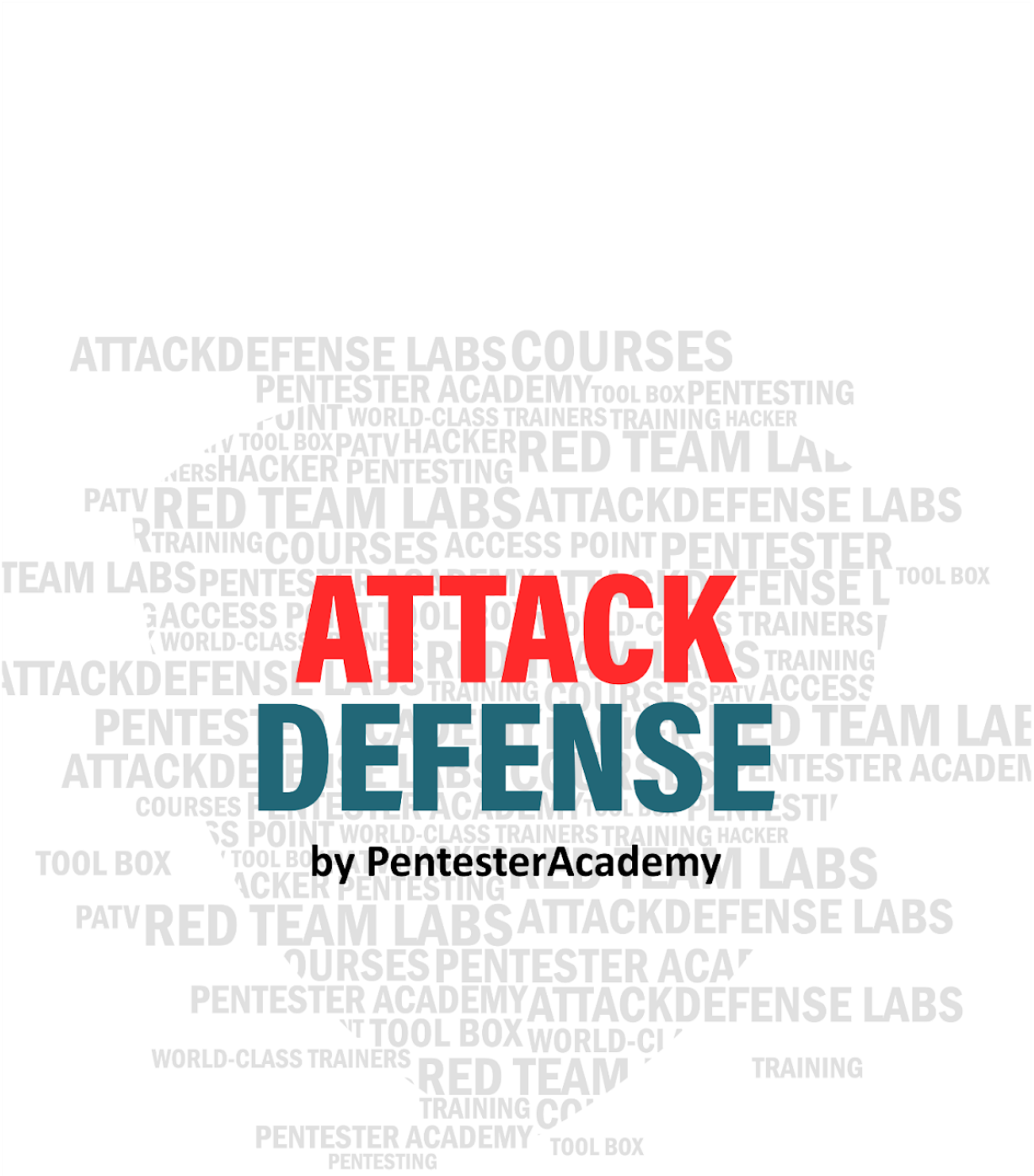

ATTACK
DEFENSE
by PentesterAcademy

Name	Backdoor Practice Lab: MIPS architecture
URL	https://attackdefense.com/challengedetails?cid=1265
Type	WiFi Attack-Defense : Backdoors

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Objective: Compile the kernel modules from the source, transfer those to the emulated router and test those out.

Solution:

Step 1: As per the challenges description, the build-system should be run as a student user. So, change to student user and switch to home directory of the student user.

Commands:
su student
cd ~

Step 2: Check the contents of the home directory of the student user.

Command: ls -l

```
student@attackdefense:~$ ls -l
total 40
drwxr-xr-x 1 student student 4096 Aug 21 19:31 Desktop
drwxr-xr-x 1 student student 4096 Aug 21 19:31 Documents
drwxr-xr-x 1 student student 4096 Aug 21 19:31 Downloads
drwxr-xr-x 1 student student 4096 Aug 21 19:31 Music
drwxr-xr-x 1 student student 4096 Aug 21 19:31 Pictures
drwxr-xr-x 1 student student 4096 Aug 21 19:31 Public
drwxr-xr-x 1 student student 4096 Aug 21 19:31 Templates
drwxr-xr-x 1 student student 4096 Aug 21 19:31 Videos
drwxr-xr-x 1 student student 4096 Oct  4 14:48 openwrt-buildsystem
drwxr-xr-x 1 student student 4096 Oct  4 14:49 rootkit-code
student@attackdefense:~$
```

Step 3: Change to build-system's directory (i.e. openwrt-buildsystem) and check the contents.

Commands:
cd openwrt-buildsystem
ls -l

```
student@attackdefense:~$ cd openwrt-buildsystem/
student@attackdefense:~/openwrt-buildsystem$
student@attackdefense:~/openwrt-buildsystem$ ls -l
total 160
-rw-r--r--  1 student student   179 Jan 30  2019 BSDmakefile
-rw-r--r--  1 student student   576 Jan 30  2019 Config.in
-rw-r--r--  1 student student 17992 Jan 30  2019 LICENSE
-rw-r--r--  1 student student  3147 Jan 30  2019 Makefile
-rw-r--r--  1 student student  1295 Jan 30  2019 README
drwxr-xr-x  4 student student  4096 Sep 19 00:33 bin
drwxr-xr-x  6 student student  4096 Sep 19 02:20 build_dir
drwxr-xr-x  2 student student  4096 Jan 30  2019 config
drwxr-xr-x  2 student student 32768 Oct  4 13:13 dl
drwxr-xr-x 10 student student  4096 Sep 18 23:50 feeds
-rw-r--r--  1 student student   395 Jan 30  2019 feeds.conf.defa
drwxr-xr-x  3 student student  4096 Jan 30  2019 include
-rw-r--r--  1 student student   176 Sep 19 00:37 key-build
-rw-r--r--  1 student student    92 Sep 19 00:37 key-build.pub
drwxr-xr-x 12 student student  4096 Sep 18 23:50 package
-rw-r--r--  1 student student 13275 Jan 30  2019 rules.mk
drwxr-xr-x  4 student student  4096 Jan 30  2019 scripts
drwxr-xr-x  8 student student  4096 Sep 19 02:23 staging_dir
drwxr-xr-x  6 student student  4096 Jan 30  2019 target
drwxr-xr-x  3 student student 12288 Sep 19 02:24 tmp
drwxr-xr-x 12 student student  4096 Jan 30  2019 toolchain
drwxr-xr-x 58 student student  4096 Jan 30  2019 tools
-rw-r--r--  1 student student    17 Jan 30  2019 version
-rw-r--r--  1 student student    11 Jan 30  2019 version.date
student@attackdefense:~/openwrt-buildsystem$
```

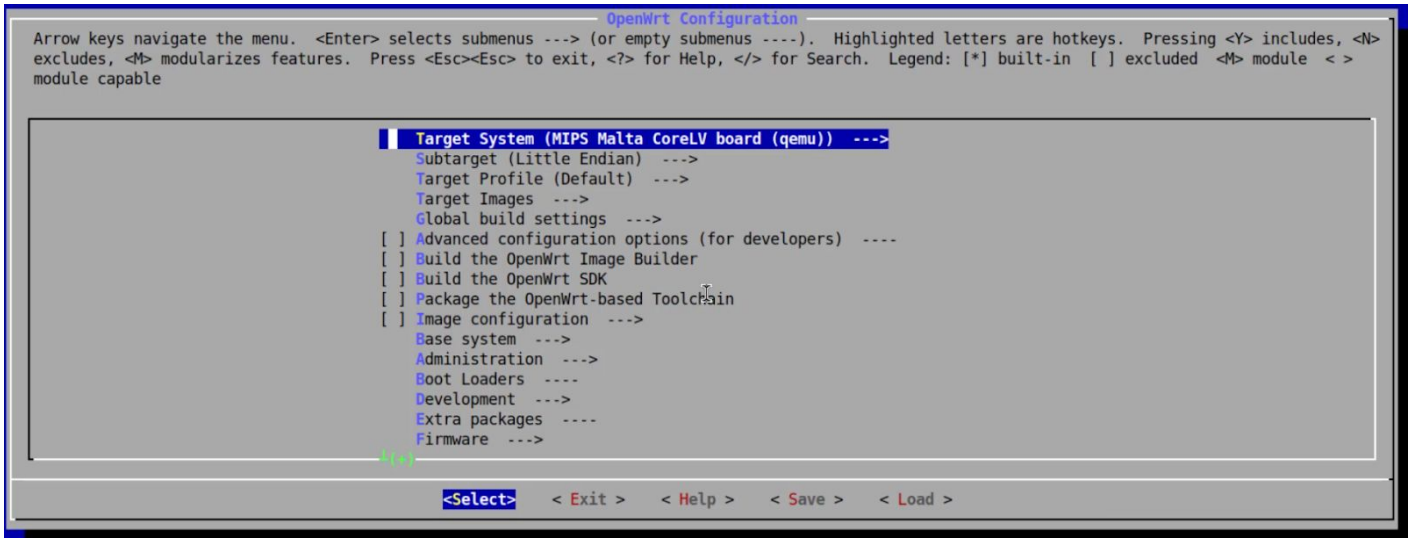
Makefile is present in the directory along with all other components.

Step 4: OpenWRT build-system allows firmware parameter selection and component customization using a ncurses based terminal UI.

Command: make menuconfig

```
student@attackdefense:~/openwrt-buildsystem$ make menuconfig


*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.
```



This TUI can be used to choose the target board and select/unselect the components.

Step 5: After making the changes, the firmware for the target board/device can be compiled using make

Command: make -j3

```
student@attackdefense:~/openwrt-buildsystem$
student@attackdefense:~/openwrt-buildsystem$ make -j3
 make[1] world
 make[2] package/cleanup
 make[2] target/compile
```

The -j3 here denotes that the compilation will be done using 3 threads. It is advisable to keep this number equal to the number of processor cores assigned to the compilation machine.

Here, compiling the firmware is NOT the main objective, hence one can cancel the compilation using Ctrl+C.

Step 6: The rootkit code is kept in the home directory of the student user (i.e. /home/student/rootkit-code). Change to rootkit code directory and check the contents.

Commands:
cd ../rootkit-code/
ls -l

```
student@attackdefense:~/openwrt-buildsystem$ cd ../rootkit-code/
student@attackdefense:~/rootkit-code$ ls -l
total 12
drwxr-xr-x 1 student student 4096 Oct  3 17:02 network
drwxr-xr-x 1 student student 4096 Oct  3 17:02 nokill
drwxr-xr-x 1 student student 4096 Oct  3 17:02 syscallmonitor
student@attackdefense:~/rootkit-code$
```

Switch to syscallmonitor directory. This directory contains kernel module code to monitor the kill syscall i.e. it will put a kernel log whenever the user will try to kill a process.

Command: cd syscallmonitor

```
student@attackdefense:~/rootkit-code$ cd syscallmonitor/
student@attackdefense:~/rootkit-code/syscallmonitor$ ls -l
total 8
-rw-r--r-- 1 student student  175 Oct  3 17:02 Makefile
-rw-r--r-- 1 student student 1825 Oct  3 17:02 killmonitor.c
student@attackdefense:~/rootkit-code/syscallmonitor$
```

Step 7: Check the contents of the Makefile.

Command: cat Makefile

```
student@attackdefense:~/rootkit-code/syscallmonitor$ cat Makefile
# Simplest possible makefile

obj-m +=  killmonitor.o


modules:
	@$(MAKE) -C $(KERNEL_ROOT) M=$(shell pwd) modules

clean:
	@$(MAKE) -C $(KERNEL_ROOT) M=$(shell pwd) clean
```

The Makefile is using the KERNEL_ROOT environment variable to point the kernel source files. Check if this environment variable is defined.

Command: echo $KERNEL_ROOT

```
student@attackdefense:~/rootkit-code/syscallmonitor$ echo $KERNEL_ROOT
/home/student/openwrt-buildsystem/build_dir/target-mipsel_24kc_musl/linux-malta_le/linux-4.14.95/
student@attackdefense:~/rootkit-code/syscallmonitor$
```

Also check the alias defined in the setup.

Command: alias

```
student@attackdefense:~/rootkit-code/syscallmonitor$ alias
alias mipsmake='ARCH=mips CROSS_COMPILE=mipsel-openwrt-linux- make'
student@attackdefense:~/rootkit-code/syscallmonitor$
```

Alias mipsmake is set which essentially contains a command to cross compile the code for MIPSEL OpenWRT.

Step 8: Build the kernel module by running the Makefile (triggered from mipsmake).

Command: mipsmake

```
student@attackdefense:~/rootkit-code/syscallmonitor$ mipsmake
make[1]: Entering directory '/home/student/openwrt-buildsystem/build_dir/target-mipsel_24kc_musl/linux-malta_le/linux-4.14.95'
  CC [M]  /home/student/rootkit-code/syscallmonitor/killmonitor.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/student/rootkit-code/syscallmonitor/killmonitor.mod.o
  LD [M]  /home/student/rootkit-code/syscallmonitor/killmonitor.ko
make[1]: Leaving directory '/home/student/openwrt-buildsystem/build_dir/target-mipsel_24kc_musl/linux-malta_le/linux-4.14.95'
student@attackdefense:~/rootkit-code/syscallmonitor$
```

Check the contents of the directory after running the mipsmake.

Command: ls -l

```
student@attackdefense:~/rootkit-code/syscallmonitor$ ls -l
total 64
-rw-r--r-- 1 student student   175 Oct  3 17:02 Makefile
-rw-rw-r-- 1 student student     0 Dec 10 16:55 Module.symvers
-rw-r--r-- 1 student student  1825 Oct  3 17:02 killmonitor.c
-rw-rw-r-- 1 student student  3636 Dec 10 16:55 killmonitor.ko
-rw-rw-r-- 1 student student   474 Dec 10 16:55 killmonitor.mod.c
-rw-rw-r-- 1 student student 14192 Dec 10 16:55 killmonitor.mod.o
-rw-rw-r-- 1 student student 28720 Dec 10 16:55 killmonitor.o
student@attackdefense:~/rootkit-code/syscallmonitor$
```

Kernel module killmonitor.ko is ready.

Important step: Check the IP address of the current machine. It should be in 192.x.y.2.

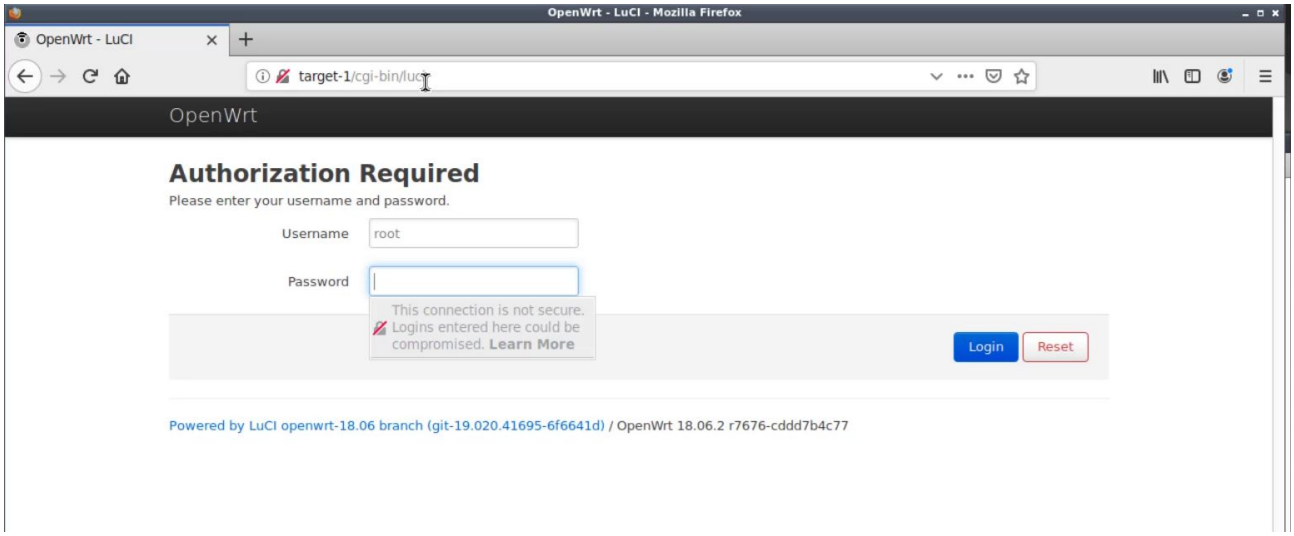
Command: ip addr

Add the entry for next IP address to domain mapping to the /etc/hosts

Command: echo "192.x.y.3 target-1" >> /etc/hosts

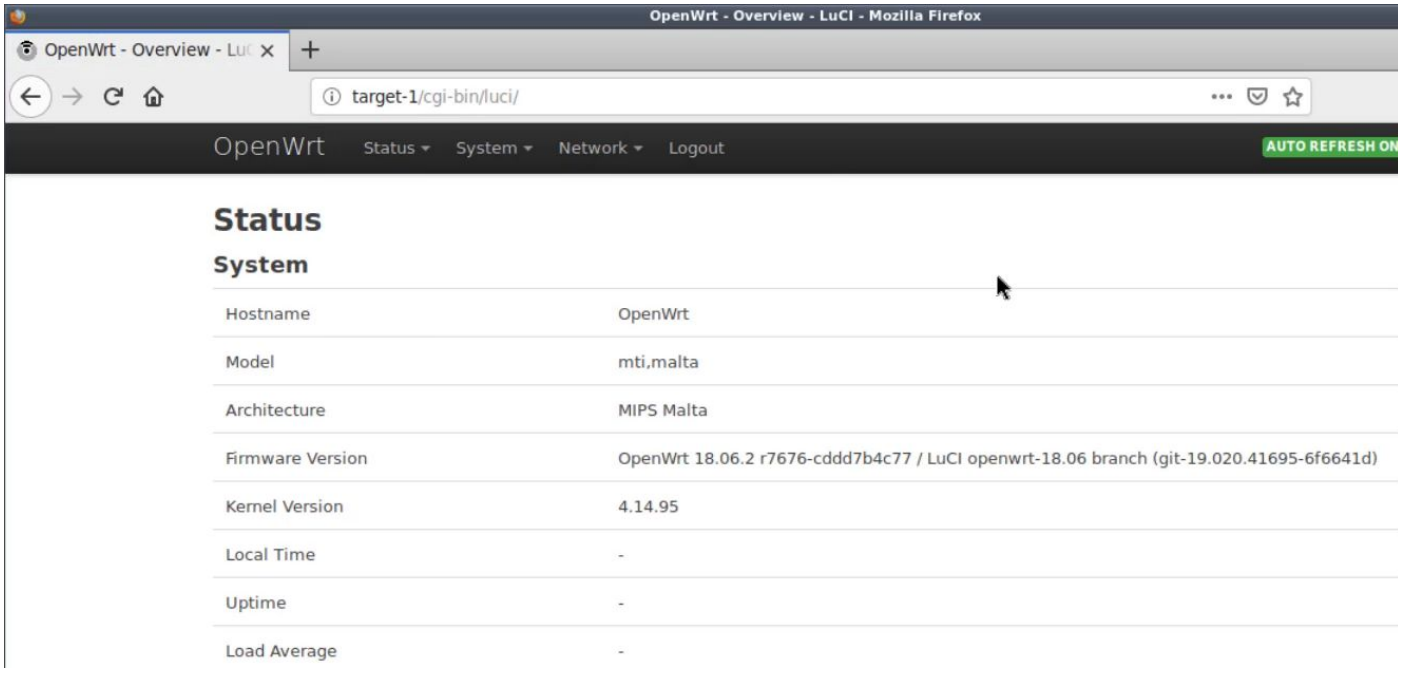
Please make sure to replace 192.x.y.3 with a real IP address.

Step 9: As mentioned in the challenge description, an emulated WiFi router is running on the same network and can be reached on “target-1”. Open the browser and navigate “to target-1”.



The credentials mentioned in the challenge description can be used to login.

Username: root
Password: chicago



The target-1 is running OpenWRT MIPS. Hence, the kernel modules can be tested on it.

Step 10: Transfer the compiled kernel module (i.e. killmonitor.ko file) to the emulated router.

Command: scp killmonitor.ko root@target-1:/root/

```
student@attackdefense:~/rootkit-code/syscallmonitor$ scp killmonitor.ko root@target-1:/root/
The authenticity of host 'target-1 (192.40.143.3)' can't be established.
ECDSA key fingerprint is SHA256:8BvWdDR+qqsV95gQDtvCihsPG1XYVtYh9UGO9bgnrx4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'target-1,192.40.143.3' (ECDSA) to the list of known hosts.
root@target-1's password:
killmonitor.ko
student@attackdefense:~/rootkit-code/syscallmonitor$
```

Step 11: Transfer the compiled kernel module (i.e. killmonitor.ko file) to the emulated router.

Command: scp killmonitor.ko root@target-1:/root/

```
student@attackdefense:~/rootkit-code/syscallmonitor$ scp killmonitor.ko root@target-1:/root/
The authenticity of host 'target-1 (192.40.143.3)' can't be established.
ECDSA key fingerprint is SHA256:8BvWdDR+qqsV95gQDtvCihsPG1XYVtYh9UGO9bgnrx4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'target-1,192.40.143.3' (ECDSA) to the list of known hosts.
root@target-1's password:
killmonitor.ko
student@attackdefense:~/rootkit-code/syscallmonitor$
```

Step 12: SSH into the target emulated router and verify the transfer.

Commands:
ssh root@target-1
ls -l

Password: chicago

```
student@attackdefense:~/rootkit-code/syscallmonitor$
student@attackdefense:~/rootkit-code/syscallmonitor$ ssh root@target-1
root@target-1's password:


BusyBox v1.28.4 () built-in shell (ash)

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 OpenWrt 18.06.2, r7676-cddd7b4c77
 -----------------------------------------------------
root@OpenWrt:~# ls -l
-rw-r--r--    1 root     root          3636 Dec 10 11:25 killmonitor.ko
root@OpenWrt:~#
root@OpenWrt:~#
```

Step 13: Insert the kernel module. Verify the successful insertion using lsmod command.

Commands:
insmod killmonitor.ko
lsmod | grep kill

```
root@OpenWrt:~# insmod killmonitor.ko
root@OpenWrt:~# lsmod | grep kill
killmonitor             1117  0
root@OpenWrt:~#
```

One can also check the kernel messages.

Command: dmesg -c

```
root@OpenWrt:~# dmesg -c
[  325.849871] [APAD killmonitor] Starting module ...
[  326.895566] [APAD killmonitor] Monitoring started...
root@OpenWrt:~#
```

Step 14: Now, to see the kernel module in action, choose a process to kill.

Command: ps -ef

```
root      2246  2143  0 11:22 ?        00:00:00 udhcpc -p /var/run/udhcpc-eth0.pid -s
root      2249  2143  0 11:22 ?        00:00:00 odhcp6c -s /lib/netifd/dhcpv6.script -
root      2297     1  0 11:23 ?        00:00:00 /usr/sbin/sshd -D
root      2327     1  0 11:23 ?        00:00:00 /usr/sbin/uhttpd -f -h /www -r OpenWrt
root      2337     1  0 11:23 ?        00:00:00 /usr/sbin/vsftpd
root      2443     1  1 11:23 ?        00:00:03 ttyd -p 8000 /bin/ash
```

Here, the target is the VSFTPD process which is running with process ID 2337. Please note that the process number will be different in each lab.

Launch the kill command on this PID.

Command: kill 2337

```
root@OpenWrt:~# kill 2337
root@OpenWrt:~#
```

Verify that the process is killed and then check the kernel logs.

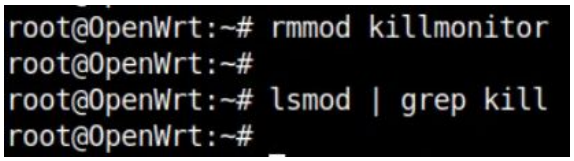
Commands:
ps -ef | grep vsft
dmesg -c

```
root@OpenWrt:~# ps -ef | grep vsft
root      3098  2815  0 11:28 pts/0    00:00:00 grep vsft
root@OpenWrt:~#
root@OpenWrt:~#
root@OpenWrt:~# dmesg -c
[  354.128276] [APAD killmonitor] Kill attempt on vsftpd (PID: 2337) by ash (PID: 2815)
[  354.129706] [APAD killmonitor] Allowing ...
root@OpenWrt:~#
```

One can observe that the call to kill syscall is logged by the kernel module.

Step 15: Remove the module and also verify the same using lsmod command.

Commands:
rmmod killmonitor
lsmod | grep kill



In this manner, one can compile and test the kernel modules in this lab. Please also try the other two kernel modules given in the rootkit-code directory.