

# The University of Texas at San Antonio<sup>™</sup>

The Cyber Center for Security and Analytics



# ZEEK INSTRUSION DETECTION SERIES

# Lab 9: Profiling and Performance Metrics of Zeek

Document Version: 03-13-2020



Award 1829698 "CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

# Contents

Overview				
bjectives				
.ab topology3				
.ab settings				
Lab roadmap4				
I Introduction to Zeek profiling 4				
2 Generating customized malicious network traffic5				
2.1 Starting a new instance of Zeek5				
2.2 Launching Mininet				
2.3 Setting up the zeek2 virtual machine for live network capture				
2.4 Using the zeek1 virtual machine for network scanning activities				
2.4.1 Terminating live network capture10				
3 Generating and viewing Zeek profiling log files11				
3.1 Applying the profiling filter 11				
14 Implementing tools to test Zeek's performance				
4.1 Using sysstat sar utility14				
4.2 Using the top utility 16				
4.3 Viewing the resource consumption of Zeek				
4.4 Closing the current instance of Zeek17				
References				

# **Overview**

With Zeek's event-based framework, anomalies can be detected, processed and analyzed with external software. In this lab, we explain Zeek's profiling log stream and Zeek's resource consumption.

### **Objectives**

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

- 1. Enable Zeek's profiling log stream for session-based statistics.
- 2. Generate customized traffic to be captured by Zeek's profiling.
- 3. Implement tools necessary for testing Zeek's resource consumption.

# Lab topology

Figure 1 shows the lab topology. The topology uses 10.0.0.0/8 which is the default network assigned by Mininet. The *zeek1* and *zeek2* virtual machines will be used to generate and collect network traffic.



# Lab settings

The information (case-sensitive) in the table below provides the credentials necessary to access the machines used in this lab.

Device	Account	Password		
Client	admin	password		

Variable Name	Absolute Path
\$ZEEK_INSTALL	/usr/local/zeek
\$ZEEK_TESTING_TRACES	/home/zeek/zeek/testing/btest/Traces
\$ZEEK_PROTOCOLS_SCRIPT	/home/zeek/zeek/scripts/policy/protocols

Table 2. Shell variables and their corresponding absolute paths.

#### Lab roadmap

This lab is organized as follows:

- 1. Section 1: Introduction to Zeek profiling.
- 2. Section 2: Generating customized malicious network traffic.
- 3. Section 3: Generating and viewing Zeek profiling log files.
- 4. Section 4: Implementing tools to test Zeek's performance.

# 1 Introduction to Zeek profiling

Zeek includes the option of enabling profiling. When profiling is enabled, a new log stream will be created to store session-related statistics. The Profile log file will contain a large variety of information, including but not limited to running time, memory usage, connection information and packet protocol statistics.

To enable profiling while using Zeek for offline packet capture file processing, you will need to implement the following functionality in a Zeek script.

```
1 module Profiling
2
3 redef profiling_file = open (fmt(<filename>, <logstream>));
4 redef profiling_interval = 3 secs;
5 redef expensive_profiling_multiple = 5;
6
7 vevent zeek_init() {
8 set_buf(profiling_file, F);
9 }
```

The script is explained as follows. Each number represents the respective line number:

- 1. Sets the module workspace as *Profiling*.
- 3. Specifies the name of the new profiling log file, as well as determines the format based off an input log stream.
- 4. Specifies the time interval for Zeek to record empirical information. In this example the time interval is 3 seconds.

- 5. Specifies the number of profiling intervals defined in Line 5. In this example, the profiling interval is 5 instances.
- 7. Initialization event.
- 8. Appends the new log stream information.
- 9. End of initialization event.

Profiling is enabled by calling the Zeek script during packet processing, as reviewed in the previous labs.

zeek -r <packet capture file> <Profiling Script>

- cpacket capture file>: denotes the input packet capture file.
- <profiling Script>: denotes the Zeek script to be run during packet processing.

In the following section we generate customized malicious traffic to be viewed within a Zeek profiling log.

# 2 Generating customized malicious network traffic

This section introduces creating and using a Zeek profiling script, which will enable session-based statistics for Zeek packet capture file processing.

# 2.1 Starting a new instance of Zeek

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



**Step 2.** The *Client* machine will now open, and the desktop will be displayed. On the left side of the screen, click on the LXTerminal icon as shown below.



**Step 3.** Start Zeek by entering the following command on the terminal. This command enters Zeek's default installation directory and invokes Zeekctl tool to start a new instance. To type capital letters, it is recommended to hold the Shift key while typing rather than using the Caps key. When prompted for a password, type password and hit Enter.



A new instance of Zeek is now active, and we are ready to proceed to the next section of the lab.

#### 2.2 Launching Mininet

**Step 1.** From the *Client* machine's desktop, on the left side of the screen, click on the MiniEdit icon as shown below. When prompted for a password, type password and hit Enter. The MiniEdit editor will now launch.



**Step 2.** The MiniEdit editor will now launch and allow for the creation of new, virtualized lab topologies. Load the correct topology by clicking the Open button within the File tab on the top left of the MiniEdit editor.

File Edit Run Help
New
Open
Save
Export Level 2 Script
Quit

**Step 3.** Navigate to the Zeek-Topologies directory by scrolling to the right of the active directories and double clicking the Zeek-Topolgies icon, or by clicking the Open button.

	Ope	n	- + x
Directory:	/home/zeek		-
<ul> <li>.presage</li> <li>.thumbna</li> <li>.wireshark</li> <li>Desktop</li> <li>Document</li> <li>Download</li> </ul>	ils in oflops c in oftest is openflow c in openflow c in pox c in public	i zeek Zeek Zeek	c k-Labs k-Topologies
1			
File <u>n</u> ame	:		<u>O</u> pen
Files of type	: Mininet Topolog	y (*.mn) 🔤	<u>C</u> ancel

**Step 4.** Select the *Topology.mn* file by double clicking the *Topolgies.mn* icon, or by clicking the Open button.

	Open	- + ×
Directory:	/home/zeek/Zeek-Topologies	-
Topology	/.mn	
•		Þ
File <u>n</u> ame	:	<u>O</u> pen
Files of <u>type</u>	: Mininet Topology (*.mn) 😐	<u>C</u> ancel

**Step 5.** To begin running the virtual machines, navigate to the  $\underline{Run}$  button, found on the bottom left of the Miniedit editor, and select the  $\underline{Run}$  button, as seen in the image below.



#### 2.3 Setting up the zeek2 virtual machine for live network capture

**Step 1.** Launch the *zeek2* terminal by holding the right mouse button on the desired machine and clicking the Terminal button.



#### **Step 2.** Navigate to the TCP-Traffic directory.

cd Zeek-Labs/TCP-T	caffic/			
X	"Host: zeek2"	-	+	×
root@admin:~# <mark>cd</mark> root@admin:~/Zee	Zeek-Labs/TCP-Traffic/ k-Labs/TCP-Traffic#			

**Step 3.** Start live packet capture on interface *zeek2-eth0* and save the output to a file named *scantraffic.pcap*.

tcpdump -i zeek2-eth0 -s 0 -w scantraffic.pcap



The *zeek2* virtual machine is now ready to begin collecting live network traffic. Next, we will use the *zeek1* machine to generate scan-based network traffic.

#### 2.4 Using the zeek1 virtual machine for network scanning activities

In this section we use the mmap software to generate TCP-based scan traffic.

This section introduces two new options for the mmap software.

- <u>f</u>: specifies to send packet fragments. By fragmenting packets, a scanner can attempt to bypass firewalls that check for entire packet signatures.
- <u>-mtu <num</u>: specifies the max number of bytes to be sent in a fragmented packet. The number variable must be a multiple of 8.

**Step 1.** Minimize the *zeek2* Terminal and open the *zeek1* Terminal by following the previous steps. If necessary, right click within the Miniedit editor to activate your cursor.



**Step 2.** Launch a fragmented TCP SYN scan against the *zeek2* machine.

```
nmap -sS -f 10.0.0.2
```



**Step 3.** Launch a fragmented TCP SYN scan with a packet size of 8 bytes against the *zeek2* machine.

```
nmap -sS -mtu 8 10.0.0.2
```



**Step 4.** Launch a fragmented TCP SYN scan with a packet size of 64 bytes against the *zeek2* machine.



#### 2.4.1 Terminating live network capture

**Step 1.** Minimize the *zeek1 Terminal* and open the *zeek2 Terminal* using the navigation bar at the bottom of the screen. If necessary, right click within the Miniedit editor to activate your cursor.

$\overline{\mathbf{A}}$	🚞 🎦 💼 🔤 sudo	MiniEdit	X ["Host: zeek1"]	T["Host: zeek2"]
-------------------------	--------------	----------	-------------------	------------------

**Step 2**. Use the <u>Ctrl+c</u> key combination to stop live traffic capture. Statistics of the capture session will we be displayed. 10,033 packets were recorded by the interface, which were then captured and stored in the new *scantraffic.pcap* file.



**Step 3.** Stop the current Mininet session by clicking the Stop button on the bottom left of the MiniEdit editor, and close the MiniEdit editor by clicking the R on the top right of the editor.



# 3 Generating and viewing Zeek profiling log files

Now that we have collected fragmented traffic, we can begin processing the packet capture file with Zeek.

# 3.1 Applying the profiling filter

**Step 1.** On the left side of the *Client* desktop, click on the LXTerminal icon as shown below.



cd Zeek-Labs/Lab-Scripts/

zeek@admin: ~/Zeek-Labs/Lab-Scripts	- + ×
File Edit Tabs Help	
zeek@admin:~\$ cd Zeek-Labs/Lab-Scripts/ zeek@admin:~/Zeek-Labs/Lab-Scripts\$	

**Step 3.** View the *EnableProfiling.zeek* Zeek script.

```
nl EnableProfiling.zeek
```

		zeek@admin: ~/Zeek-Labs/Lab-Scripts	⊦×
File	Edit	Tabs Help	
zeel	@adr 1	n <b>in:~/Zeek-Labs/Lab-Scripts</b> \$ <mark>nl EnableProfiling.zeek</mark> module Profiling;	
	2 3	function log_suffix(): string {	
	4	local rprof = getenv("ZEEK_LOG_SUFFIX");	
	5	if ( rprof == "" )	
	6	return "log";	
	7	return rprof;	
	8	}	
	9	<pre>redef profiling_file = open(fmt("prof.%s", Profiling::log_suffix()))</pre>	;
	10	<pre>redef profiling_interval = 15 secs;</pre>	
	11	<pre>redef expensive_profiling_multiple = 20;</pre>	
	12	<pre>event zeek_init()</pre>	
	13		
	14	<pre>set_buf(profiling_file, F);</pre>	
	15	}	
zeel	@adr	nin:~/Zeek-Labs/Lab-Scripts\$	

Similar to the example in the introduction, the *EnableProfiling.zeek* Zeek script is used to create a new log file named *Statistics.log* containing Zeek profiling statistics. The script enables the intervals to be 15 seconds apart, with 20 total intervals.

**Step 4.** Navigate to the TCP-Traffic directory.



**Step 5.** Process the *ntraffic.pcap* packet capture file.

zeek -C -r scantraffic.pcap ../Lab-Scripts/EnableProfiling.zeek

	Zeek@admin: ~/Zeek-Labs/TCP-Traffic						+ ×				
File	Edit	Tabs	Help								
zeek	(@adr	nin:-	-/Zeek-Labs	/TCP-Traffic\$	zeek	-C -r	scantraf	fic.pcap	/Lab-	Scri	pts
/Ena	able	Prof	iling.zeek								
zeek	(@adr	nin:-	-/Zeek-Labs	/TCP-Traffic\$							

**Step 6.** Display the contents of the *prof.log* file.

nano prof.log

zeek@admin: ~/Zeek-Labs/TCP-Traffic	- + ×
File Edit Tabs Help	
zeek@admin:~/Zeek-Labs/TCP-Traffic\$ nano prof.log zeek@admin:~/Zeek-Labs/TCP-Traffic\$	

The *prof.log* file will be displayed.

	zeek@admin	~/Zeek	Labs/TCF	P-Traffic			+ ×
Tabs Help							
10 2.9.3		prof	.log				
Command line: Memory: total= Run-time: user Conns: total=0 Conns: tcp=0/0 TCP-States: TCP-States:Sr TCP-States:SA TCP-States:SA TCP-States:Par TCP-States:Fir TCP-States:Fir TCP-States:Rst Connections ex Total reassemt	zeek -C -r s =89364K total +sys=0.0 use ) current=0/0 ) udp=0/0 icm Inact. act.	cantraf _adj=0K r=0.0 s ext=0 p=0/0 Syn. inacti	fic.pca malloc ys=0.0 mem=0K SA Vity: 0	p/Lab- ed: 66980 real=0.0 avg=0.0 t Part.	Scripts, K able=0K Est.	/EnablePr connvals Fin.	ofi\$ =0K R\$ \$ \$ \$ \$ \$ \$
	Tabs Help To 2.9.3 Command line: Memory: total= Run-time: user Conns: total=0 Conns: tcp=0/0 TCP-States: TCP-States:SA TCP-States:SA TCP-States:SA TCP-States:Fir TCP-States:Fir TCP-States:Fir TCP-States:Rst Connections ex Total reassemb Timers: currer	Zeek@adminFabs HelpTo 2.9.3Command line: zeek -C -r sMemory: total=89364K totalRun-time: user+sys=0.0 useConns: total=0 current=0/0Conns: tcp=0/0 udp=0/0 icmTCP-States: Inact.TCP-States: Inact.TCP-States:SATCP-States:Part.TCP-States:Fin.TCP-States:Fin.TCP-States:Rst.Connections expired due toTotal reassembler data: 0KTimers: current=30 max=30	zeek@admin: ~/Zeek@Fabs Helpno 2.9.3profCommand line: zeek -C -r scantrafMemory: total=89364K total_adj=0KRun-time: user+sys=0.0 user=0.0 sConns: total=0 current=0/0 ext=0Conns: tcp=0/0 udp=0/0 icmp=0/0TCP-States:Inact. Syn.TCP-States:SATCP-States:Part.TCP-States:Fin.TCP-States:Fin.TCP-States:Rst.Connections expired due to inactiTotal reassembler data: 0KTimers: current=30 max=30 mem=1K	zeek@admin: ~/Zeek-Labs/TCFFabs Helpno 2.9.3prof.logCommand line: zeek -C -r scantraffic.pcaMemory: total=89364K total_adj=0K mallocRun-time: user+sys=0.0 user=0.0 sys=0.0Conns: total=0 current=0/0 ext=0 mem=0KConns: total=0 current=0/0 ext=0 mem=0KConns: tcp=0/0 udp=0/0 icmp=0/0TCP-States:Inact. Syn.SATCP-States:Syn.TCP-States:Part.TCP-States:Fin.TCP-States:Fin.TCP-States:Fin.TCP-States:Rst.Connections expired due to inactivity: 0Total reassembler data: 0KTimers: current=30 max=30 mem=1K lag=0.0	zeek@admin: ~/Zeek-Labs/TCP-TrafficFabs Helpno 2.9.3prof.logCommand line: zeek -C -r scantraffic.pcap/Lab-3Memory: total=89364K total_adj=0K malloced: 669800Run-time: user+sys=0.0 user=0.0 sys=0.0 real=0.0Conns: total=0 current=0/0 ext=0 mem=0K avg=0.0 toConns: total=0 current=0/0 ext=0 mem=0K avg=0.0 toConns: tcp=0/0 udp=0/0 icmp=0/0TCP-States:Inact. Syn. SA Part.TCP-States:Inact.TCP-States:SATCP-States:Part.TCP-States:Fin.TCP-States:Fin.TCP-States:Rst.Connections expired due to inactivity: 0Total reassembler data: 0KTimers: current=30 max=30 mem=1K lag=0.00s	zeek@admin: ~/Zeek-Labs/TCP-TrafficFabs Helpno 2.9.3prof.logCommand line: zeek -C -r scantraffic.pcap/Lab-Scripts/Memory: total=89364K total_adj=0K malloced: 66980KRun-time: user+sys=0.0 user=0.0 sys=0.0 real=0.0Conns: total=0 current=0/0 ext=0 mem=0K avg=0.0 table=0KConns: tcp=0/0 udp=0/0 icmp=0/0TCP-States:Inact. Syn.SAPart.Est.TCP-States:SATCP-States:Fin.TCP-States:Fin.TCP-States:Fin.TCP-States:Rst.Connections expired due to inactivity: 0Total reassembler data: 0KTimers: current=30 max=30 mem=1K lag=0.00s	zeek@admin: ~/Zeek-Labs/TCP-TrafficFabs HelpTo 2.9.3prof.logCommand line: zeek -C -r scantraffic.pcap/Lab-Scripts/EnablePrMemory: total=89364K total_adj=0K malloced: 66980KRun-time: user+sys=0.0 user=0.0 sys=0.0 real=0.0Conns: total=0 current=0/0 ext=0 mem=0K avg=0.0 table=0K connvalsConns: tcp=0/0 udp=0/0 icmp=0/0TCP-States:Inact. Syn. SATCP-States:Syn.TCP-States:SATCP-States:Part.TCP-States:Fin.TCP-States:Fin.TCP-States:Fin.TCP-States:Rst.Connections expired due to inactivity: 0Total reassembler data: 0KTimers: current=30 max=30 mem=1K lag=0.00s

Viewing the *Statistics.log* file, each profiling\_interval will be displayed between a line separator made by dashes \_\_\_\_.

Within the *prof.log* file, we can see the total memory used while processing the packet capture file, the Run-time, as well as a number of TCP flags, connections and *Triggers*. Within the first iteration of profiling\_interval we see that no TCP packet flags have been recorded.

**Step 7.** Go to the next iteration of profiling\_interval within the *Statistics.log* file.

Sec. 1	zeek@admin: ~/Zeek-Labs/TCP-Traffic - + ×
File Edit Tabs Help	
GNU nano 2.9.3	prof.log
1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306	<pre>Memory: total=89364K total_adj=0K malloced: 66984K Run-time: user+sys=0.0 user=0.0 sys=0.0 real=0.0 Conns: total=0 current=0/0 ext=0 mem=0K avg=0.0 table=0K co\$ Conns: tcp=0/0 udp=0/0 icmp=0/0 TCP-States: Inact. Syn. SA Part. Est. \$ TCP-States:Inact. \$ TCP-States:Syn. \$ TCP-States:SA \$ TCP-States:Part. \$ TCP-States:Est. \$ TCP-States:Fin. \$ TCP-States:Rst. \$ \$ TCP-States:Rst. \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$</pre>
1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 1578948752.164306 ^G Get Help ^0 ^X Exit ^R	Connections expired due to inactivity: 0 Total reassembler data: 0K Timers: current=27 max=30 mem=1K lag=1578948751.16s DNS_Mgr: requests=0 succesful=0 failed=0 pending=0 cached_h\$ Triggers: total=0 pending=0 ScheduleTimer = 2 Write Out ^W Where Is ^K Cut Text ^J Justify Read File ^\ Replace ^U Uncut Text ^T To Spell

By scrolling through the *prof.log* file, we can see information found in the next iteration of a profiling\_interval. We can see the total number of *TCP-States:Syn* has updated multiple parameters, with additional *Triggers* being been included. This includes the total memory usage, displayed towards the bottom of the image.

Zeek profiling is a great tool for generating more detailed session-based statistics while processing packet capture files with Zeek.

**Step 8.** Press Ctrl + x to exit out the nano.

# 4 Implementing tools to test Zeek's performance

While Zeek profiling will display the resulting statistics after processing a packet capture file, it is important to monitor Zeek resource consumption during network traffic analysis.

A number of Linux-based software utilities can be used to track system resource consumption in real time.

# 4.1 Using sysstat sar utility

The sar command can be used to display a number of system resources over specific time intervals. The following steps will highlight the ways to enable sar resource tracking.

**Step 1.** Launch the sar utility to track CPU consumption.

- sar: calls the sar utility, belonging to the syystat packages.
- 2: indicates each iteration of CPU statistics is separated by a 2 second time interval.
- 30: indicates that a total of 30 iterations of CPU statistics should be displayed.

			zeek@ad	min: ~/Ze	ek-Labs/TCP-1	Traffic		- + ×
File Edit	Tabs Help	) ::						
zeek@adm: Linux 4.: )	in:~/Ze 15.0-70	<b>ek-Labs/</b> -generic	(admin)	ic\$ <mark>sar</mark>	2 30 01/13/2020	_x86_	64_	(8 CPU
04:08:37	PM	CPU	%user	%nice	%system	%iowait	%steal	%idl
e 04:08:39 1	PM	all	0.13	0.00	0.06	0.00		99.8
04:08:41	PM	all		0.00	0.00			100.0
04:08:43 8	PM	all	0.06	0.00	0.06	0.00		99.8
04:08:45 4	PM	all	0.06	0.00		0.00		99.9
04:08:47 0	PM	all		0.00	0.00	0.00	0.00	100.0

Use the CTRL + C keyboard combination to terminate the sar utility and return to the terminal.

**Step 2.** Launch the sar utility to track memory consumption.

sar -r 3 25

- sar: calls the sar utility, belonging to the syystat packages.
- <u>-r</u>: indicates memory consumption in kilobytes.
- 3: indicates each iteration of memory statistics is seperated by a 3 second time interval.
- 25: indicates that a total of 25 iterations of memory statistics should be displayed.

			zeek@admin: ~/Zeek-Labs/TCP-Traffic						
File Ed	it Tabs H	elp							
zeek@a	dmin:~/	Zeek-Labs/	TCP-Traffi	c\$ sar -r	3 25		3 <b>4</b>		
)	4.15.0-	/⊍-generic	(admin)	01/	13/2020	_x86_6	4_	(8 CPU	
04:09:	04 PM kl	bmemfree	kbavail k	bmemused	%memused	kbbuffers	kbcached	kbcom	
m1t 04:09:	%commit 07 PM	7266672	7496160	792116	9.83	92692	332116	1172	
688 04 · 09 ·	12.99	432692	121120	48	9.83	92692	332116	1172	
688	12.99	432692	121120	48	3.05	JLOJL	552220		
04:09: 688	13 PM 12.99	7266672 432692	7496160 121120	792116	9.83	92692	332116	1172	
688	12.99	432692	121120	(		52052			

Use the CTRL + C keyboard combination to terminate the sar utility and return to the terminal.

#### 4.2 Using the top utility

Alternative to the syystat sar utility, the top utility can be used to display the resource consumption of every active process.

**Step 1.** Launch the top utility to track resource consumption.

top -i

- top: calls the top utility.
- -i: toggles idle processes off, so that only active processes will be displayed.

Zeek@a	admin: ~ - + ×
File Edit Tabs Help	
zeek@admin:∼\$ top -i	

After entering the command, the Terminal will display the resource consumption.

Each row will belong to a unique process and display the related CPU and memory resource usage.

	zeek@admin: ~/Zeek-Labs/TCP-Traffic	- + ×
File Edit Tabs Help		
top - 16:09:37 up	2:53, 1 user, load average: 0.00, 0.00, 0.00	
Tasks: <b>174</b> total,	1 running, 112 sleeping, 0 stopped, 0 zombie	
%Cpu(s): 0.0 us,	0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si,	0.0 s
KiB Mem : 8058788	total, 7265812 free, 293128 used, 499848 buff/cac	he
KiB Swap: <b>969960</b>	total, 969960 free, 0 used. 7495516 avail Mer	m
PID USER PR	NI VIRT RES SHR S %CPU %MEM TIME+ COMMA	ND
765 root 20	0 360328 56920 32028 S 0.3 0.7 0:10.61 Xorg	

#### 4.3 Viewing the resource consumption of Zeek

Step 1. Using the File drop down options, create a New Tab within the Terminal.

2		zeek@admin: ~/Zeek-Labs/TCP-Traffic	- + ×
File Edit Tabs H	Ielp		
+ New Window	Shift+Ctrl+N	1 user, load average: 0.00, 0.00, 0.00	
+ New Tab	Shift+Ctrl+T	nning, <b>112</b> sleeping, <b>0</b> stopped, <b>0</b> zombie	0.0.0
× Close Tab	Shift+Ctrl+W	y, 0.0 hi, 99.7 id, 0.0 wa, 0.0 hi, 0.0 si, . 7266308 free. 292624 used. 499856 buff/cad	the s
🕞 Close Window	Shift+Ctrl+Q	, 969960 free, 0 used. 7496028 avail Me	em

**Step 2.** In the second tab, begin packet capture file processing of the *bigFlows.pcap* file using Zeek.

zeek -C -r	/Sample-PCAP/big	gFlows.pcap	
File Edit Tabs He	zeek@	admin: ~/Zeek-Labs/TCP-Traffic -	+ ×
* zeek@ad X	zeek@admi 🗙		
zeek@admin:~/Z zeek@admin:~/Z	eek-Labs/TCP-Tra eek-Labs/TCP-Tra	affic\$ zeek -C -r/Sample-PCAP/bigFlows.pcap affic\$	

Step 3. Return to the first Terminal tab and view the active processes.

2		z	eek@adm	in: ~/Zeel	k-Labs/1	CP-Tra	ffic			+ ×
File Edit Ta	bs Help									
zeek@admi	. × zee	k@admi	×							
top - 16:1 Tasks: <b>176</b> %Cpu(s): KiB Mem : KiB Swap:	l1:25 up 5 total, 2.5 us, 8058788 969960	2:54, <b>2</b> run <b>0.5</b> sy total, total,	1 user, ning, 113 , 0.0 n; 6811430 969960	load a 3 sleepi 1, <b>97.0</b> 5 free, 9 free,	verage ng, 0 id, 0 3861	: 0.10 9 stop .0 wa, 84 use 0 use	), 0.03 pped, <b>0.0</b> ed, <b>8</b> ed, <b>74</b>	, 0.01 0 zombie hi, 0.0 61168 buf 01608 ava	si, <b>0.0</b> f/cache ail Mem	S
PID USER	R PR	NI	VIRT	RES	SHR S	%CPU	%MEM	TIME+	COMMAND	
2599 zeel	c 20	0 14	02392 109	9280 18	468 R	20.6	1.4	0:00.62	zeek	
765 root	20	0 3	60328 56	5920 32	028 S	1.0	0.7	0:11.00	Xorg	
2382 zeel	c 20	0 5	47208 34	1304 26	072 S	1.0	0.4	0:03.14	lxtermin	+

Use the CTRL + C keyboard combination to terminate the top utility and return to the terminal.

#### 4.4 Closing the current instance of Zeek

After you have finished the lab, it is necessary to terminate the currently active instance of Zeek. Shutting down a computer while an active instance persists will cause Zeek to shut down improperly and may cause errors in future instances.

**Step 1.** Stop Zeek by entering the following command on the terminal. If required, 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.



Concluding this lab, we introduced Zeek's profiling capabilities and generated fragmented traffic to be processed into a profiling log file. Lastly, we introduced Terminal utilities that

can be used to track Zeek's resource consumption per process. Regular checking of Zeek profiling and resource consumption is necessary to ensure the IDS is working optimally in a real-time environment.

Furthermore, we have concluded introducing Zeek's capabilities as an IDS. The remaining labs within this series will focus on further processing Zeek log files for advanced analysis.

# References

1. "Profiling", Zeek user manual, [Online], Available: https://docs.zeek.org/en/stable/scripts/policy/misc/profiling.zeek.html