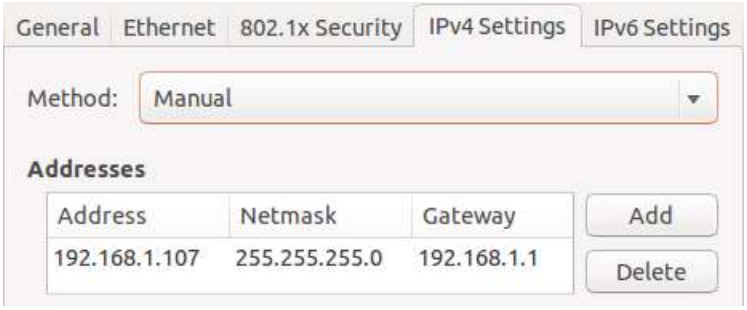


Figure 2. IPv4 Settings for Ugly

5. Click on Save, then Close.
6. Open the terminal with [Ctrl] + [Alt] + [T]. Check that the IP address has been properly set up by using the command `ifconfig`. Confirm that the IPv4 address is 192.168.1.107.

On Sheriff:

1. Power on the Sheriff VM. You will not need to enter a user name or password.
2. When the VM completes booting up, confirm that the network interfaces have been configured with the correct IP addresses

   WAN: 192.168.2.1/24

   LAN: 192.168.1.1/24

```
*** Welcome to pfSense 2.3.2-RELEASE (amd64 full-install) on Sheriff ***

 WAN (wan)       -> le1        -> v4: 192.168.2.1/24
 LAN (lan)       -> le0        -> v4: 192.168.1.1/24
```

Figure 1. Network Interfaces on Sheriff

3. If the set up does not match Figure 1, set the interface IP address(es) by typing `2` then hitting [Enter].
4. Select the interface to edit (`1` for WAN, `2` for LAN)
5. Type `n` when asked if you want the interface to be configured via DHCP
6. Enter the new IP address as noted in Step 2 above.
7. Enter `24` for the subnet.
8. Enter `192.168.2.1` for the WAN upstream gateway address (if updating WAN). Just hit [Enter] if updating LAN.

9. Type `n` when asked if you want to configure the IPv6 address via DHCP6.
10. Just hit [Enter] when asked for the IPv6 address.
11. Hit [Enter] again to complete the process.
12. Repeat for the other interface as needed.

Set up of Bro

For this lab, you have the Bro tool already installed and ready with its default settings. You will need to edit/verify the settings as per the network requirements and enable it. Additionally, you will initialize and begin to monitor using Bro.

Configuration of Bro

1. Log on to Good using:

   Username: **good** and Password: **good@1**

2. Open the terminal with [Ctrl] + [Alt] + [T].
3. To set up the right interface to monitor, type the following command to open up the folder where the configuration files are stored:

```
$ cd /opt/bro2.4/etc/
```

4. Display the list of configuration files by typing:

```
$ ls
```

Figure 1. Configuration Files

5. Open the node file, which is a configuration file that enables you to change how Bro is set up and which interface it monitors.

```
$ sudo vi node.cfg
```

6. When prompted for the password, type in **good@1**.
7. Verify that the below options are the default in the file. If they are not change them to the ones below by pressing [i]:

   type=standalone;

   host=localhost;

   interface=eth0

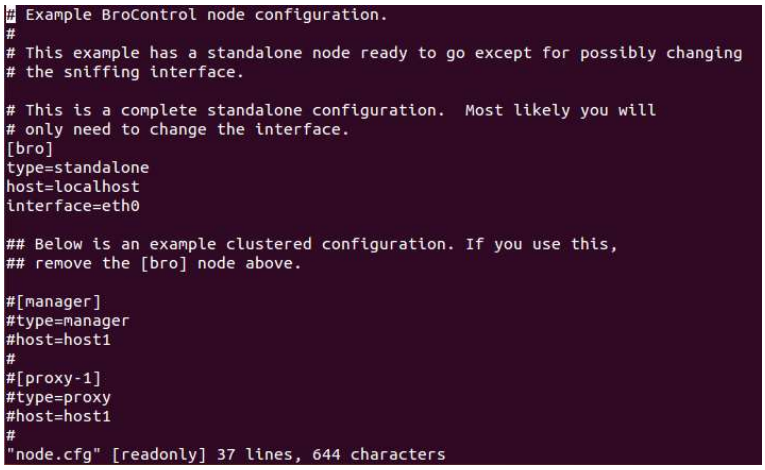


Figure 2. BroControl node configuration

8. If you have made any changes, save them by pressing [esc] and typing `:wq`. Then press [Enter].
9. Now open the networks.cfg file. This file enables you to change the local network IP that Bro should monitor. Use:

```
$ sudo vi networks.cfg
```

10. When prompted for the password, type in **good@1**.
11. Press [i].
12. Replace the default network(s) identified to **192.168.1.0/24**. Your output should look like the below.

Figure 3. networks.cfg file

13. Save your changes and exit by pressing [esc]. Type `:wq`. Then press [Enter].

    In a full-scale Bro deployment, there would be more settings to configure, such as Bro Clusters. For the purpose of this lab, we will only be concerned with the above changes.

Starting up Bro

Since you have ensured that Bro settings are configured and the settings are verified as per the network requirements, you can now start up Bro by using BroControl, Bro's command line utility.

1. In the terminal, type:

```
$ sudo /opt/bro2.4/bin/broctl
```

2. When prompted for the password, type **good@1**.
3. Use the below command to start Bro

```
> start
```

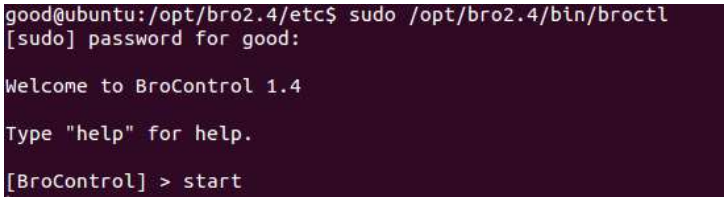


Figure 4. Starting Bro

4. To exit the Bro terminal type `exit`. Note that Bro will still be enabled. You are just exiting of BroControl to get back to the good@ubuntu command line.

   Bro is now monitoring the traffic passing through the network. It will collect information and report on what it finds by updating the different Bro logs.

Bad Scans the Internal Network

Now Bad will attempt to scan the internal network via a SYN Scan. You will be reviewing the Bro logs on Good to see how Bro will detect such activity.

Bad performs the SYN Scan

1. Open up Bad and log in with username: **root** and password **bad@1**.
2. Open the terminal by clicking  on the left hand side of the screen.
3. Go to the scripts directory by typing

```
# cd ~/Desktop/scripts
```

4. To check the files in the scripts directory, type `ls`. You will see the below.

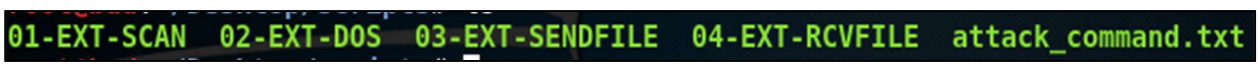


Figure 1. Scripts in Bad

5. The SYN scan will be performed using nmap. The attack has already been encoded in a script. To initiate the SYN scan with nmap type:

```
# ./01-EXT-SCAN
```

6. Wait for the nmap to complete (you will see a summary), and then stop the scanning by pressing [Ctrl] + [C].

Check Logs on Bro

1. Open up Good and log in with username: **good** and password: **good@1**
2. Open the terminal by pressing [Ctrl] + [Alt] + [T].
3. Go to the current logs directory by typing

```
$ cd /opt/bro2.4/logs/current
```

4. Open the notice.log file in this folder by using the below command.

```
$ cat notice.log
```

5. Bro has identified the port scan that has happened. You should see something similar to Figure 2 below. **Note:** Bro comes with some default scripts that automatically checks for certain events. In this case, it is checking for port scanning activity.

Figure 2. Bro detects the port scanning activity

6. Now open up the conn.log file. This will show all connections in the local network that Bro is monitoring.

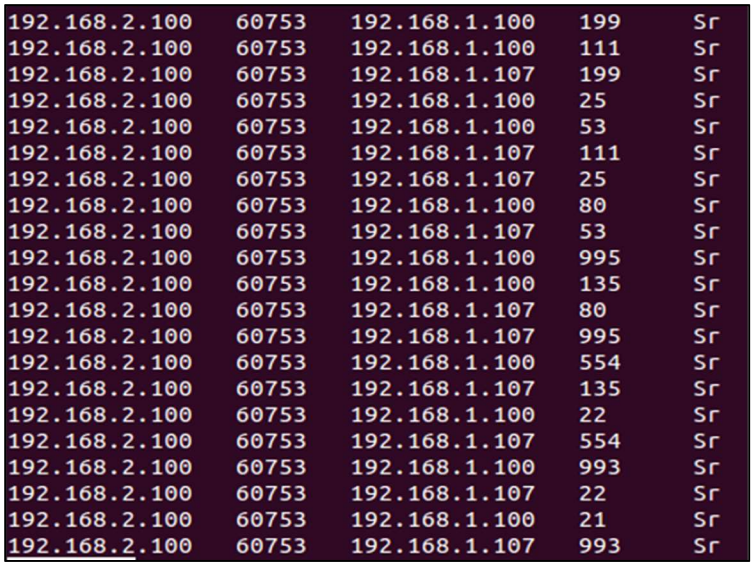
```
$ sudo cat conn.log | /opt/bro2.4/bin/bro-cut id.orig_h id.orig_p
id.resp_h id.resp_p history | more
```

It will pull out following fields in that log

a. Source IP
b. Source port

c. Destination IP
d. Destination port
e. History
   - The History field shows you state history of connections as a string of letters
   - Capital letters - comes from the source host
   - Lowercase letters - come from the destination host

7. You should see something like the below screenshot as you scroll through the report.



192.168.2.100	60753	192.168.1.100	199	Sr
192.168.2.100	60753	192.168.1.100	111	Sr
192.168.2.100	60753	192.168.1.107	199	Sr
192.168.2.100	60753	192.168.1.100	25	Sr
192.168.2.100	60753	192.168.1.100	53	Sr
192.168.2.100	60753	192.168.1.107	111	Sr
192.168.2.100	60753	192.168.1.107	25	Sr
192.168.2.100	60753	192.168.1.100	80	Sr
192.168.2.100	60753	192.168.1.107	53	Sr
192.168.2.100	60753	192.168.1.100	995	Sr
192.168.2.100	60753	192.168.1.100	135	Sr
192.168.2.100	60753	192.168.1.107	80	Sr
192.168.2.100	60753	192.168.1.107	995	Sr
192.168.2.100	60753	192.168.1.100	554	Sr
192.168.2.100	60753	192.168.1.107	135	Sr
192.168.2.100	60753	192.168.1.100	22	Sr
192.168.2.100	60753	192.168.1.107	554	Sr
192.168.2.100	60753	192.168.1.100	993	Sr
192.168.2.100	60753	192.168.1.107	22	Sr
192.168.2.100	60753	192.168.1.100	21	Sr
192.168.2.100	60753	192.168.1.107	993	Sr

Figure 3. Connections that Bro is monitoring

8. You will see entries from 192.168.2.100 (Bad) to hosts in 192.168.1.0 network. The entries are all going to different ports and the history says Sr – This means that the source sent a SYN packet and the destination responded with a RST, indicating that the port was closed. Looks like a classic SYN Scan!
9. Exit the report by pressing [Ctrl] + [C].

Bad Performs the DoS Attack

Now that Bad has found a vulnerable host, Bad will attempt to bring Ugly down with a DoS attack. You will be reviewing the Bro logs on Good to see how Bro will detect such activity.

The attack

1. Open up Bad and log in with username: **root** and password: **bad@1.**
2. Go to the scripts directory by typing

```
# cd ~/Desktop/scripts
```

3. The DoS attack will be executed via an Hping flood. The attack has already been encoded in a script. Execute the DoS attack on Ugly by typing

```
# ./02-EXT-DOS
```

4. Stop the command after 15 seconds by typing [Ctrl] + [C].
5. Wait 2 minutes before checking the logs on Bro.

Checking Logs on Bro

1. Open up Good and log on with username: **good** and password: **good@1**.
2. Open the terminal by pressing [Ctrl] + [Alt] + [T].
3. Go to the current logs directory by typing

```
$ cd /opt/bro2.4/logs/current
```

4. Open the conn.log file by typing the below.

```
$ sudo cat conn.log | /opt/bro2.4/bin/bro-cut id.orig_h id.orig_p
id.resp_h id.resp_p history proto | more
```

Note, we have added the proto field in addition to the previous fields included in the report. This new field will display the transport protocol used. You will see a report like the Figure 1 when you scroll down the report (you will need to scroll down several pages, as the conn.log file will have all of the entries from the SYN scan that was executed earlier at the top of the report).

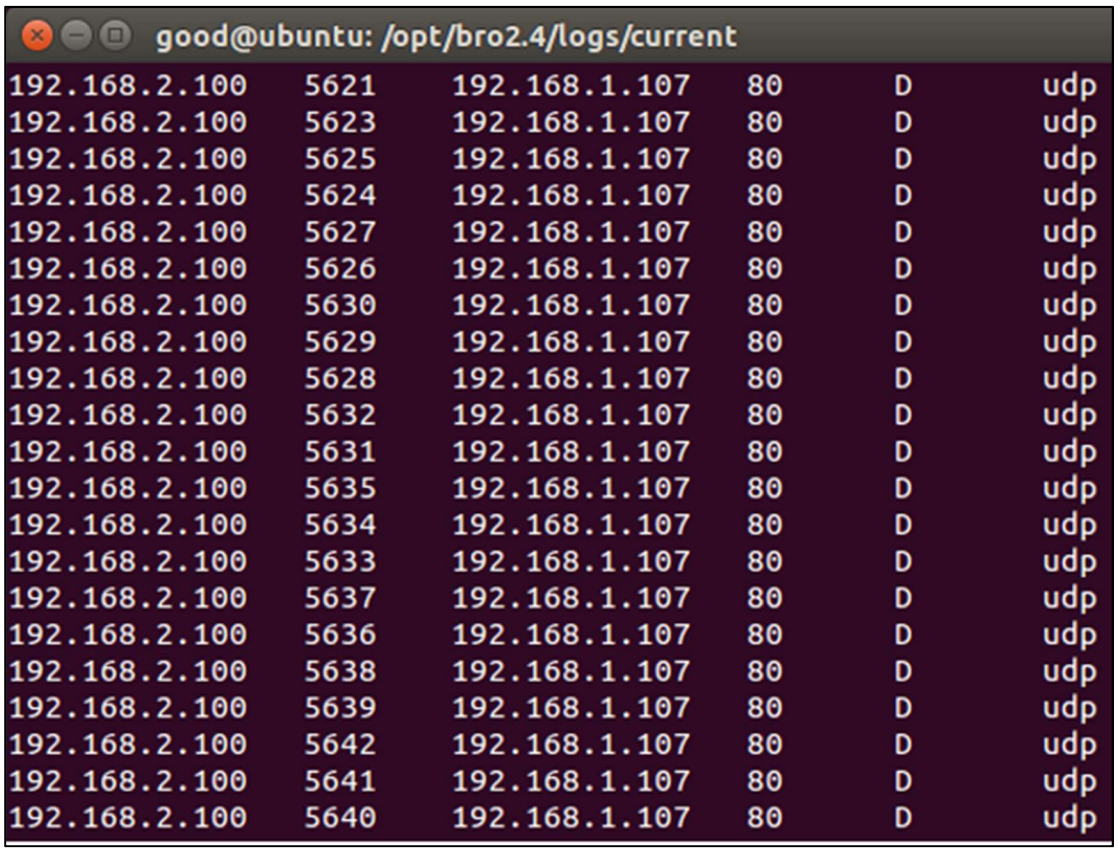


Figure 1. DoS attack logs

There will be thousands of entries from 192.168.2.100 (Bad) with increasing port numbers all targeting port 80 on Ugly. Making it even more suspicious… UDP traffic is being sent to port 80. Port 80 is HTTP, which is TCP not UDP.

Seems like a DoS attack!

Ugly is Compromised

Prepare Bro

Before we run any scripts, we need to prepare Bro.

1. Log on to Good with username: **good** and password: **good@1**
2. Open the terminal by pressing [Ctrl] + [Alt] + [T].
3. Go to the directory

```
$ cd /opt/bro2.4/share/bro/base/protocols/conn
```

4. Open the contents.bro file present in this location. This is a Bro script that extracts file contents that get transferred over the network. If Bro detects a file transfer, it will extract the contents and write it to a data file. We will edit this script to enable this feature.

```
$ sudo vi contents.bro
```

5. When prompted for the password, enter **good@1**.
6. Press the [i] key.
7. In this file, you will:

   Comment out: `#const default_extract = F & redef;`

   Add: `const default_extract = T & redef;`

We are setting this variable to true so that this script can read the packets being transferred and extract any file data it sees.

```
#const default_extract = F &redef;

const default_extract = T &redef;
```

Figure 1. Contents.bro file

8. We will now set up Good to capture network traces. This will be used later to check if files were transferred. Go to:

```
$ cd ~/Desktop
```

9. Make the traces directory by typing

```
$ mkdir traces
```

```
$ cd traces
```

10. We will now run tcpdump and save the result into the network.pcap file. Type

```
$ sudo tcpdump -i eth0 -s 0 -vvv -w network.pcap
```

Note: we will let this command run until this section ends.

The Attack File is Downloaded

Having been compromised, Ugly will now receive instructions from Bad. These commands control Ugly and will get] him to do harmful things.

1. Open up Ugly and log on using username: **ugly** and password: **ugly@1.**
2. Open the terminal by pressing [Ctrl] + [Alt] + [T].
3. Go to the scripts directory by typing:

```
$ cd ~/Desktop/scripts
```

4. In order to simulate the infected user downloading a command file from the attacker, type the following command:

```
$ ./01-INT-DOWNLOAD
```

This opens up a netcat session, listening on port 60000. Bad will be sending the command file to Ugly using this port.

5. Go back to Bad and log on using username: **root** and password: **bad@1**.
6. Open the terminal if not already opened.
7. Go to the scripts directory:

```
# cd ~/Desktop/scripts
```

6. Bad will now send a command file to Ugly using netcat. This command has already been encoded in a script. Send the command file to Ugly by typing:

```
# ./03-EXT-SENDFILE
```

8. Wait 15 seconds and stop the command on Bad by pressing [Ctrl] + [C].
9. Now go back to Ugly and stop the running command by pressing [Ctrl] + [C].
10. To confirm that the file was downloaded correctly in Ugly, type `ls`. You should see the attack_command.txt file in the scripts directory.

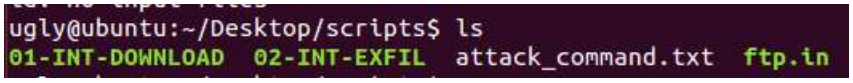


Figure 1. attack_command.txt file in Ugly's script directory.

Checking Logs on Bro

1. Wait 2 minutes, then log on to Good with username: **good** and password: **good@1.**
2. Open the terminal if not already opened by pressing [Ctrl] + [Alt] + [T]. **Note**: Do not use the same terminal where tcpdump is running. We do not want to stop that process at this time.
3. Go to the current logs directory by typing:

```
$ cd /opt/bro2.4/logs/current
```

4. Now we wish to look at the current traffic that is passing through the network. To do so, type the following command (this is all on one line):

```
$ sudo /opt/bro2.4/bin/bro-cut id.orig_h id.orig_p id.resp_h id.resp_p
history proto < conn.log | awk '{ if ($1 == "192.168.2.100" && ($5 ==
"ShADa" || $5 == "ShADaFf" || $5 == "ShADadFf"))  { print $1, $2, $3,
$4, $5, $6 }}' | more
```

Note that we are searching for traffic coming from our known attacker (Bad with IP address 192.168.2.100). Also, we are filtering based on the history where some data got sent.

ShADaFf stands for

- Capital S - SYN sent by source host
- Lowercase h - SYN-ACK response by destination
- Capital A - ACK sent by source host
- Capital D - data payload sent by source host
- Lowercase a - ACK response by destination host
- Capital F - FIN sent by source host to close connection
- Lowercase f - FIN sent by destination host to close connection

Several variations of this history were used in the awk command.

5. Expect to get a report like the below

```
good@ubuntu:/opt/bro2.4/logs/current$ sudo /opt/bro2.4/bin/bro-cut id.orig_h id.
orig_p id.resp_h id.resp_p history proto < conn.log | awk '{ if ($1 == "192.168.
2.100" && ($5 == "ShADa" || $5 == "ShADaFf" || $5 == "ShADadFf"))  { print $1, $
2, $3, $4, $5, $6 }}' | more
192.168.2.100 59420 192.168.1.107 60000 ShADaFf tcp
```

Figure 1. Suspicious Traffic

Looks like the suspicious IP address has sent some data to Ugly – possibly this is a command file instructing Ugly to misbehave. We will confirm this later via some other commands using Bro.

Ugly Misbehaving

Data Exfiltration

1. Log on to Bad with username: **root** and password: **bad@1**.
2. Open the terminal if it's not already opened.
3. Go to the scripts directory

```
# cd ~/Desktop/scripts
```

4. We will enable a listening port on Bad by typing:

```
# ./04-EXT-RCVFILE
```

This opens up a netcat session, listening on port 21. Ugly will be sending the compromised data to Bad using this port.

5. Now go onto Ugly and log in with username: **ugly** and password: **ugly@1**.
6. Open the terminal if it's not already opened.
7. In the terminal go to the scripts directory by typing

```
$ cd ~/Desktop/scripts
```

8. To send out confidential information, type the below command:

```
$ ./02-INT-EXFIL
```

The compromised data is sent as a TCP packet on the FTP port - simulating an FTP connection.

9. Wait a couple of minutes, and stop the command on Bad by pressing [Ctrl] + [C].
10. Now go back to Ugly and stop the running command by pressing [Ctrl] + [C].
11. Go back to Bad, and you will now see a file called ftp.in in the scripts directory after this executes. Type `ls` to see the new file.

Figure 1. ftp.in file in the Bad's script directory.

Checking Logs on Bro

1. Wait 2 minutes, then log on to Good with username: **good** and password: **good@1**.
2. Open the terminal if not already opened by pressing [Ctrl] + [Alt] + [T]. **Note:** At this point, you can already stop the tcpdump by pressing [Ctrl]+[C].
3. Open the current Bro logs directory by typing:

```
$ cd /opt/bro2.4/logs/current
```

4. In order to look at the FTP traffic, you need to type in the below command (this is all on one line):

```
$ sudo /opt/bro2.4/bin/bro-cut id.orig_h id.orig_p id.resp_h id.resp_p
history proto < conn.log | awk '{ if ($4 ==21) { print $1, $2, $3, $4,
$5, $6 } } ' | more
```

Note that we are looking for all traffic on port 21 which is the FTP port. We assume this organization does not use FTP, so this is suspicious traffic. You will get a report like the one in Figure 1.

```
good@ubuntu:/opt/bro2.4/logs/current$ sudo /opt/bro2.4/bin/bro-cut id.orig_h id.
orig_p id.resp_h id.resp_p history proto < conn.log | awk '{ if ($4 ==21) { prin
t $1, $2, $3, $4, $5, $6 } } ' | more
192.168.1.107 33518 192.168.2.100 21 ShADFaf tcp
192.168.1.107 33520 192.168.2.100 21 ShADFaf tcp
192.168.1.107 33522 192.168.2.100 21 Sr tcp
192.168.1.107 33524 192.168.2.100 21 ShADFaf tcp
192.168.1.107 33526 192.168.2.100 21 Sr tcp
192.168.1.107 33528 192.168.2.100 21 ShADFaf tcp
192.168.1.107 33530 192.168.2.100 21 Sr tcp
```

Figure 1. FTP traffic

Note the multiple FTP connections from 192.168.1.107 (Ugly) to 192.168.2.100 (Bad). Also, note the history - looks like Ugly has successfully sent data outside of the network.

Bro File Extraction

1. In Good, open a new terminal by pressing [Ctrl] + [Alt] + [T].
2. Go to the traces directory.

```
$ cd ~/Desktop/traces
```

3. To look at the generated traffic, type the below command (this is all on the same line). When prompted for the password, type **good@1.**

```
$ sudo /opt/bro2.4/bin/bro -r network.pcap
/opt/bro2.4/share/bro/base/protocols/conn/contents.bro
```

4. Once this is executed, check the list of files by typing `ls`. You can see some new files get created in the directory.

```
conn.log
contents_192.168.1.107:33574-192.168.2.100:21_orig.dat
contents_192.168.1.107:33574-192.168.2.100:21_resp.dat
contents_192.168.1.107:33576-192.168.2.100:21_orig.dat
contents_192.168.1.107:33576-192.168.2.100:21_resp.dat
contents_192.168.2.100:35696-192.168.1.107:60000_orig.dat
contents_192.168.2.100:35696-192.168.1.107:60000_resp.dat
network.pcap
packet_filter.log
```

Figure 1. Newly Created Files

Recall that the contents.bro script is used to extract content of files being transferred. The *.dat files would only get created if some files were indeed being transferred over the network.

This verifies our speculations that the files were indeed transferred between Bad and Ugly.

The newly created *dat files will have filenames with the following format:

`"contents_" + Source IP address:Source Port-Destination IP address:Destination port + "_orig.dat"` → these will provide the file contents for data coming from the Source IP

"contents_" + Source IP address:Source Port-Destination IP address:Destination port + "_resp.dat" → these will provide the file contents for data coming from the Destination IP (if any)

You will notice several orig/resp pairs for port 21 (since the exfiltration scripts will send the compromised data multiple times). There will only be one pair of orig/resp files for port 60000 (this is the data that Bad sends to Ugly).

5. Open the *orig.dat file that uses port 60000. Let's take a look at what command Bad sent to Ugly. Type in the following:

```
$ cat contents_192.168.2.100:35696-192.168.1.107:60000_orig.dat
```

Replace `contents_192.168.2.100:35696-192.168.1.107:60000_orig.dat` with the actual file name created by Bro.

6. Now let's take a look how Ugly responded to this command. We will do this by opening one of the *orig.dat files that uses port 21. Type in the following:

```
$ cat contents_192.168.1.107:33574-192.168.2.100:21_orig.dat
```

Replace `contents_192.168.1.107:33574-192.168.2.100:21_orig.dat` with the actual file name created by Bro.

What data did Ugly send out?

Key Takeaways

Bro, the network security monitor tool, is a resourceful way to detect and respond to potential attacks that happen on a network. It's a flexible, versatile tool that can be used to detect a multitude of traffic patterns and create logs, reports, and alerts.

In this lab, you have learned how to use Bro to detect certain attacks that can happen in the real world. You have learned how to play both sides: acting as the malicious attacker and playing on the defensive side. You now know how to configure Bro, customize Bro using scripts, and analyze logs using Bro.

Appendices

Appendix A

Our suggested lab verification steps to ensure the lab was executed properly are the following:

1. Check that the attack_command.txt file is on Ugly
2. Check that the ftp.in file is on Bad
3. Check that the network.pcap file has been created on Good
4. Check that the logs on Good have been created (conn.log, notice.log)
5. Check that the *orig.dat files have been created on Good

Appendix B

1. What command is used to go to the Bro command line utility?
   - Bro
   - **Broctl**
   - Broccoli
   - Bro-cut
2. Which script did we edit and later execute to pull out the data on files transferred?
   - conn.bro
   - connection.bro
   - **contents.bro**
   - files.bro
3. Which of the history data below indicates some data was transferred?
   - **ShADa**
   - **ShADadA**
   - **ShADaFf**
   - **ShADadFf**
4. Which logs provided a notification that a port scan had been executed?
   - **Notice.log**
   - Conn.log
   - Alert.log
   - Scan.log
5. What command is used to extract data from Bro logs?
   - Bro
   - Broctl
   - Broccoli
   - **Bro-cut**