ZEEK INTRUSION DETECTION SERIES

Lab 1: Introduction to the Capabilities of Zeek

Document Version: 02-01-2020

Award 1829698
“CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers”
# Contents

Overview ................................................................................................................................. 3
Objectives ................................................................................................................................. 3
Lab topology ........................................................................................................................... 3
Lab settings ............................................................................................................................. 3
Lab roadmap ............................................................................................................................ 4

1  Introduction to Zeek ............................................................................................................. 4
   1.1  The Zeek event engine ................................................................................................. 5
      1.1.1  State management ................................................................................................. 5
      1.1.2  Transport layer analyzers ..................................................................................... 6
      1.1.3  Application layer analyzers .................................................................................. 6
      1.1.4  Infrastructure ......................................................................................................... 6
   1.2  The Zeek policy script interpreter .................................................................................. 6
   1.3  Zeek analyzers .............................................................................................................. 6
   1.4  Signatures ...................................................................................................................... 7
   1.5  ZeekControl .................................................................................................................. 7

2  Using ZeekControl to update the status of Zeek ............................................................... 8
   2.1  Starting a new instance of Zeek .................................................................................... 9
   2.2  Stopping the active instance of Zeek .......................................................................... 10

3  Introduction to Zeek’s traffic analysis capabilities ............................................................ 11
   3.1  Processing offline packet capture files ....................................................................... 11
      3.1.1  Command format for processing packet capture files .......................................... 11
      3.1.2  Leveraging a script to detect brute force attacks present in a pcap file ............... 12
   3.2  Generating and analyzing live network traffic capture .............................................. 13
      3.2.1  Leveraging the Tcpdump utility for capturing live network traffic ....................... 13
      3.2.2  Capturing live network traffic .............................................................................. 13
      3.2.3  Analyzing the newly captured network traffic ..................................................... 15

References ....................................................................................................................................... 17
Lab 1: Introduction to the Capabilities of Zeek

Overview

This lab introduces Zeek, an open-source network analysis framework primarily used in security monitoring and traffic analysis. The focus in this lab is on explaining Zeek’s layered architecture and demonstrating Zeek’s capabilities of performing network traffic analysis.

Objectives

By the end of this lab, students should be able to:

1. Understand Zeek’s layered architecture.
2. Start and manage a Zeek instance using the ZeekControl utility.
3. Use Zeek to process packet captures files.
4. Generate and analyze live network traffic in Zeek.

Lab topology

Figure 1 displays the topology of the lab. This lab will primarily use the Zeek2 machine for offline packet capture processing and analysis.

![Lab topology diagram]

Figure 1. Lab topology.

Lab settings

The information (case-sensitive) in the table below provides the credentials to access the machines used in this lab.
Table 1. Device credentials for lab workspace.

<table>
<thead>
<tr>
<th>Virtual Machine</th>
<th>IP Address</th>
<th>Account</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zeek1</td>
<td>192.168.1.2</td>
<td>admin</td>
<td>password</td>
</tr>
<tr>
<td>DTN</td>
<td>192.168.1.3</td>
<td>root</td>
<td>password</td>
</tr>
<tr>
<td>Client</td>
<td>192.168.3.2</td>
<td>root</td>
<td>@dmin123</td>
</tr>
<tr>
<td>Zeek2</td>
<td>192.168.2.2</td>
<td>admin</td>
<td>password</td>
</tr>
<tr>
<td></td>
<td>192.168.3.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Router</td>
<td>192.168.1.1</td>
<td>root</td>
<td>password</td>
</tr>
<tr>
<td></td>
<td>192.168.2.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>203.0.113.2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Shell variables and their corresponding absolute paths.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Absolute Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ZEEK_INSTALL</td>
<td>/usr/local/zeek</td>
</tr>
<tr>
<td>$ZEEK_TESTING_TRACES</td>
<td>/home/vlab/Zeek/testing/btest/Traces/</td>
</tr>
<tr>
<td>$ZEEK_PROTOCOLS_SCRIPT</td>
<td>/home/vlab/Zeek/scripts/policy/protocols/</td>
</tr>
<tr>
<td>$ZEEK_LABS</td>
<td>/home/vlab/Zeek-Labs-Workspace/</td>
</tr>
</tbody>
</table>

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction to Zeek.
2. Section 2: Using ZeekControl to update the status of Zeek.
3. Section 3: Introduction to Zeek’s traffic analysis capabilities.

1 Introduction to Zeek

Zeek is a passive, open-source network traffic analyzer. It is primarily a security monitor that inspects all traffic on a link in depth for signs of suspicious activity. It can run on commodity hardware with standard UNIX-based systems and can be used as a passive network monitoring tool.
Setting Zeek as a node with an assigned IP address on the monitored network is not mandatory. Figure 2 shows Zeek’s layered architecture. Once Zeek receives packets, its event engine converts them into events. The events are then forwarded to the policy script interpreter, which generates logs, notifications, and/or actions.

Zeek uses the standard libpcap library for capturing packets to be used in network monitoring and analysis.

1.1 The Zeek event engine

The event engine layer performs low-level network packets analysis. It receives raw packets from the network layer (packet capture), sorts them by connection, reassembles data streams, and decodes application layer protocols. Whenever it encounters something potentially relevant to the policy layer, it generates an event.

The event engine consists of several analyzers responsible for well-defined tasks. Typical tasks include decoding a specific protocol, performing signature-matching, identifying backdoors, etc. Usually, an analyzer is accompanied by a default script which implements some general policy adjustable to the local environment. The event engine can be divided into four major parts.

1.1.1 State management

Zeek’s main data structure is a connection which follows typical flow identification mechanisms, such as 5-tuple approaches. The 5-tuple structure consists of the source IP address/port number, destination IP address/port number, and the protocol in use. For a connection-oriented protocol like TCP, the definition of a connection is more clear-cut,
however for others such as UDP and ICMP, Zeek implements a flow-like abstraction to aggregate packets. Each packet belongs to exactly one connection.

1.1.2 Transport layer analyzers

On the transport layer, Zeek analyzes TCP, UDP packets. In TCP, Zeek’s associated analyzer closely follows the various state changes, keeps track of acknowledgments, handles retransmissions and much more.

1.1.3 Application layer analyzers

The analysis of the application layer data of a connection depends on the service. There are analyzers for a wide variety of different protocols, e.g. HTTP, SMTP or DNS, that generally conduct detailed analysis of the data stream.

1.1.4 Infrastructure

The general infrastructure of Zeek includes the event and timer management components, the script interpreter, and data structures.

1.2 The Zeek policy script interpreter

While the event engine itself is policy-neutral, the top layer of Zeek defines the environment-specific network security policy. By writing handlers for events that may be raised by the event engine, the user can precisely define the constraints within the given network. If a security breach is detected, the policy layer generates an alert.

New event handlers can be created in Zeek’s own scripting language. While providing all expected convenience of a powerful scripting language, it has been designed with network intrusion detection in mind. While it is expected that additional policy scripts are written by the user, there are nevertheless several default scripts included with the initial installation of Zeek. These default scripts already perform a wide range of analyses and are easily customizable.

1.3 Zeek analyzers

The majority of Zeek’s analyzers are in its event engine with accompanying policy scripts that can be customized by the user. Sometimes, however, the analyzer is just a policy script implementing multiple event handlers. The analyzers perform application layer decoding, anomaly detection, signature matching and connection analysis. Zeek has been designed so that it is easy to add additional analyzers.
1.4 Signatures

Most network intrusion detection systems (NIDS) match a large set of signatures against the network traffic. Here, a signature is a pattern of bytes that the NIDS tries to locate in the payload of network packets. As soon as a match is found, the system generates an alert.

A well-known IDS system is Snort; conversely, Zeek’s general approach to intrusion detection has a much broader scope than traditional signature-matching, yet still contains a signature engine providing a functionality that is similar to that of other systems. Furthermore, while Zeek implements its own flexible signature language, there exists a converter which directly translates Snort’s signatures into Zeek’s syntax, as shown below:

<table>
<thead>
<tr>
<th>(a) Snort</th>
<th>(b) Bro</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert tcp any any -&gt; [a.b.0.0/16,c.d.e.0/24] 80</td>
<td></td>
</tr>
<tr>
<td>( msg:&quot;WEB-ATTACKS conf/httpd.conf attempt&quot;; nocode; sid:1373; flow:to_server,established; content:&quot;conf/httpd.conf&quot;; [...] )</td>
<td></td>
</tr>
<tr>
<td>signature sid-1373 {</td>
<td></td>
</tr>
<tr>
<td>ip-proto == tcp</td>
<td></td>
</tr>
<tr>
<td>dst-ip == a.b.0.0/16,c.d.e.0/24</td>
<td></td>
</tr>
<tr>
<td>dst-port == 80</td>
<td></td>
</tr>
<tr>
<td># The payload below is actually generated in a</td>
<td></td>
</tr>
<tr>
<td># case-insensitive format, which we omit here</td>
<td></td>
</tr>
<tr>
<td># for clarity.</td>
<td></td>
</tr>
<tr>
<td>payload /*.conf\httpd.conf/</td>
<td></td>
</tr>
<tr>
<td>tcp-state established,originator</td>
<td></td>
</tr>
<tr>
<td>event &quot;WEB-ATTACKS conf/httpd.conf attempt&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3. Example of signature conversion. (a) Snort’s signature. (b) Bro’s signature.

1.5 ZeekControl

ZeekControl, formerly known as BroControl, is an interactive shell for easily operating and managing Zeek installations on a single system or across multiple systems in a traffic-monitoring cluster.
2 Using ZeekControl to update the status of Zeek

Step 1. From the top of the screen, click on the Bro2 button as shown below to enter the Bro2 machine.

Step 2. The Bro2 machine will now open, and the desktop will be displayed. On the left side of the screen, click on the Terminal icon as shown below.
Step 3. Using the Terminal, input the following command to enter the ZeekControl directory. To type capital letters, it is recommended to hold the Shift key while typing rather than using the Caps key.

```
cd $ZEEK_INSTALL/bin/
```

The active directory will change, as seen on the second line of the Terminal. Note that $ZEEK_INSTALL variable was substituted by its value (/usr/local/zeek) listed in Table 2.

Step 4. Use the following command to view the contents of the active directory.

```
ls
```

The directory contents will be displayed. The green file name portrays an executable file.

Step 5. Use the following command to launch the ZeekControl tool. When prompted for a password, type password and hit Enter.

```
sudo ./zeekctl
```

Once active, ZeekControl prompt will be displayed within the Terminal. The help command will display additional information regarding ZeekControl.

2.1 Starting a new instance of Zeek
Lab 1: Introduction to the Capabilities of Zeek

Step 1. To initialize Zeek, enter the following command into the ZeekControl prompt.

```
start
```

Step 2. Use the following command to view the status of the currently active Zeek instance to ensure that it is active.

```
status
```

The running status indicates that Zeek is currently active and functioning properly. The output of the status command includes other useful parameters:

- **Name**: the name of the Zeek instance.
- **Type**: the type of the instance (standalone in our case).
- **Host**: the hostname (localhost).
- **Pid**: the process ID. This ID can be used with other tools like `kill` to send a signal to the process.
- **Started**: the starting date and time of the instance.

2.2 Stopping the active instance of Zeek

Step 1. To stop Zeek, enter the following command into the ZeekControl prompt.

```
stop
```

Step 2. Use the following command to verify the exit status of Zeek.

```
status
```
The stopped status indicates that Zeek is currently stopped.

**Step 3.** To restart Zeek, enter the following command into the ZeekControl prompt.

```
start
```

**Step 4.** Use the following command to exit ZeekControl.

```
exit
```

Note that exiting the ZeekControl tool does not stop Zeek. Zeek is only stopped by explicitly using the `stop` command in the ZeekControl prompt.

3 Introduction to Zeek’s traffic analysis capabilities

Zeek's broad range of traffic analysis capabilities makes it an exceptional intrusion detection system (IDS) and network analysis framework. Zeek is proficient in processing packet capture (pcap) files and logging traffic on a given network interface.

3.1 Processing offline packet capture files

Linux-based systems process packet capture (pcap) files using the `libpcap` library. In Zeek, it is possible to capture live traffic and analyze trace files. In the following example, we analyze a pcap file using a premade script that detects brute force attacks.

3.1.1 Command format for processing packet capture files
The general format for initializing offline packet capture analysis is as follows:

```
$ zeek -r <pcap_file_location> <script_location>
```

- **zeek**: command to invoke Zeek.
- `-r`: option signifies to Zeek that it will be reading from an offline file.
- `<pcap_file_location>`: indicates the pcap file location.
- `<script_location>`: indicates the script location.

### 3.1.2 Leveraging a script to detect brute force attacks present in a pcap file

Zeek installs a number of default scripts and trace files that can be used for testing purposes. In this section, we use the `bruteforce.pcap` as the input packet capture file and `ZeekBruteforceDetection.zeek` as the detection script. The packet capture file contains network traffic of a brute force password attack, while the script defines the brute forcing event for the Zeek event engine.

**Step 1.** Enter the lab workspace directory. To type capital letters, it is recommended to hold the Shift key while typing rather than using the Caps key.

```
$ cd $ZEEK_LABS
```

**Step 2.** Initialize Zeek offline packet parsing on the packet capture file. Use the Tab key for autocompletion. To type capital letters, it is recommended to hold the Shift key while typing rather than using the Caps key.

```
$ zeek -r Sample-Captures/bruteforce.pcap ZeekBruteforceDetection.zeek
```

**Step 3.** After running the command, if a brute forcing attack was found, it will be logged in the `notice.log` output log file. We will use the `cat` command to view the file.

```
$ cat notice.log
```
Examining the proceeding image, brute forcing was detected. The log shows that 20 login attempts failed on an FTP server with an IP 192.168.56.1.

3.2 Generating and analyzing live network traffic capture

The Tcpdump command utility is a famous network packet analyzing tool that is used to display TCP/IP and other network packets being transmitted over the network.  

3.2.1 Leveraging the Tcpdump utility for capturing live network traffic

The general format for `tcpdump` is the following:

```
sudo tcpdump -i <interface_name> -s <num> -w <pcap_file_location>
```

- `sudo`: option to enable higher level privileges.
- `tcpdump`: program for capturing live network traffic.
- `-i`: option used to specify a network interface.
- `<interface_name>`: denotes the interface name.
- `-s`: option used to specify number of packets to capture.
- `<num>`: denotes the number of packets to capture. 0 equals infinite.
- `-w`: option used to specify that we will be writing to a new file.
- `<pcap_file_location>`: indicates the file location.

3.2.2 Capturing live network traffic

`Bro2` machine’s `ens33` interface is used to record sample network traffic.
Lab 1: Introduction to the Capabilities of Zeek

**Step 1.** Enter the TCP-Traffic directory by using the `cd` command.

```
    cd TCP-Traffic/
```

**Step 2.** Use the following command to begin live packet capture. When prompted for a password, type `password` and hit `Enter`. If the Terminal session has not been terminated or closed, you may not be prompted to enter the password. Live packet capture will start on interface ens33.

```
    sudo tcpdump -i ens33 -s 0 -w ntraffic.pcap
```

**Step 3.** Click on *File*, then *New Tab* to open a new tab within the same terminal window.

**Step 4.** Generate traffic by using the `ping` utility. Ping operates by sending Internet Control Message Protocol (ICMP) echo request packets to the target host and waiting for an ICMP echo reply. Issue the following command on the newly opened tab to conduct a ping on the Bro1 machine. When prompted for a password, type `password` and hit `Enter`.

```
    sudo ping -c 3 192.168.1.2
```
The \texttt{c} option is used to indicate the number of packets to send – in this example, 3 packets.

**Step 5.** Re-open the first Terminal tab as shown in the figure below.

**Step 6.** Use the \texttt{Ctrl+c} key combination to stop live traffic capture. Statistics of the capture session will be displayed. 10 packets were recorded by the interface, while 6 were captured and stored in the new \texttt{ntraffic.pcap} file.

### 3.2.3 Analyzing the newly captured network traffic

**Step 1:** A new file \texttt{ntraffic.pcap} was generated after using the \texttt{tcpdump} tool. Use the \texttt{ls} command to list the current directory’s contents.
Step 2: Initialize Zeek offline packet parsing on the packet capture file. The `-r` option is used to read from a given pcap file, and the `-C` option is for disabling checksums validation.

```
zeek -C -r ntraffic.pcap
```

Step 3: Open the `conn.log` file in a text editor. The following command uses `nano` text editor to open the file for editing. Unlike the `cat` tool which displays plain text, `nano` displays formatted log files, which makes it easier to visualize the output.

```
nano conn.log
```

By default, Zeek will generate tab delimited columns for packet information, with each row corresponding to an individual connection. It may be necessary to maximize the Terminal window to properly view the following IP address.

A sample row is highlighted in the image above, displaying the `Bro2` machine’s IP address (192.168.2.2) and the `Bro1` machine’s IP address (192.168.1.2). The line shows that the protocol is ICMP since we used the `ping` tool. Note that the figure displays partial output.
The user can scroll to the right to see more fields. Use the `Ctrl+x` shortcut combination to exit the text editor.

**Step 4:** Stop Zeek by entering the following command on the terminal. If necessary, type `password` as the password. If the Terminal session has not been terminated or closed, you may not be prompted to enter a password. To type capital letters, it is recommended to hold the `Shift` key while typing rather than using the `Caps` key.

```
cd $ZEEK_INSTALL/bin && sudo ./zeekctl stop
```

The above command navigates to Zeek’s installation directory and executes the stop command in `zeekctl`.

Concluding this lab, we have reviewed the Zeek (Bro)’s architecture and event-based engine, as well as introduced both offline and live network traffic capture.

**References**