Final Project Covert Communication

Mario Enriquez

British Columbia Institute of Technology

COMP 8505, 7D

Aman Abdulla

December 12, 2016

Contents

Introduction 3
How to Run 3
Victim 3
Host 4
Design Work 5
Test Cases 6
Test 1 8
Test 2 11
Test 3 13
Test 4 15
Test 5 18
Observations 21
Pseudocode 21
Victim Machine 21
Host Machine 23

Introduction

For this assignment, I created a Covert Communication program. The program has two functions:

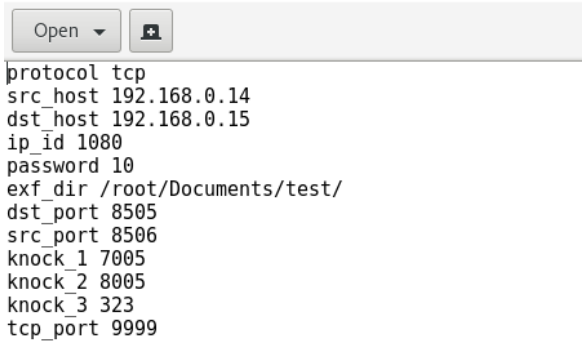
- To send and execute commands from a remote machine on a victim machine, and to get the results of the commands.
- To act as an exfiltration tool which detects the creation or modification of a file in a selected directory of a victim machine, and sends the contents of the created or modified file to a remote machine.

This is with the purpose of using everything we learned during this term in a single program. We have to implement raw sockets, backdoors, port knocking and exfiltration to create one tool which lets us gain control of a remote machine.

How to Run

Victim

-Set configuration file parameters



```
root@datacomm:~/Documents
File  Edit  View  Search  Terminal  Help
[root@datacomm victim]# make clean
rm -f *.o core covert
[root@datacomm victim]# make covert
g++ -c send_packet.cpp
g++ -c headers.cpp
g++ -c exfilt.cpp
g++ -o covert covert.cpp send_packet.o headers.o exfilt.o -lpcap -lpthread
[root@datacomm victim]# ./covert HELLO config.txt
```

Host

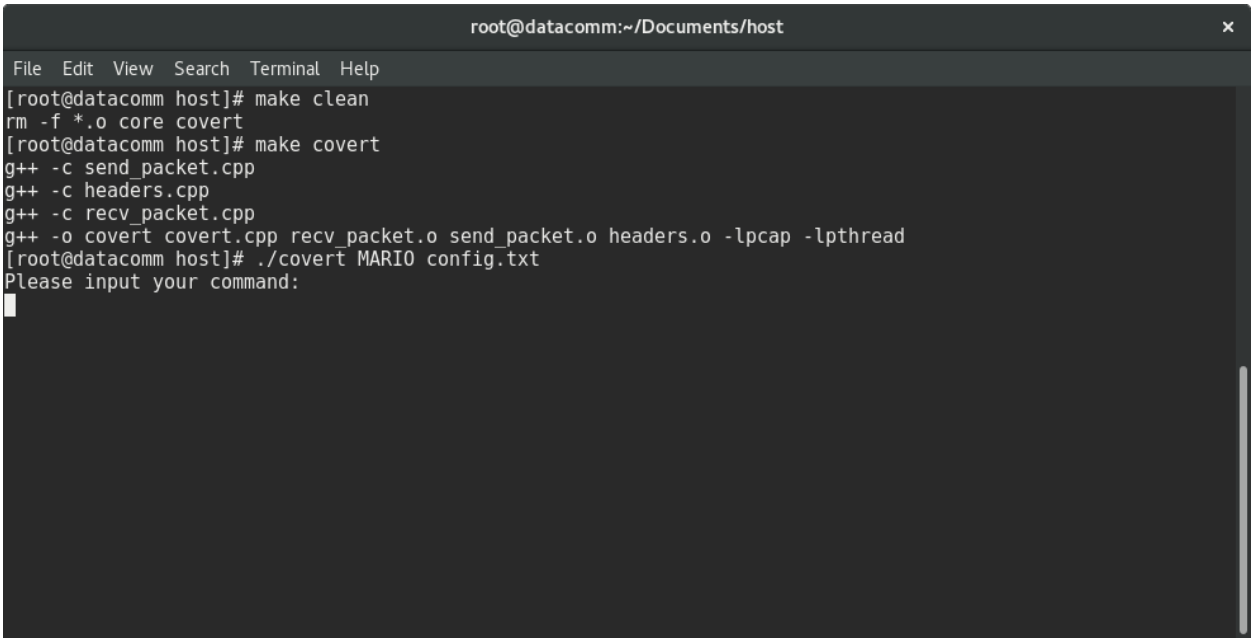
-Set configuration file parameters

```
Open    config.txt    Save
~/Documents/host
protocol tcp
src_host 192.168.0.15
dst_host 192.168.0.14
ip_id 1080
password 10
dst_port 8505
src_port 8506
knock_1 7005
knock_2 8005
knock_3 323
tcp_port 9999
```

-Make clean

-Make covert

- run $./covert (mask) (configuration file)



Design Work

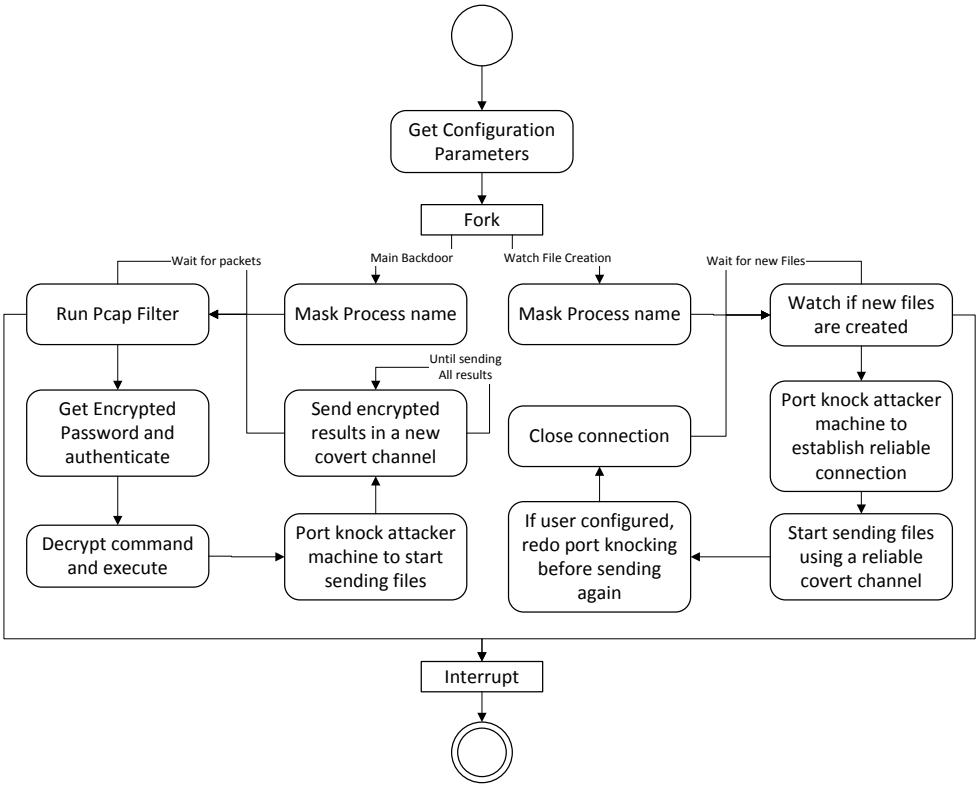


Figure 1 Victim Machine

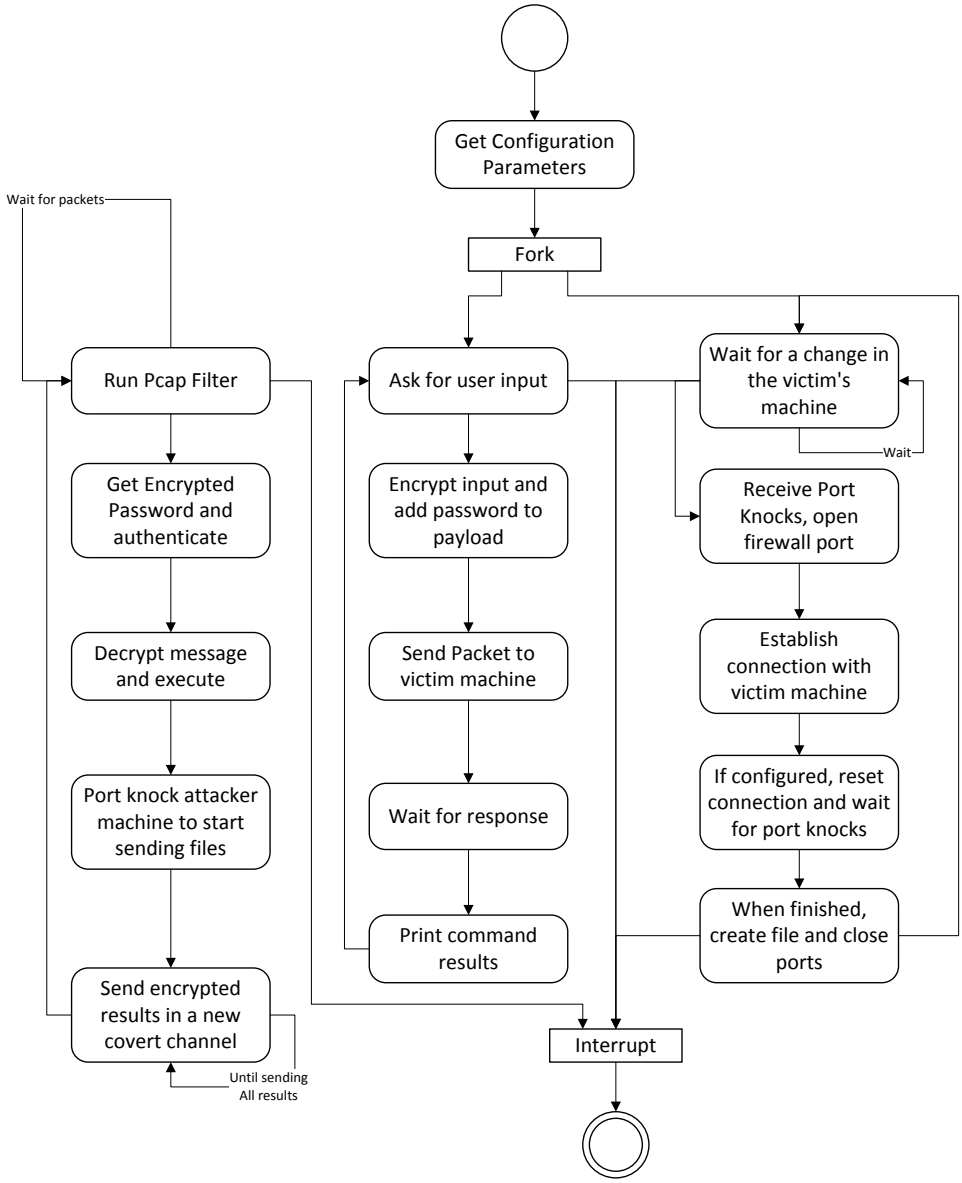


Figure 2 Host Machine

Test Cases

Test	Scenario	Tools	Expected Results	Actual Results
1	Test running a command to get current machine	Fedora	Success	Success, ipconfig returns the victim's IP
2	Test running a	Fedora	Success	Success we get the

	command to get text			results through the payload
3	Test Exfiltration component by creating and transferring a small file	Fedora	Success	Success, we get the new file in our host machine
4	Test Port knocking by analyzing the iptables rules	Fedora	Success	Success, the iptales allow the victim to establish connection
5	UDP test, run different commands	Fedora	Success	Partial Success, while the command execution runs fine, it makes the exfiltration don’t work.

Test 1

Host

```
root@datacomm:~/Documents/host
File  Edit  View  Search  Terminal  Help
root     24585  0.0  0.0  45644  1928 tty2     S+   07:40   0:00 /usr/sbin/dumpcap -n -
i eno1 -y EN10MB -Z none
root     24613  0.0  0.0  17808  1148 ?        Ss   07:40   0:00 /usr/lib/systemd/syste
md-hostnamed
root     24623  0.0  0.0 151188  3596 pts/0    R+   07:40   0:00 ps auxw
sudo ifconfig
udp
sending datagram
Please input your command:
eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.14  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::9804:585d:31c2:9bab  prefixlen 64  scopeid 0x20<link>
        ether 98:90:96:dc:ed:2f  txqueuelen 1000  (Ethernet)
        RX packets 104920  bytes 56180488 (53.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 34457  bytes 3549395 (3.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xf7d00000-f7d20000

enp3s2: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 00:0e:0c:51:2c:cc  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 68  bytes 5256 (5.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 68  bytes 5256 (5.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 00:00:00:00:00:00  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 3 ifconfig victim

```
root@datacomm:~/Documents/host
File  Edit  View  Search  Terminal  Help
        inet6 fe80::9804:585d:31c2:9bab  prefixlen 64  scopeid 0x20<link>
        ether 98:90:96:dc:ed:2f  txqueuelen 1000  (Ethernet)
        RX packets 104920  bytes 56180488 (53.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 34457  bytes 3549395 (3.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xf7d00000-f7d20000

enp3s2: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 00:0e:0c:51:2c:cc  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 68  bytes 5256 (5.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 68  bytes 5256 (5.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 00:00:00:00:00:00  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

^C
[root@datacomm host]# sudo ifconfig
eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.15  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::7ec7:657a:eaf6:64f4  prefixlen 64  scopeid 0x20<link>
        ether 98:90:96:d4:af:4f  txqueuelen 1000  (Ethernet)
        RX packets 164805  bytes 116920741 (111.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 66194  bytes 8410954 (8.0 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xf7d00000-f7d20000

enp3s2: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
```

Figure 4 ifconfig host vs victim

Victim

```
root@datacomm:~/
File  Edit  View  Search  Terminal  Tabs  Help
root@datacomm:~/Documents/victim
[root@datacomm victim]# sudo ifconfig
eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.14  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::9804:585d:31c2:9bab  prefixlen 64  scopeid 0x20<link>
        ether 98:90:96:dc:ed:2f  txqueuelen 1000  (Ethernet)
        RX packets 104961  bytes 56189812 (53.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 34460  bytes 3551190 (3.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xf7d00000-f7d20000

enp3s2: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 00:0e:0c:51:2c:cc  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 68  bytes 5256 (5.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 68  bytes 5256 (5.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 00:00:00:00:00:00  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 5 Ifconfig of Victim

Test 2

Host

```
root@datacomm:~/Documents/host
File  Edit  View  Search  Terminal  Help
g++ -c send_packet.cpp
g++ -c headers.cpp
g++ -c recv_packet.cpp
g++ -o covert covert.cpp recv_packet.o send_packet.o headers.o -lpcap -lpthread
[root@datacomm host]# ./covert MARIO config.txt
Please input your command:
ls -l
tcp
sending packet
Packet Sent. Length : 45
Please input your command:
total 148
-rw-r--r-- 1 root root   190 Dec 12 06:13 config.txt
-rwxr-xr-x 1 root root 73536 Dec 12 07:26 covert
-rw-r--r-- 1 root root  4247 Dec 12 05:42 covert.cpp
-rw-r--r-- 1 root root  4419 Dec 12 06:49 exfilt.cpp
-rw-r--r-- 1 root root  8968 Dec 12 07:26 exfilt.o
-rw-r--r-- 1 root root  5909 Dec 12 06:11 headers.cpp
-rw-r--r-- 1 root root  3020 Dec 12 05:51 headers.h
-rw-r--r-- 1 root root 12152 Dec 12 07:26 headers.o
-rw-r--r-- 1 root root   250 Dec 11 22:50 Makefile
-rw-r--r-- 1 root root  6244 Dec 12 05:22 send_packet.cpp
-rw-r--r-- 1 root root  7120 Dec 12 07:26 send_packet.o
```

Figure 6 Successful command

```
root@datacomm:~/Documents/host
File  Edit  View  Search  Terminal  Tabs  Help
root@datacomm:~/Documents/host    root@datacomm:~/Documents/host
[root@datacomm host]# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[root@datacomm host]#
```

Figure 7 Ip tables

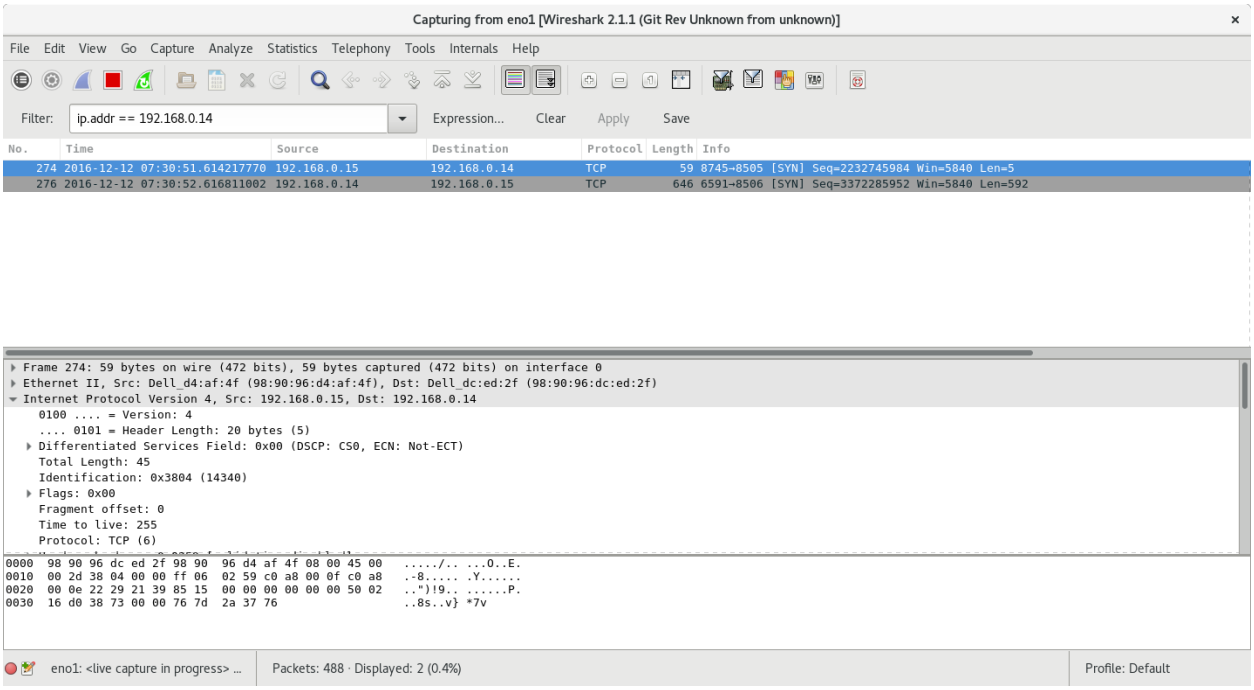


Figure 8 Exchange Victim and host

Victim

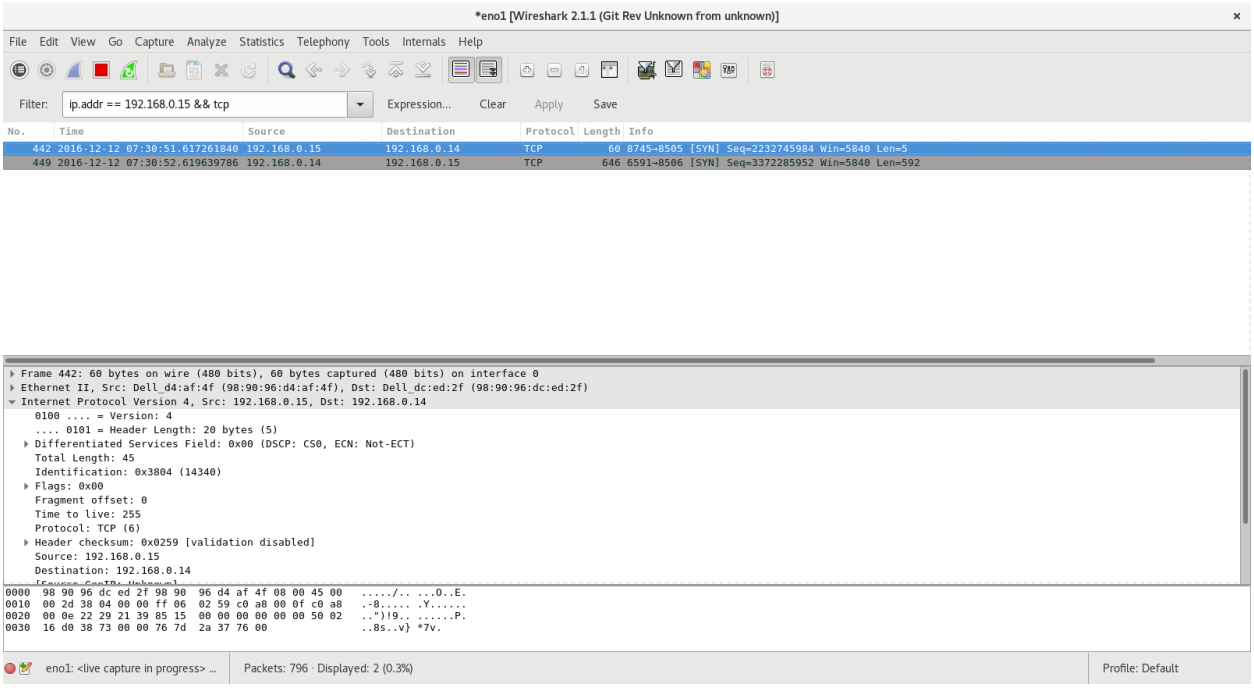


Figure 9 Exchange victim and host

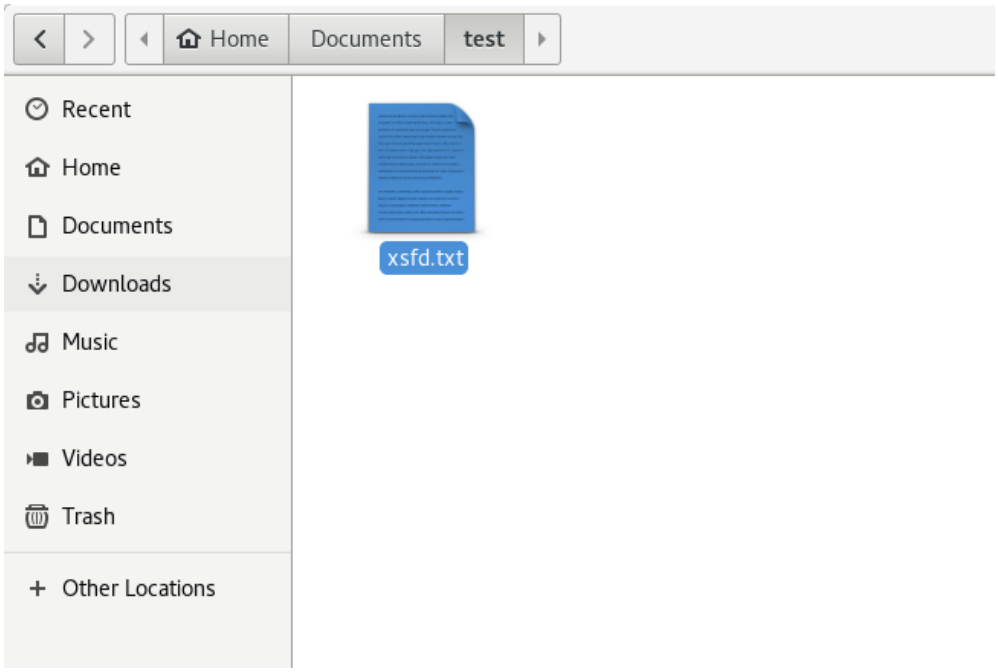


Figure 10 File to Transfer

Test 3

Host

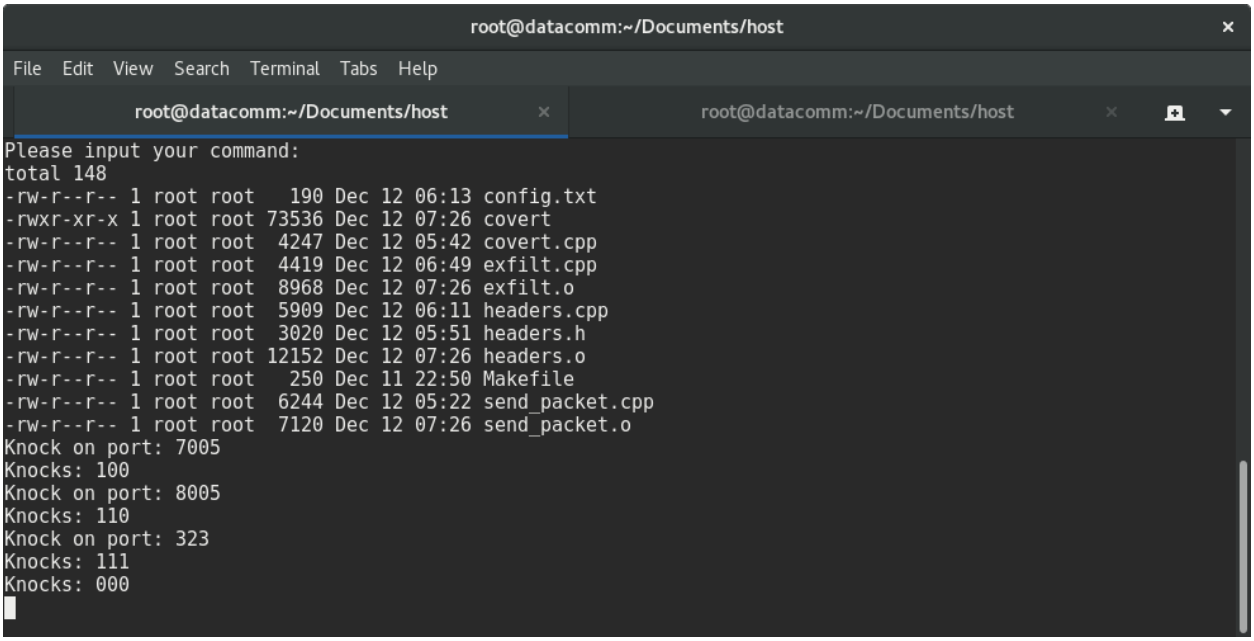


Figure 11 Port Knocking

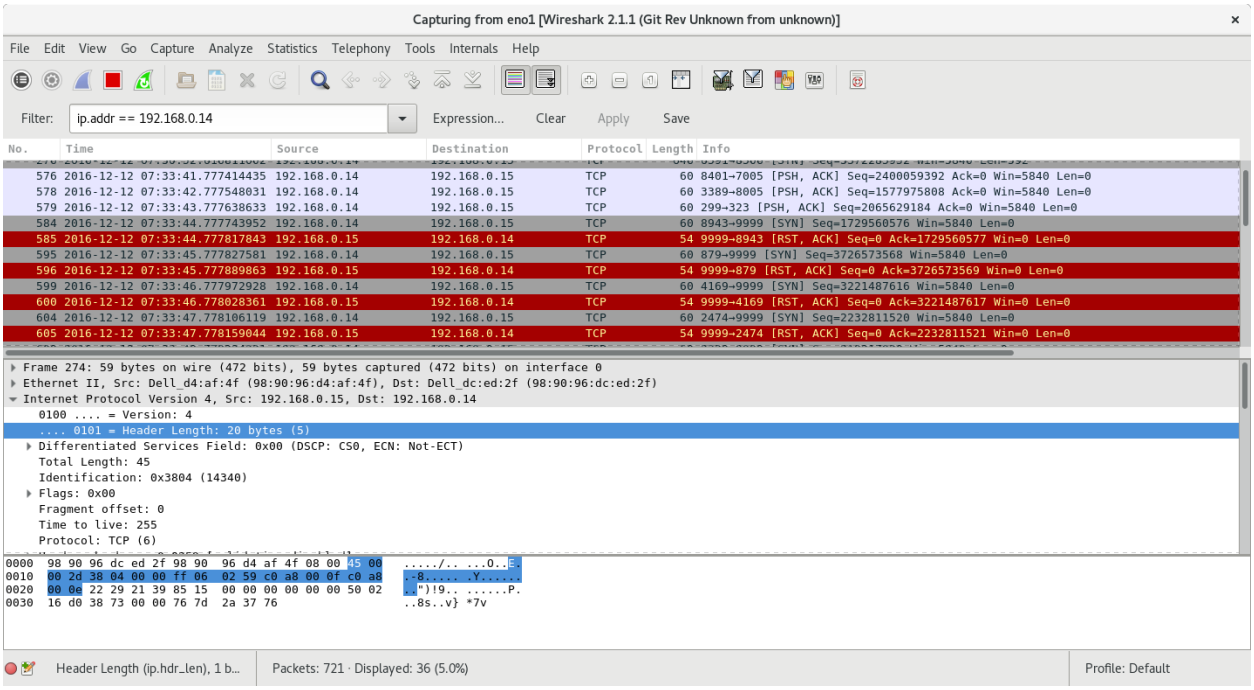


Figure 12 Exchange

Victim

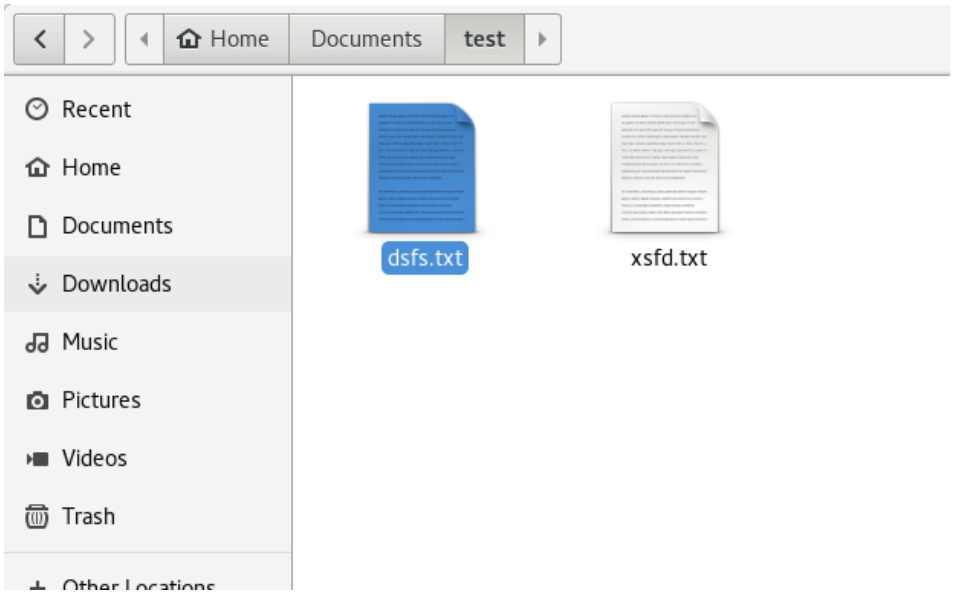


Figure 13 Files transferred

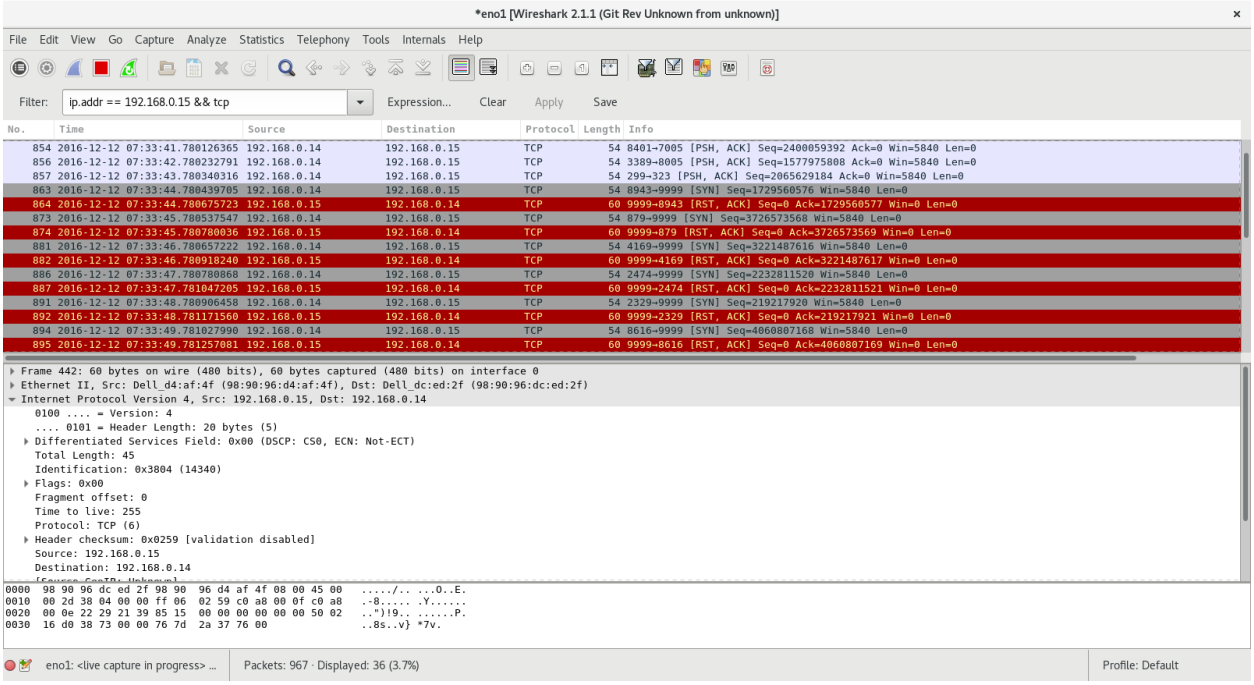
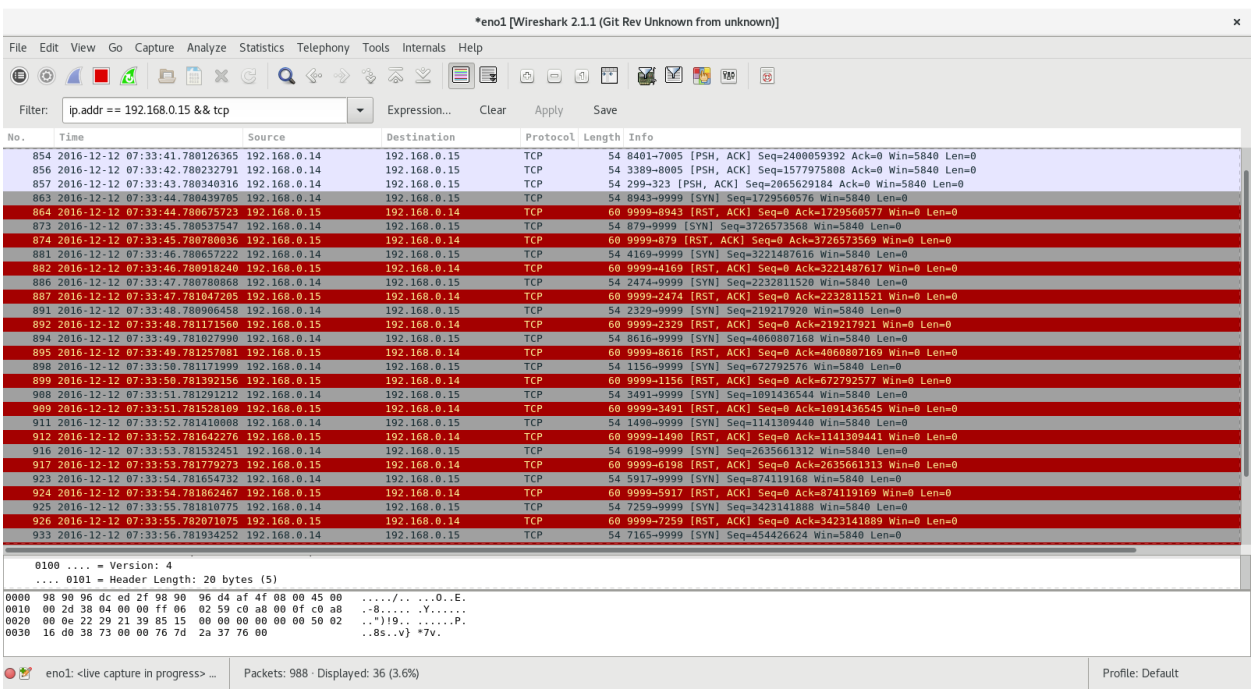


Figure 14 Port Knock exchange



Test 4

Host

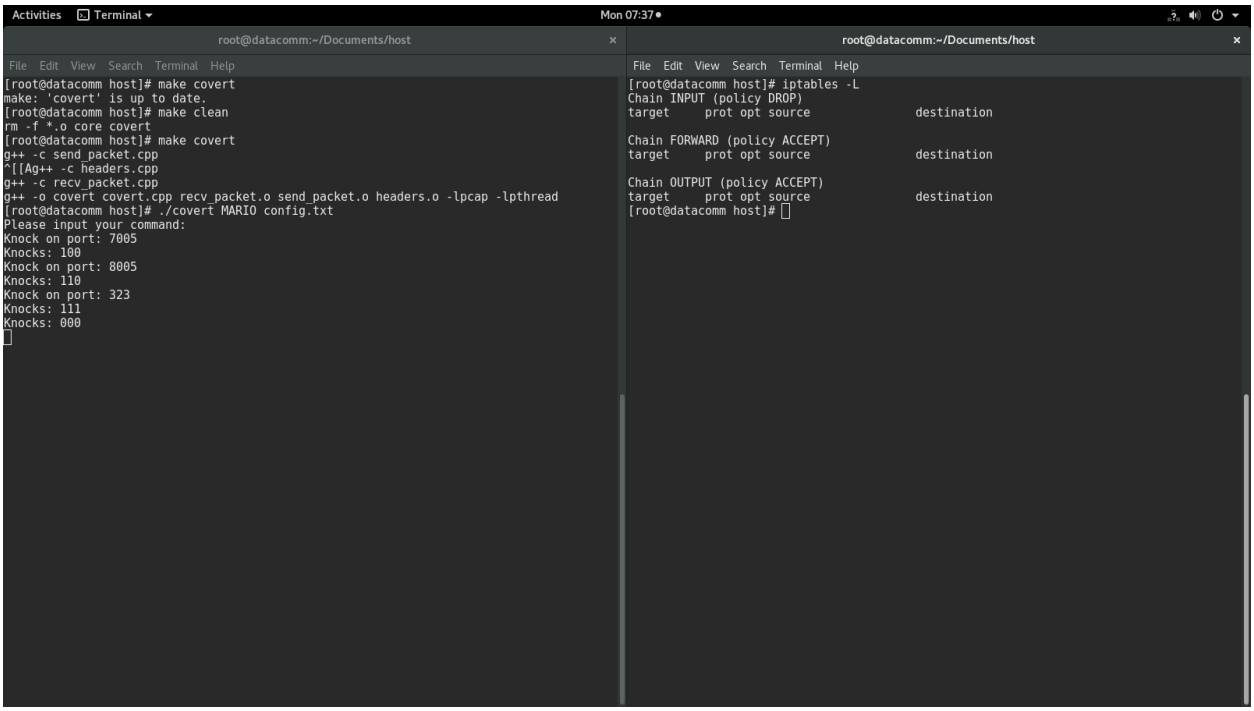


Figure 15 Default

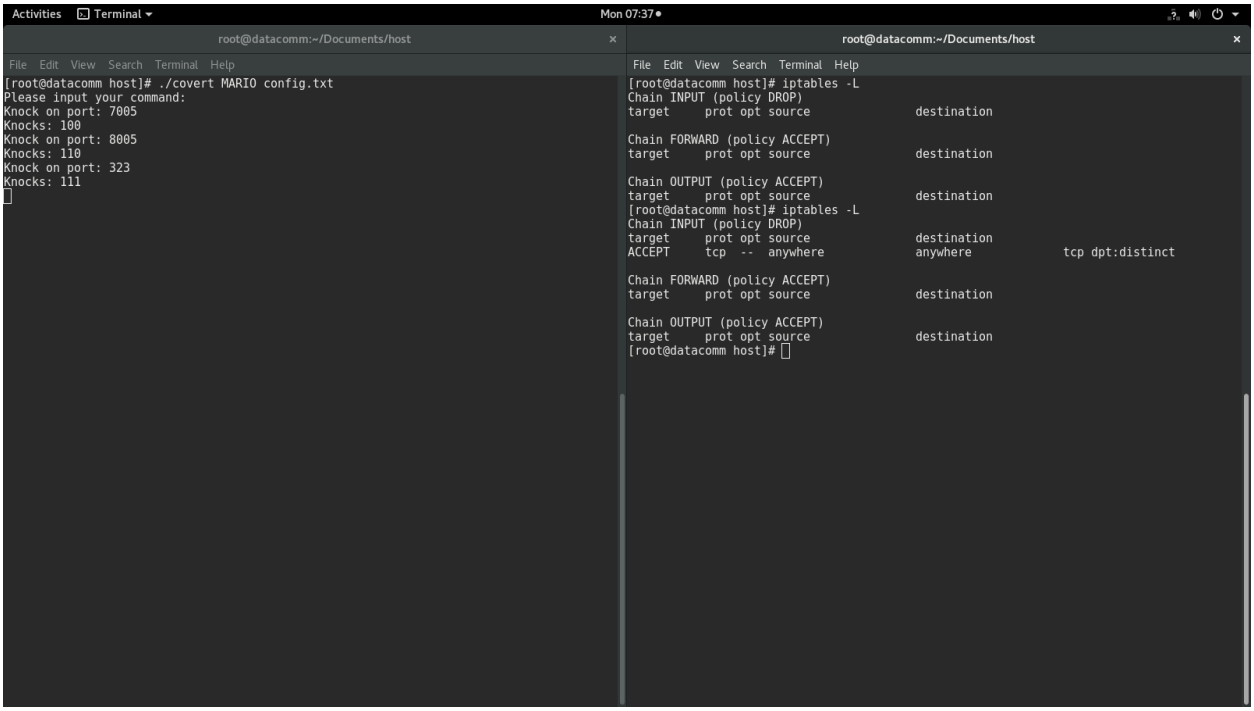


Figure 16 Port Knocking Successful

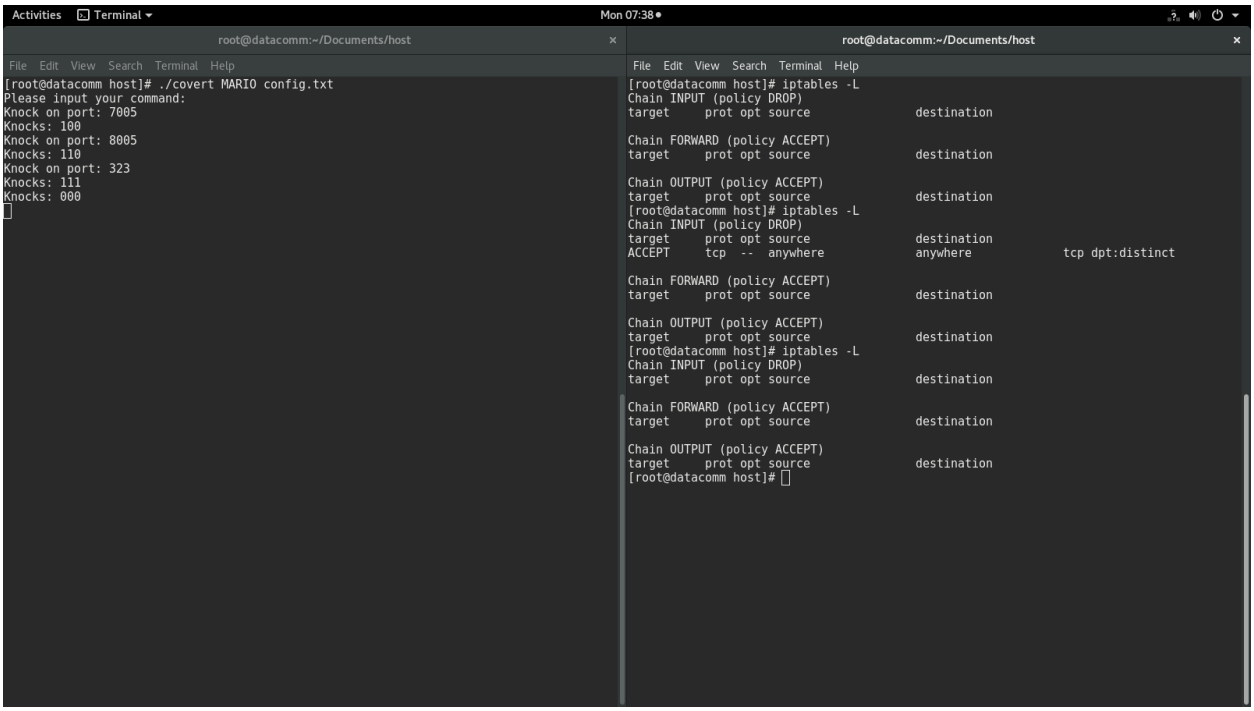


Figure 17 IP tables rules again set to drop

Victim

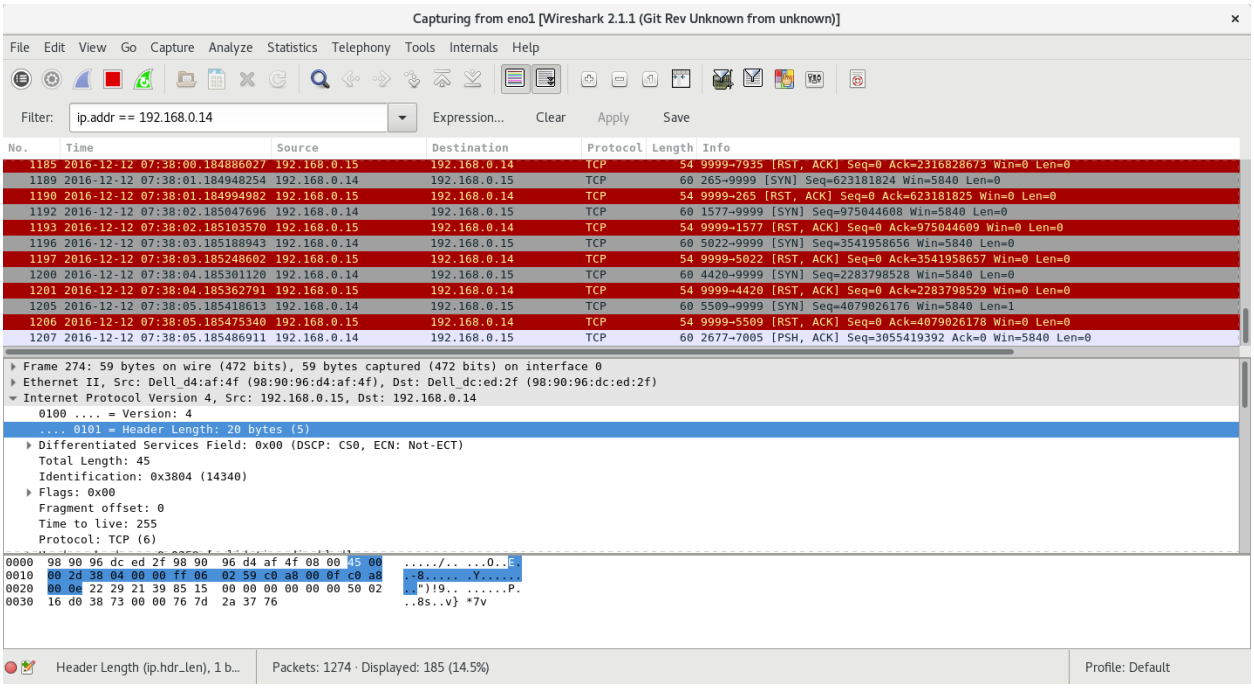


Figure 18 Exchange between Host and Victim

Test 5

Host

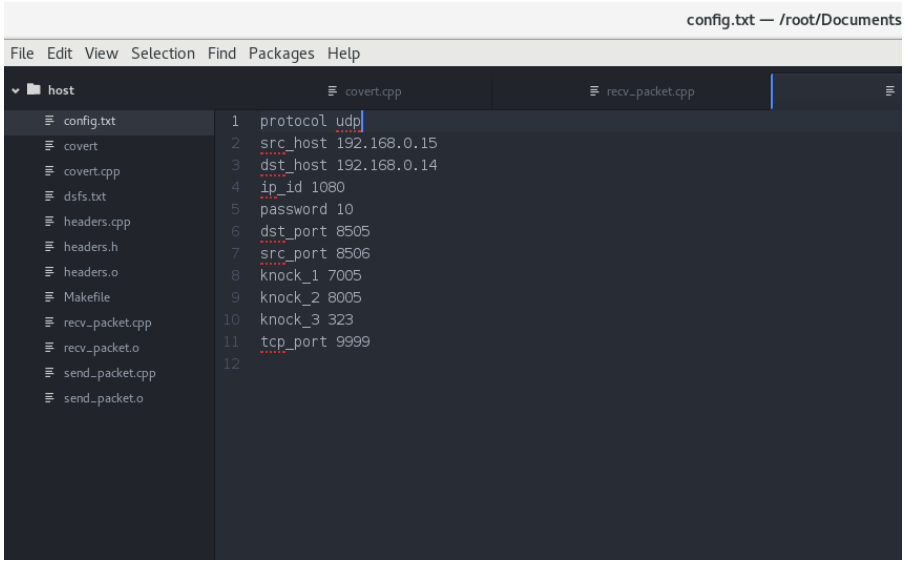


Figure 19 Configuration file for UDP

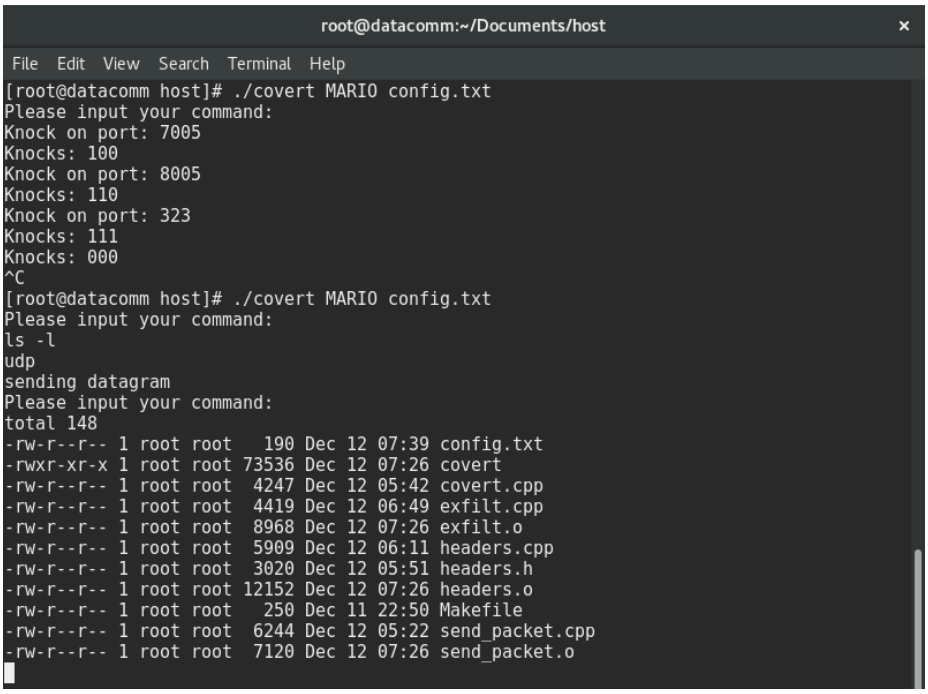


Figure 20 Results of UDP

```
root@datacomm:~/Documents/host
File  Edit  View  Search  Terminal  Help
root        24  0.0  0.0      0     0 ?        S    Dec11   0:00 [watchdog/2]
root        25  0.0  0.0      0     0 ?        S    Dec11   0:00 [migration/2]
root        26  0.0  0.0      0     0 ?        S    Dec11   0:00 [ksoftirqd/2]
root        28  0.0  0.0      0     0 ?        S<   Dec11   0:00 [kworker/2:0H]
root        29  0.0  0.0      0     0 ?        S    Dec11   0:06 [rcuos/2]
root        30  0.0  0.0      0     0 ?        S    Dec11   0:00 [rcuob/2]
root        31  0.0  0.0      0     0 ?        S    Dec11   0:00 [cpuhp/3]
root        32  0.0  0.0      0     0 ?        S    Dec11   0:00 [watchdog/3]
root        33  0.0  0.0      0     0 ?        S    Dec11   0:00 [migration/3]
root        34  0.0  0.0      0     0 ?        S    Dec11   0:00 [ksoftirqd/3]
root        36  0.0  0.0      0     0 ?        S<   Dec11   0:00 [kworker/3:0H]
root        37  0.0  0.0      0     0 ?        S    Dec11   0:04 [rcuos/3]
root        38  0.0  0.0      0     0 ?        S    Dec11   0:00 [rcuob/3]
root        39  0.0  0.0      0     0 ?        S    Dec11   0:00 [kdevtmpfs]
root        40  0.0  0.0      0     0 ?        S<   Dec11   0:00 [netns]
root        41  0.0  0.0      0     0 ?        S    Dec11   0:00 [oom_reaper]
root        42  0.0  0.0      0     0 ?        S<   Dec11   0:00 [writeback]
root        43  0.0  0.0      0     0 ?        S    Dec11   0:00 [kcompactd0]
root        44  0.0  0.0      0     0 ?        SN   Dec11   0:00 [ksmd]
root        45  0.0  0.0      0     0 ?        SN   Dec11   0:00 [khugepaged]
root        46  0.0  0.0      0     0 ?        S<   Dec11   0:00 [crypto]
root        47  0.0  0.0      0     0 ?        S<   Dec11   0:00 [kintegrityd]
root        48  0.0  0.0      0     0 ?        S<   Dec11   0:00 [bioset]
root        49  0.0  0.0      0     0 ?        S<   Dec11   0:00 [kblockd]
root        50  0.0  0.0      0     0 ?        S<   Dec11   0:00 [ata_sff]
root        51  0.0  0.0      0     0 ?        S<   Dec11   0:00 [md]
root        52  0.0  0.0      0     0 ?        S<   Dec11   0:00 [devfreq_wq]
root        53  0.0  0.0      0     0 ?        S<   Dec11   0:00 [watchdogd]
root        56  0.0  0.0      0     0 ?        S    Dec11   0:00 [kswapd0]
root        57  0.0  0.0      0     0 ?        S<   Dec11   0:00 [vmstat]
root       103  0.0  0.0      0     0 ?        S<   Dec11   0:00 [kthrotld]
root       105  0.0  0.0      0     0 ?        S<   Dec11   0:00 [acpi_thermal_pm]
root       106  0.0  0.0      0     0 ?        S    Dec11   0:00 [scsi_eh_0]
root       107  0.0  0.0      0     0 ?        S<   Dec11   0:00 [scsi_tmf_0]
root       108  0.0  0.0      0     0 ?        S    Dec11   0:00 [scsi_eh_1]
root       109  0.0  0.0      0     0 ?        S<   Dec11   0:00 [scsi_tmf_1]
root       110  0.0  0.0      0     0 ?        S    Dec11   0:00 [scsi_eh_2]
root       111  0.0  0.0      0     0 ?        S<   Dec11   0:00 [scsi_tmf_2]
root       112  0.0  0.0      0     0 ?        S    Dec11   0:00 [scsi_eh_3]
root       113  0.0  0.0      0     0 ?        S<   Dec11   0:00 [scsi_tmf_3]
root       114  0.0  0.0      0     0 ?        S    Dec11   0:00 [scsi_eh_4]
root       115  0.0  0.0      0     0 ?        S<   Dec11   0:00 [scsi_tmf_4]
root       116  0.0  0.0      0     0 ?        S    Dec11   0:00 [scsi_eh_5]
root       11
```

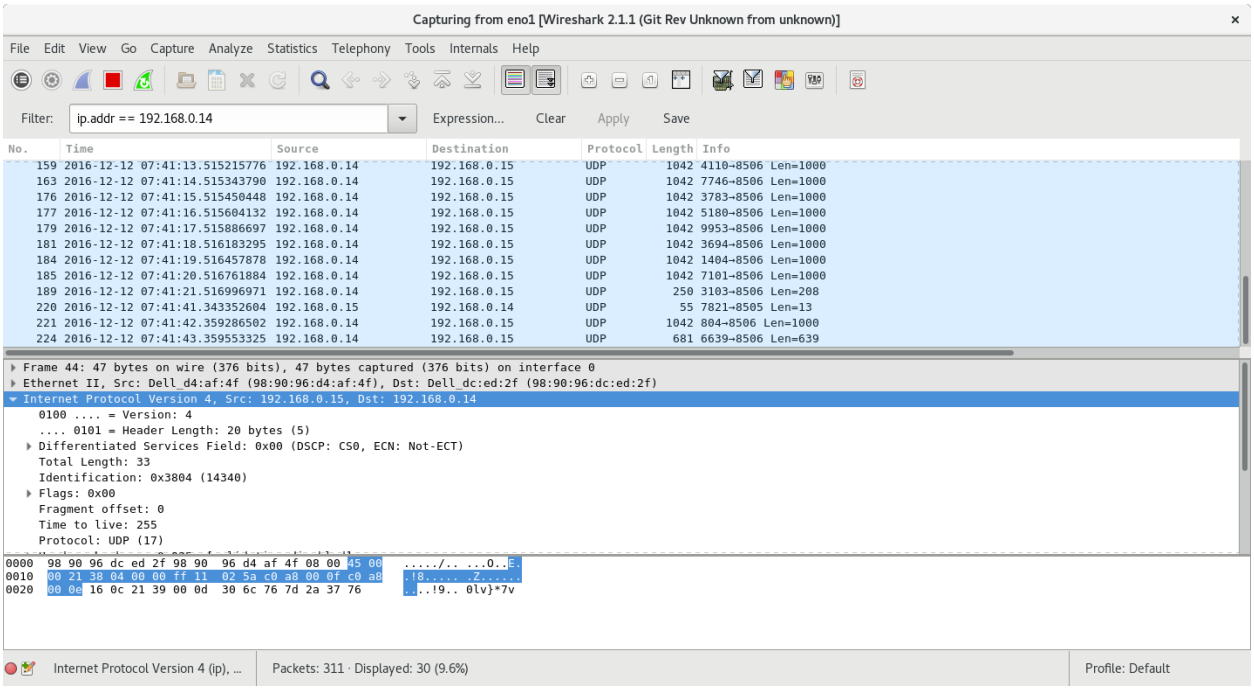


Figure 21 Exchange UDP

Victim

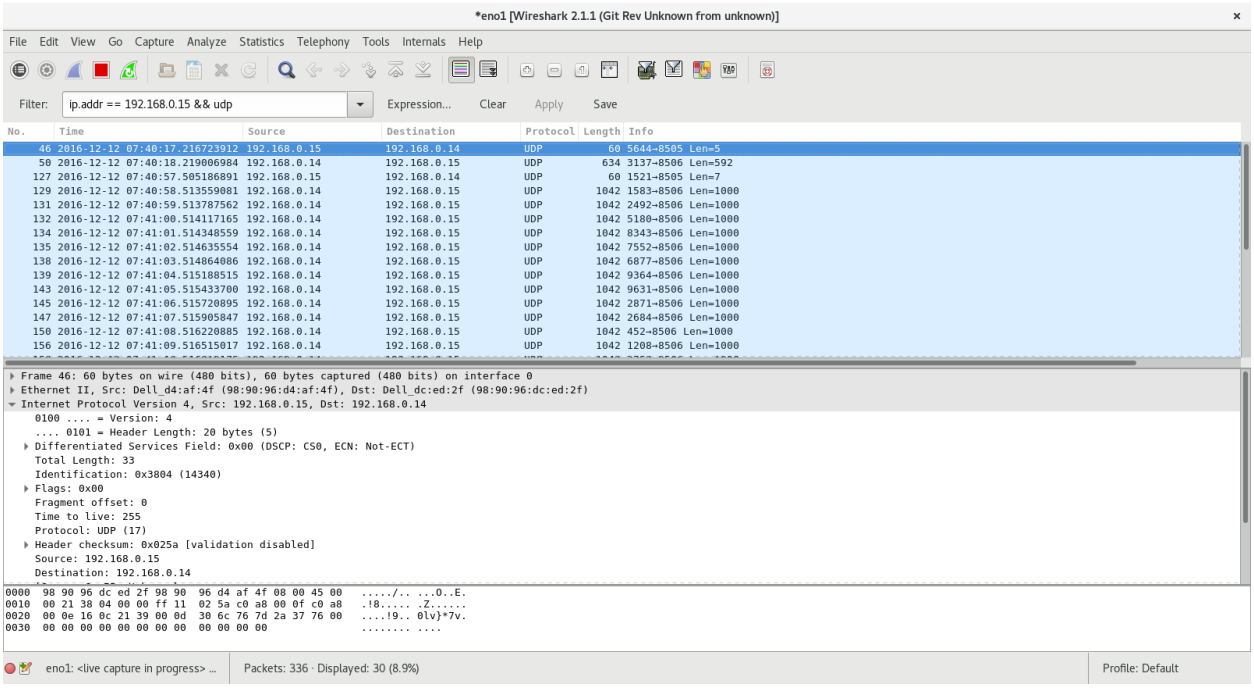


Figure 22 UDP results

Observations

- By default, inotify may have trouble depending on how the file is modified or created. Gedit on Linux can be troublesome because it creates and modifies over a temporal file.
- Port knocking allows us to set the firewall as a filter, as it can allow only raw sockets directed to a specific port to get to us.

Pseudocode

Victim Machine

```
Include libraries


Main Function
        Open Configuration file
        Read Configuration file
        Assign values
        Create interruption handler
        Start Thread Function Capture
        Start Thread Function Ex-filtration
        Join
        Exit
End of Main


Capture Function, receives Filter parameters, password
        Start libpcap filter to capture packets from host machine directed to an specific port
        Grab payload contents
```

```
        Authenticate payload by using Decrypt function
        Use Decrypt Function to Decrypt rest of payload
        Execute Command
        Encrypt Command Results with Encrypt Function
        While there are still command result values
                Send result values using Craft_packet
        End of While
End of Capture

Ex-filtration Function receives directory to watch, host parameters
        Set Function to detect changes of selected file directory
        Set select function to detect the changes on the directory
        if there is a change in the directory
                Get Filename
                Port knock host machine
                Establish connection with host machine
                Encrypt Filename and contents with Encrypt Function
                Use Craft_packet to send contents of Filename
                Port Knock if neccessary and repeat sending
        End if
End of Ex-filtration

Craft_packet Function receives message, host parameters
        Set IP header parameters
        if TCPs protocol, set TCP header parameters
```

```
        if UDP protocol, set UDP header parameters
        Create pseudo-header
        Copy Packet parameters
        Perform Checksum
        Send Packet
        if TCP selected, wait for ack
            if ack is not received resend packets
            end if
        end if
End of Craft_packet Function

Decrypt Function, receives message and password
        Generate Key from password
        Decrypt message with Key
End of Decrypt Function

Encrypt Function, receives message and password
        Generate Key from password
        Encrypt message with Key
End of Encrypt Function
```

Host Machine

```
Include libraries

Main Function
```

```
        Open Configuration file
        Read Configuration file
        Assign values
        Create interruption handler
        Start Thread Function Command
        Start Thread Function Receive_File
        Start Thread Function Capture
        Join
        Exit
End of Main


Command Function
        While True
                Prompt user Input
                Create and send packet using Craft_packet function
                Wait for Response
                Print Result
        End While
End of Command Function


Receive_File Function
        While true
                Wait until start of port knocking
```

```
                If Port Knocking sequence is correct drop Firewall rules
                        Establish connection with victim machine
                        Receive File data from the covert channel
                        If necessary receive port knocking sequence again
                        Decrypt File using Decrypt function
                        Write File in Directory
                End if
        End While
End Receive_File


Craft_packet Function receives message, host parameters
        Set IP header parameters
        if TCPs protocol, set TCP header parameters
        if UDP protocol, set UDP header parameters
        Create pseudo-header
        Copy Packet parameters
        Perform Checksum
        Send Packet
        if TCP selected, wait for ack
                if ack is not received resend packets
                end if
        end if
End of Craft_packet Function
```

```
Capture Function, receives Filter parameters, password
        Start libpcap filter to capture packets from host machine directed to an specific port
        Grab payload contents
        Authenticate payload by using Decrypt function
        Use Decrypt Function to Decrypt rest of payload
        Execute Command
        Encrypt Command Results with Encrypt Function
        While there are still command result values
                Send result values using Craft_packet
        End of While
End of Capture


Decrypt Function, receives message and password
        Generate Key from password
        Decrypt message with Key
End of Decrypt Function


Encrypt Function, receives message and password
        Generate Key from password
        Encrypt message with Key
End of Encrypt Function
```