DISTRIBUTED DENIAL OF SERVICE ATTACKS

COMPE-571 FINAL PROJECT REPORT

11 December 2017

KAVISHA SARASWAT (820866027)

TASNEEM SINGH (822046102)

SAN DIEGO STATE UNIVERSITY

ELECTRICAL AND COMPUTER ENGINEERING

Introduction

A Denial of Service (DoS) attack is an attack with the purpose of preventing legitimate users from using a specified network resource such as a website, web service, or computer system. A **Distributed Denial of Service (DDoS)** attack is a coordinated attack on the availability of services of a given target system or network that is launched indirectly through many compromised computing systems. The services under attack are those of the primary victim, while the compromised systems used to launch the attack are often called the secondary victims/bots.

The type of DDoS attack we worked with is the **Smurf Attack**. In a conventional scenario, a host A sends ping request to host B, prompting an automatic reply. In this attack, the ping packet's return IP address is forged with the IP of the computer that is targeted, which is then issued to the entire IP multicast address. This approach causes every computer to respond to the bogus ping packets and reply to the targeted computer, overwhelming the network.

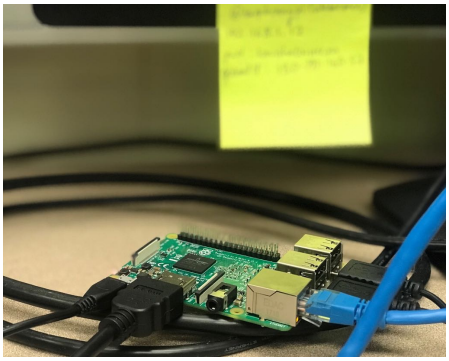

Fig. 1 The victim Pi

Fig. 2 Kali linux interface

Fig. 3 The attacking Pi

How a Smurf attack works:
A Smurf attack is not terribly sophisticated; it's just a matter of routing and letting IP take its course. The attack usually unfolds in five simple steps:

1. Hacker identifies a victim IP address (your Web server is usually a nice high-profile target).
2. Hacker identifies an intermediary site that will amplify the attack (usually several are selected, to further disguise the attack).
3. Hacker sends a large amount of **ICMP** (ping, layer 3) traffic at the broadcast address of the intermediary sites. These packets have the source IP address spoofed to point towards the victim.
4. Intermediaries deliver the broadcast at layer 2 to all the hosts on their subnet.
5. Hosts reply to the victim network.

ACCOMPLISHMENTS:

S.No	Deliverables	Success metrics
1	Setting up Raspberry Pi-1 as a web server	1. Created a LAMP webserver. 2. The server was powerful enough to accept a million requests from 1000 concurrent users.This was determined using the Apache ab benchmark tool. 3. Made our IP static and enabled port forwarding.
2	Setting up Raspberry Pi- 2 as a hacking botnet for DDoS attack using Kali Linux	1. Installed THC-IPV6 , DAVOSET and UFONET software packages.
3	Hacking Raspberry Pi-2 using Raspberry Pi-1	1. Made our wordpress web server bulky; reduced the number of concurrent clients to 385. 2. Attacked Raspberry pi-1 using smurf6 attacking tool(included in the THC-IPV6 package). 3. The server went down after using the concurrent attack command lines on 6 terminals.
4	Implementation of DDoS Detection Process	1. Used the ping, netstat -s and traceroute commands to establish a plot to depict that our web server was under attack. 2. Used Wireshark to sniff received requests under normal conditions and under DDoS attack.
5	Development of an algorithm to defend the DDoS attack	Couldn't implement the prevention mechanism due to time limitations.

HARDWARE COMPONENTS USED	SOFTWARE COMPONENTS USED
• Raspberry Pi-3	• Raspbian Jessie
• Ethernet cable	• Apache
• Monitors/Screens	• PHP
	• MySQL
	• Wordpress
	• Apache ab
	• VNC Server
	• THC-IPV6

ISSUES FACED:

1. (Deliverable 1) In the process of making our IP static ,we changed the network configuration settings in /etc/network/interfaces, which led to the Pi losing its ethernet connection permanently. The entire system had to be re-installed from scratch and we learned that the network configuration settings are always made in the /etc/dhcpcd.conf file.

2. (Deliverable 1) While setting up wordpress as the web host, we faced an 'Error establishing a database connection". We were stuck on this error for 2 consecutive days. It appeared that by simply changing the database values in the /var/www/wordpress/wp-config.php file does not help the user login to wordpress, instead the same values are to be defined and should be granted privileges in the MySQL database too.

3. (Deliverable 3) To make our website bulky, so as to reduce the number of concurrent users, we decided to upload a 1GB movie onto our website. The maximum limit to upload a media file was mere 2MB which was modified and changed to 2GB in the php.ini file. After this, though wordpress was able to accept such a large file, this somehow did not increase our web server's document length. To increase it, we uploaded and copied quite a many pictures (8~10MB in size) on various pages of our website. The document length increased and the number of concurrent users came down to 385.

4. (Deliverable 3) To perform the smurf attack, we tried and tested various attacking tools such as DAVOSET and UFONET before we finalised smurf6. To carry out the attack using DAVOSET, we needed a domain name for our website. Since we just had an IP address for our webpage, we tried using the wordpress dashboard to change the site URL. This resulted in us losing our entire website as the browser was now directing us to a page(the changed URL) every time we typed in our IP. Unfortunately, that website didn't exist and we lost our entire web server again. Also, DAVOSET and UFONET triggered HTTP flood attack. We wanted to conduct ICMP Flooding. ICMP is a very useful protocol. Most traceroute utilities use ICMP to discover the network path a packet takes to its destination. Network Administrators use ICMP daily to monitor the status of servers or routers, using a simple ICMP Echo (a.k.a. ping). Most Internet devices will respond to pings and it can reveal host OS information.

5. (Deliverable 3) While attacking the web server, we observed that the interface name on the server pi changed from eth0 to enxb827eb9eefaa. This resulted in the change of local ip of our webserver. We, however, configured the /etc/dhcpcd.conf and /etc/resolv.conf files on our system to restore the former settings. We're assuming that the change of interface name occurred due the detection of an attack at router level.

IMPLEMENTATION OUTLINE

A. THE ATTACK SCENARIO

To attack the LAMP web server we created, we initially installed a couple of attacking tools such as DAVOSET and UFONET but in the end we ultimately decided to go with Smurf6.

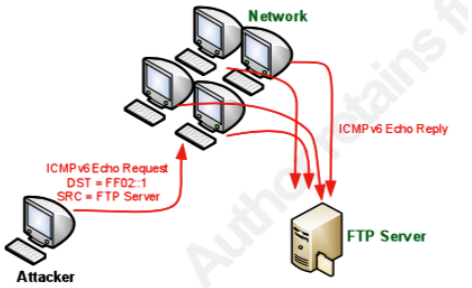

Fig.4 Actual depiction of the smurf6 attack

Smurf6 is a tool to perform a smurf attack on IPv6 network. This tool sends a whole lot of ICMP Ping requests to the multicast address in IPv6 with the spoofed IP address of the victim. Eventually all the nodes gives echo replies to the victim host making it a DDoS Attack.

To make use of smurf6 in the command line, we first had to install THC-IPV6 which is a complete tool set to attack the inherent protocol weaknesses of IPV6 and ICMP6.The test is performed using ***THC smurf6*** from Attacker's computer to flood the victim webserver.

To increase the force of our attack, we executed the same command on multiple terminals of the kali linux interface. This resulted in the victim server (Raspberry Pi-1) going down within a couple of minutes.

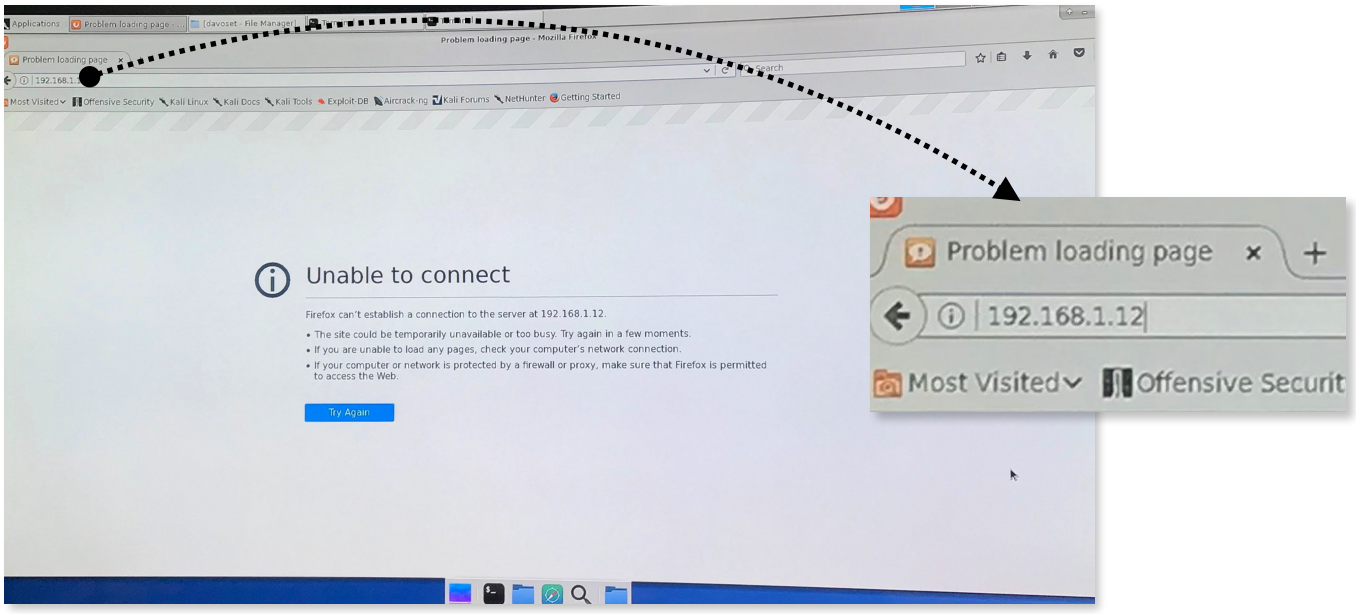

Fig. 5 Image of the web server after the onset of DDoS attack

B(i)THE DETECTION PROCESS

To detect that an attack has been going on our web server, we made use of 2 command line instructions.

- ping <webserver's IP>
- netstat -s

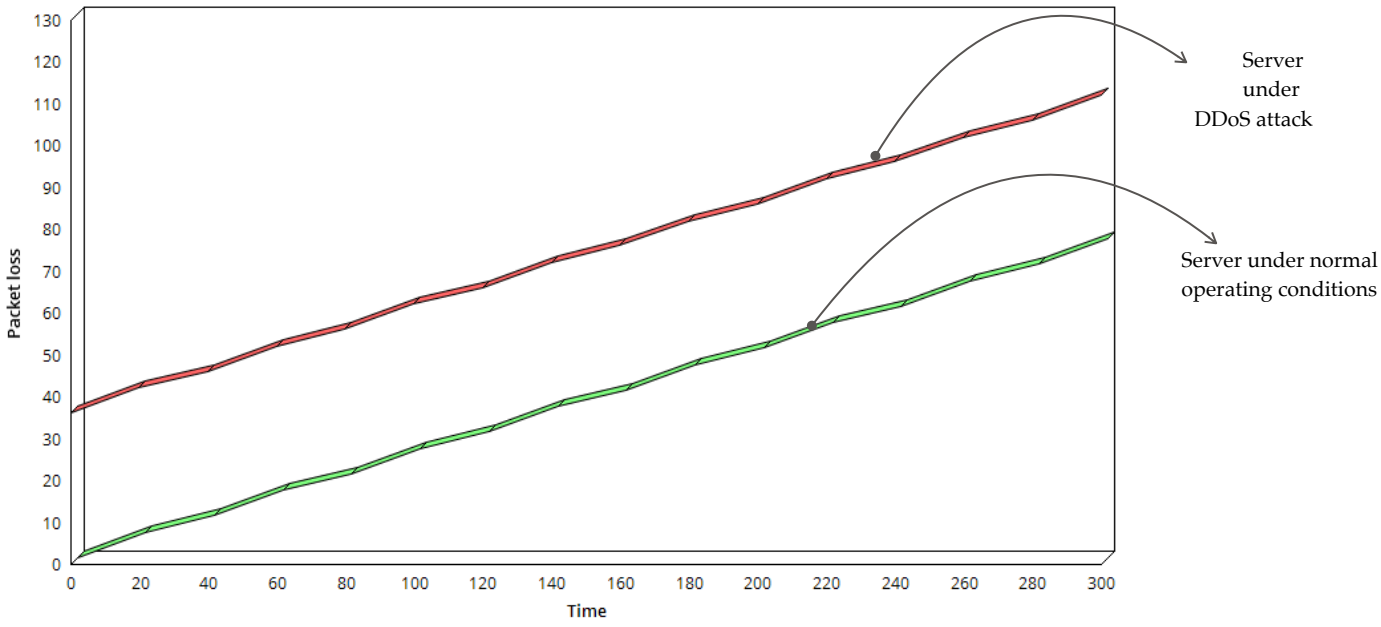

Fig. 6 Packet loss vs time graph using netstat -s

The ping command provided us with the time it took the packet to reach the IP address we provided. It was observed that as soon as we started with the attacking mechanism, the time started increasing gradually (highest value observed: 5.3 ms). One plausible reason for this is congestion. This situation arises when there are too many packets present in the network. Since a number of hosts are responding back to the victim server, the receiver is thereby flooded with requests more than it can handle. The most effective way to reduce congestion is to reduce the load. As the offered load approaches the carrying capacity, bursts of traffic occasionally fill up the buffers inside routers and some packets are lost. The number of packets lost increases in proportion to how much the congestion increases.

Using the netstat - s command gave us a clearer picture of how many packets were being received and how many out of them failed. Using this instruction for 5 minutes (in the intervals of 20 seconds) for both the scenarios; when the server was operating normally and when it was under the attack, we observed that the number of messages failed shot up at the very first instance when the server was under a DDoS attack. To demonstrate this, we have plotted the values we got at different instances of time for both the situations.

B(ii) DETECTION USING WIRESHARK

Wireshark is an open source network packet analyzer. It can be used to examine the details of traffic at a variety of levels ranging from connection-level information to the bits that make up a single packet. Packet capture can provide a network administrator with information about individual packets such as transmit time, source, destination, protocol type and header data.

We used wireshark IO graphs to monitor the request/response packets on our web server. This is one of the basic graphs that are created using the packets available in the capture file. Smurf attack uses ICMP packets to flood and eventually crash the web server. While recording the IO graphs on Wireshark, we observed that the number of ICMP packets increased from 0 to 150000 when the attack was initiated. The number gradually increased with time. We see an abrupt change when the attacks are aborted. Since the botnets are still active the number of packets doesn't reach zero level instantly. However, the number does go to zero after a certain amount of time, the cumulative time required by botnets to become dormant.

Following are some of the statistics we obtained (from a particular IP address) in the duration of 3 minutes while performing the detection process using wireshark.

Time(mins)	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
1	4,06,564	31M	2,03,318	15M	2,03,246	15M
2	4,45,546	34M	2,22,809	17M	2,22,737	17M
3	5,63,036	43M	2,81,554	21M	2,81,482	21M

The following figures depicts the network before and after the DDoS attack. The number of ICMP packets in the network shoot up drastically once the attack kicks in and since our Pi's web server cannot handle this many requests, it stops responding. As evident, a sharp increase is imminent in all the given parameters.

Percent Packets	Packets	Percent Bytes	Bytes	Bits/sec	End Packets	End Bytes	End bits/ sec
6.3	10	1.6	256	19	10	256	19
100	2,39,521	30.8	57,48,540	450K	2,39,521	57,48,540	450k

Note: The number of packets and all other values in the given table are dynamic.The values of the given parameters surge as the time increases and they all start decreasing once we stop the attack.

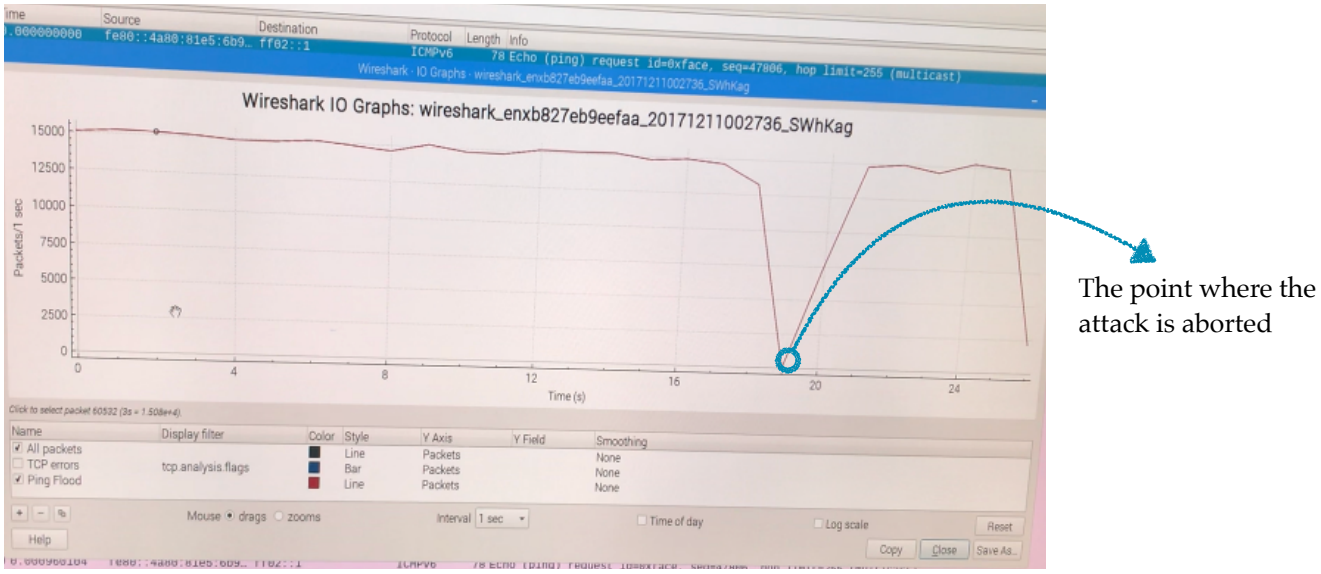

Fig. 7 Real time graph for the web server under DDoS attack using wireshark

C.PREVENTION AND POSSIBLE EXTENSIONS TO THE PROJECT

Although we couldn't implement the prevention mechanism, we did research about several ways with which such attacks can be stopped or mitigated.

The Smurf Attack sounds cute but poses real risks if servers are overwhelmed. Disabled IP broadcasting and reliable detection tools help limit the chance and impact of this attack. Here are a couple of steps to for Smurf attack mitigation:

1. Make sure to block directed broadcast traffic coming into the network configure hosts and routers not to respond to ICMP echo requests.

2. Once you have identify the IP Address from where the DDos is happening we can migrate to countermeasure this attack and block the IP Address. To do that:

iptables -A INPUT 1 -s <IPADDRESS> -j DROP/REJECT

iptables -A INPUT 1 -s <IPADDRESS> -j DROP/REJECT

After firing the above command, KILL all httpd connections to clean your system and than restart httpd service by using the following commands:

killall -KILL httpd

/etc/init/d/apache2 restart

3. If you must allow ping, you can limit the amount of ICMP traffic by implementing Committed Access Rate (CAR). This is where you can limit the amount of traffic identified by an access list. Here's another Cisco IOS example:

```
config t
Access-list 100 permit icmp any {your network} {your subnet} echo-reply
Access-list 100 permit icmp any (your Network) (your Subnet) echo
Interface e1
Rate-limit input access-group 100 512000 8000 8000 conform action transmit exceed action drop
```

This example limits ICMP transmission to 512 Kbps with a transmit burst rate of 8,000 bits. All exceeding packets are dropped. More than one rate-limit command can be added to an interface in order to control other kinds of traffic.

4. In the case of smurf attacks, one method of prevention is to configure the router to block broadcast packets that did not originate from that network. On Linux systems, you can configure the kernel to disregard ICMP ECHO and TIMESTAMP requests that were sent to broadcast or multicast addresses by setting the kernel parameter net.ipv4.icmp_echo_ignore_broadcasts to one.

The growing number of DDoS incidents as well as sophistication in DDoS attacks of present age poses undefeated challenges before security researchers. Possible extensions to this project can include:

1. Researching and implementing algorithms to react against huge volumes of packets generated by DDoS attacks in an automated manner with minimum collateral damage.

2. To detect all types of DDoS attacks reliably and accurately. For example, there is a new type of DDoS attack called degrading DDoS attacks, or non-disruptive DDoS attacks. This type of DDoS attack consumes a large portion of victim network resources but does not stop the network services completely. The traditional DDoS defense mechanisms react poorly to degrading DDoS attacks.

3. Development of an Intrusion Detection System on end server so that it alerts and notify the firewall as soon as an attack is detected.

CONCLUSION

In this project, we executed a real time DDoS smurf attack on a self made web server using Kali Linux. In addition to this, we also implemented the detection process using netstat -s and Wireshark. By making use of these detection tools, we obtained and compared various parameters of the victim web server before and after the onset of attack.

We learned that Raspberry pi, though being pretty small in size is quite powerful when it comes to using it as a web server. Our Pi was able to accept a million request when it was initially set up!

Along the way we also learned how to use Linux and Kali Linux to perform tasks ranging from installing packages to modifying their internal package files or changing configurations of others to suit our needs.

REFERENCES

[1] **For Setting up LAMP Webserver**: https://diyhacking.com/raspberry-pi-web-server/

[2]**For Installing PHPMyAdmin and configuring FTP**: https://www.codedonut.com/raspberry-pi/turn-raspberry-pi-web-server/

[3] **For installing Wordpress**: https://www.raspberrypi.org/forums/viewtopic.php?f=36&t=33246

[4] **For Creating a MySQL database and user via SSH**: https://wiki.gandi.net/en/hosting/using-linux/tutorials/ubuntu/createdatabase

[5] **To check if MySQL database exists**: https://stackoverflow.com/questions/7364709/bash-script-check-if-mysql-database-exists-perform-action-based-on-result

[6] **For installing the Raspberry Pi VNC Server**: https://pimylifeup.com/raspberry-pi-vnc-server/

[7] **List of tools available to simulate a DDoS Attack**:https://serverfault.com/questions/515908/tools-for-simulating-ddos-attacks

[8] **For Performing Apache ab load testing**: https://stackoverflow.com/questions/12732182/ab-load-testing

[9] **For installing THC-IPV6**: https://github.com/vanhauser-thc/thc-ipv6

[10] **Smurf6 attack**: http://kalilinuxtutorials.com/smurf6/

[11] **To learn about various netstat commands**: http://www.thegeekstuff.com/2010/03/netstat-command-examples/

[12] **For installing wireshark on linux**: http://donsthinktank.blogspot.com/2015/07/wireshark-raspberry-pi.html

[13]**To fix wired network interface "device not managed" error in Kali Linux**:https://www.blackmoreops.com/2013/11/25/how-to-fix-wired-network-interface-device-not-managed-error/

[14]**Methods for detecting surf attacks**:http://resources.infosecinstitute.com/theoretical-methodology-detecting-icmp-reflected-attacks-smurf-attacks/#gref

[15] **Guide to Defending Against Distributed Denial of Service Attacks:** https://www.cisco.com/c/en/us/about/security-center/guide-ddos-defense.html

[16] **For Protection against DDoS attacks**: http://www.infosecisland.com/blogview/14788-Protecting-Linux-Against-DoSDDoS-Attacks.html

[17] A.M Sukhov, E.S Sagatov and A.V Baskakov. "Rank distribution for determining the threshold values of network variables and the analysis of DDoS attacks" In *proceedings of the 3rd International conference on "Information Technology and Nanotechnology"*, April 2017