CYBR371 – Lab 3
greenthom – 300536064

Part 1 – Denial of Service Attacks

Q1.1 Why aren't new operating systems susceptible to Ping of Death attack?

Ping of Death is an older DoS attack that aims to crash the target system or reboot by sending ping packets that exceed the maximum byte size allowed by the TCP/IP protocol.

Most new operating systems are not prone to PoD attacks because of being designed with improved network stack implementations that validate incoming packets (inputs) to check buffer overflow attacks. This works by when receiving a ping packet, the system will first check the size of the packets. It will look at if it is too large or malformed and if it is it will then reject these packets. There is other countermeasure in modern systems such as firewalls and other security features that can detect and reject/block PoD attacks.

Q1.2 How can you make the Ping of Death packets effective against a target these days?

Ping of Death attack is an older DoS attack so an attacker can launch a DDoS attack by flooding the target with a large volume of ICMP Echo request packets. These packets would exploit vulnerabilities in target systems, leading to crashes. Here are some ways to do this:

- Attacker can launch Smurf attack by sending ICMP echo request packets with victims IP address as src address to multiple broadcast address on a network. This would cause the machines on the network to send an echo reply to target system leading to the system being overwhelmed with traffic. This can be combined with PoD.
- Fragmenting oversized ICMP packets into smaller packets that would be reassembled in the target system. By sending fragmented packets this would successfully overwhelm the target system causing DoS when reassembled. This can be combined with PoD

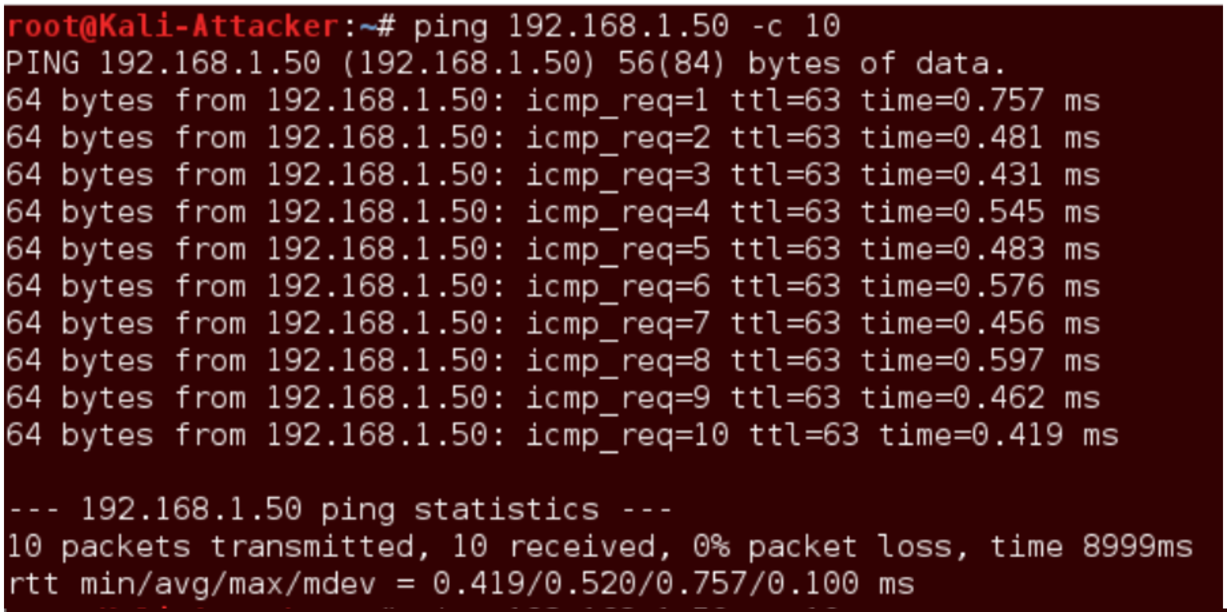
Q1.3 Briefly explain the countermeasures to stop and defend against a Smurf attack

Smurf attack is where an attacker sends ICMP echo request packets with a victims IP address as src address to the broadcast address causing hosts on the network to respond to the target system causing it to flood the network leading to DoS. Countermeasures to this are:

- Drop the traffic with invalid source IP.
- Filter on layer 3 devices to not reply for IP directed broadcast.
- Using DoS mitigation services – monitor network traffic and block malicious traffic
- Filter number of ICMP packets that can be sent on the network – rate limiting on router/firewall

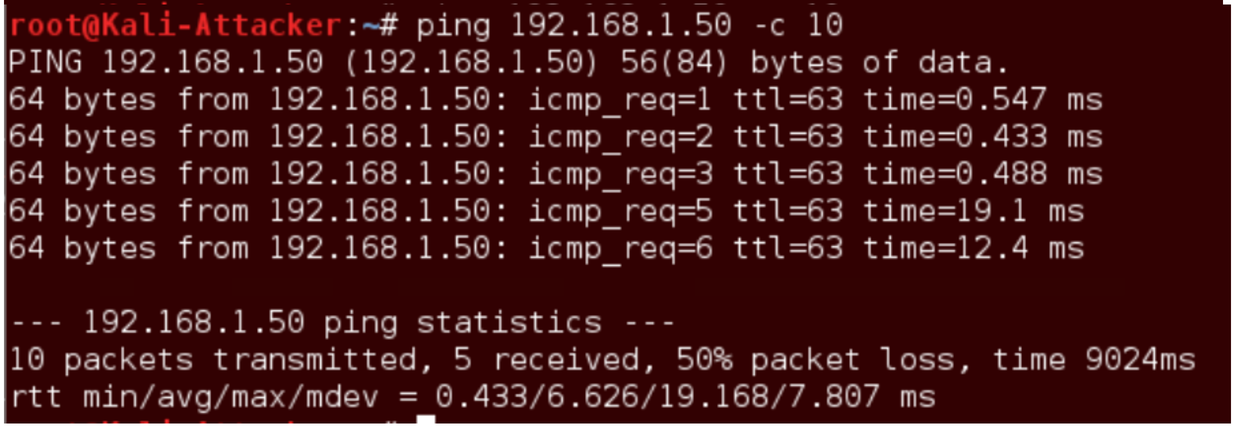
Q1.4 Did the Smurf attack slow down the network? Compare the average ping response time before and during the Smurf attack

Smurf attack causes the network to be flooded with large amounts of ICMP echo requests which leads to devices on the network to be overwhelmed. Doing the lab the target device is flooded with ICMP echo requests which resulted in an increase of network traffic and delay.



```
root@Kali-Attacker:~# ping 192.168.1.50 -c 10
PING 192.168.1.50 (192.168.1.50) 56(84) bytes of data.
64 bytes from 192.168.1.50: icmp_req=1 ttl=63 time=0.757 ms
64 bytes from 192.168.1.50: icmp_req=2 ttl=63 time=0.481 ms
64 bytes from 192.168.1.50: icmp_req=3 ttl=63 time=0.431 ms
64 bytes from 192.168.1.50: icmp_req=4 ttl=63 time=0.545 ms
64 bytes from 192.168.1.50: icmp_req=5 ttl=63 time=0.483 ms
64 bytes from 192.168.1.50: icmp_req=6 ttl=63 time=0.576 ms
64 bytes from 192.168.1.50: icmp_req=7 ttl=63 time=0.456 ms
64 bytes from 192.168.1.50: icmp_req=8 ttl=63 time=0.597 ms
64 bytes from 192.168.1.50: icmp_req=9 ttl=63 time=0.462 ms
64 bytes from 192.168.1.50: icmp_req=10 ttl=63 time=0.419 ms

--- 192.168.1.50 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 8999ms
rtt min/avg/max/mdev = 0.419/0.520/0.757/0.100 ms
```



```
root@Kali-Attacker:~# ping 192.168.1.50 -c 10
PING 192.168.1.50 (192.168.1.50) 56(84) bytes of data.
64 bytes from 192.168.1.50: icmp_req=1 ttl=63 time=0.547 ms
64 bytes from 192.168.1.50: icmp_req=2 ttl=63 time=0.433 ms
64 bytes from 192.168.1.50: icmp_req=3 ttl=63 time=0.488 ms
64 bytes from 192.168.1.50: icmp_req=5 ttl=63 time=19.1 ms
64 bytes from 192.168.1.50: icmp_req=6 ttl=63 time=12.4 ms

--- 192.168.1.50 ping statistics ---
10 packets transmitted, 5 received, 50% packet loss, time 9024ms
rtt min/avg/max/mdev = 0.433/6.626/19.168/7.807 ms
```

Before: 0.520ms. After: 6.626. Different of 6.120ms when ping attack launched

Part 2 – ARP Spoofing and MiTM Attacks

Q2.1 Is Ettercap using ARP spoofing to manipulate http images and JavaScripts? Explain your answer.

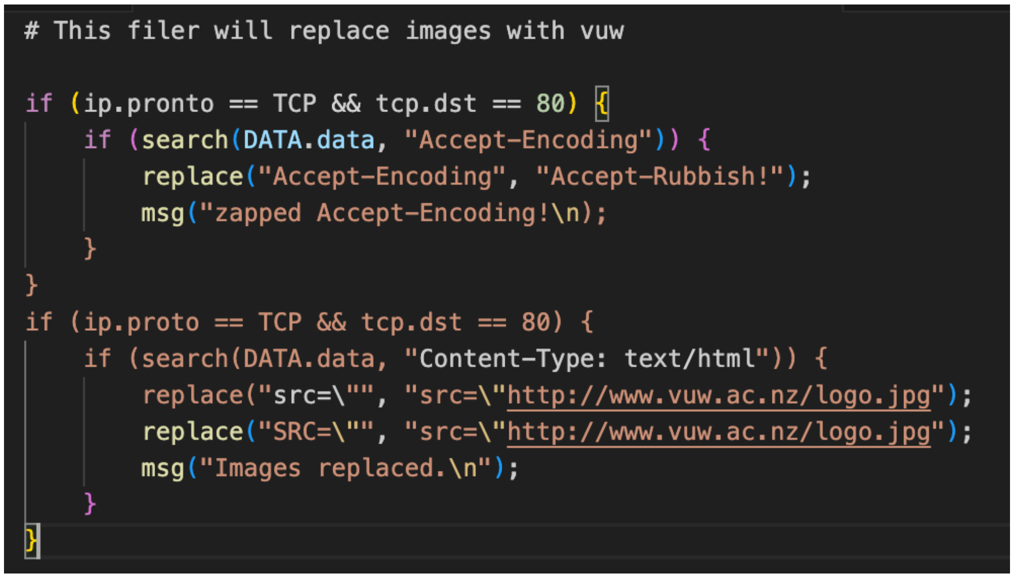
Ettercap can be used to man-in-the-middle attacks on a LAN. ARP spoofing is one of the primary techniques it employs to intercept and manipulate traffic between two devices (HTTP images on traffic and JavaScript). When ARP spoofing is used with Ettercap, it can intercept HTTP traffic between a client and a server. It does this through spoofing the ARP table of the target device. This will then trick the target device to send its traffic through the attacker's machine. This interception allows Ettercap to intercept and manipulate/modify the traffic coming through their machine including the contents of HTTP responses including images and JavaScript files. It can then send these to the target device. Ettercap can be used to inject content of scripts into web pages that are being accessed by the victim, thereby manipulating the images and scripts served to the client.

Q2.2 What would be the effect if instead of http, we had https? Explain your answer

HTTPS encrypts the data exchanged between the browser and the server using TLS/SSL protocols which ensures that the content of the communication is secure and cannot be easily intercepted or manipulated by tools such as Ettercap. Prevention of tampering or modification during transit using HTTPS ensures the integrity of the data exchanged between the client and server. Using Ettercap to intercept HTTPS traffic would likely be detected due to cryptographic checks that are performed during TLS/SSL handshake.

Q2.3 What would you change to replace every image in the html page with an image of VUW logo

Change second if statement in logo. Replace img src tags in html with the address of the VUW logo. This would change all images to the VUW logo when run. This is my interpretation of the script in the lab:



```
# This filer will replace images with vuw

if (ip.pronto == TCP && tcp.dst == 80) {
    if (search(DATA.data, "Accept-Encoding")) {
        replace("Accept-Encoding", "Accept-Rubbish!");
        msg("zapped Accept-Encoding!\n);
    }
}
if (ip.proto == TCP && tcp.dst == 80) {
    if (search(DATA.data, "Content-Type: text/html")) {
        replace("src=\"", "src=\"http://www.vuw.ac.nz/logo.jpg");
        replace("SRC=\"", "src=\"http://www.vuw.ac.nz/logo.jpg");
        msg("Images replaced.\n");
    }
}
```