STUDY OF FLOODING BASED DDoS ATTACKS AND THEIR EFFECT USING DETER TESTBED

Daljeet Kaur[1], Monika Sachdeva[2]

[1, 2]Associate Professor, Computer Science and Engineering, Department, SBSSTC, Punjab, India
daljeetkaur617@gmail.com, monika.sal@rediffmail.com

Abstract

Today, Internet is the primary medium for communication which is used by number of users across the Network. At the same time, its commercial nature is causing increase vulnerability to enhance cyber crimes and there has been an enormous increase in the number of DDOS (distributed denial of service attack) attacks on the internet over the past decade. Whose impact can be proportionally severe. With little or no advance warning, a DDoS attack can easily exhaust the computing and communication resources of its victim within a short period of time. Network resources such as network bandwidth, web servers and network switches are mostly the victims of DDoS attacks. In this paper different types of DDoS attacks has been studied, a dumb-bell topology have been created and effect of UDP flooding attacks has been analyzed on web service by using attack tools available in DETER testbed. Throughput of web server is analyzed with and without DDoS attacks.

Index Terms: *Vulnerability, DDoS, availability, confidentiality, throughput*

---***---

1. INTRODUCTION

Denial of Service (DoS) attacks is undoubtedly a very serious problem in the Internet, whose impact has been well demonstrated in the computer network literature. The main aim of DoS is the disruption of services by attempting to limit access to a machine or service instead of subverting the service itself. This kind of attack aims at rendering a network incapable of providing normal service by targeting either the network's bandwidth or its connectivity. These attacks achieve their goal by sending at a victim a stream of packets that swamps his network or processing capacity denying access to his regular clients. There have been some large-scale attacks targeting high profile Internet sites [1–3].

Distributed Denial of Service (DDoS), is a relatively simple, yet very powerful technique to attack Internet resources. DDoS attacks add the many-to-one dimension to the DoS problem making the prevention and mitigation of such attacks more difficult and the impact proportionally severe. DDoS exploits the inherent weakness of the Internet system architecture, its open resource access model, which ironically, also happens to be its greatest advantage.

DDoS attacks are comprised of packet streams from disparate sources. These attacks engage the power of a vast number of coordinated Internet hosts to consume some critical resource at the target and deny the service to legitimate clients. The traffic is usually so aggregated that it is difficult to distinguish legitimate packets from attack packets. More importantly, the attack volume can be larger than the system can handle. Unless special care is taken, a DDoS victim can suffer from damages ranging from system shutdown and file corruption, to total or partial loss of services.

According to the WWW Security FAQ [4] on Distributed Denial of Service (DDoS) attacks: ‘‘A DDoS attack uses many computers to launch a coordinated DoS attack against one or more targets Using client/server technology, the perpetrator is able to multiply the effectiveness of the DoS significantly by harnessing the resources of multiple unwitting accomplice computers, which serve as attack platforms’’. The DDoS attack is the most advanced form of DoS attacks. It is distinguished from other attacks by its ability to deploy its weapons in a ‘‘distributed’’ way over the Internet and to aggregate these forces to create lethal traffic. DDoS attacks never try to break the victim's system, thus making any traditional security defense mechanism inefficient. The main goal of a DDoS attack is to cause damage on a victim either for personal reasons, either for material gain, or for popularity.

DDoS attacks mainly take advantage of the Internet architecture and this is that makes them even more powerful. The Internet was designed with functionality, not security, in mind. Its design opens several security issues that can be exploited by attackers:

- **Internet security is highly interdependent:** No matter how secure a victim's system may be, whether or not this system will be a DDoS victim depends on the rest of the global Internet [5].

Internet resources are limited: No Internet host has unlimited resources that sooner or later can be consumed by a sufficient number of users.

- **Many against a few:** If the resources of attackers are greater than the resources of the victims then the success of the attack is almost definite.
- **Intelligence and resources are not collocated:** Most of the intelligence needed for service guarantees is located in end hosts. At the same time in order to have large throughput high bandwidth pathways are designed in the intermediate network. This way, attackers can exploit the abundant resources of an unwitting network in order to flood a victim with messages.

A Distributed Denial of Service Attack is composed of four elements, as shown in Fig. 1:

- The real attacker
- The handlers or masters, which are compromised hosts with a special program running on them, capable of controlling multiple agents.
- The attack daemon agents or zombie hosts, who are compromised hosts that are running a special program and are responsible for generating a stream of packets towards the intended victim. Those machines are commonly external to the victim's own network, to avoid efficient response from the victim, and external to the network of the attacker, to avoid liability if the attack is traced back.
- A victim or target host.

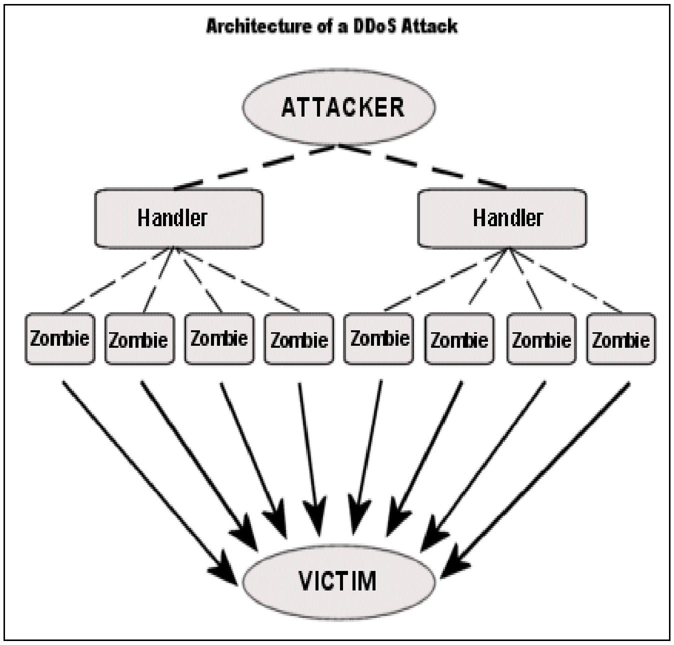

Fig -1: Architecture of DDoS Attack

The following steps take place while preparing and conducting a DDoS attack:

1. Selection of agents: The attacker chooses the agents that will perform the attack. These machines need to have some vulnerability that the attacker can use to gain access to them. They should also have abundant resources that will enable them to generate powerful attack streams. At the beginning this process was performed manually, but it was soon automated by scanning tools.

2. Compromise: The attacker exploits the security holes and vulnerabilities of the agent machines and plants the attack code. Furthermore he tries to protect the code from discovery and deactivation. Self-propagating tools such as the Ramen worm [6] and Code Red [7] soon automated this phase. The owners and users of the agent systems typically have no knowledge that their system has been compromised and that they will be taking part in a DDoS attack. When participating in a DDoS attack, each agent program uses only a small amount of resources (both in memory and bandwidth), so that the users of computers experience minimal change in performance.

3. Communication: The attacker communicates with any number of handlers to identify which agents are up and running, when to schedule attacks, or when to upgrade agents. Depending on how the attacker configures the DDoS attack network, agents can be instructed to communicate with a single handler or multiple handlers. The communication between attacker and handler and between the handler and agents can be via TCP, UDP, or ICMP protocols.

4. Attack. At this step the attacker commands the onset of the attack. The victim, the duration of the attack as well as special features of the attack such as the type, length, TTL, port numbers etc, can be adjusted. The variety of the properties of attack packets can be beneficial for the attacker, in order to avoid detection.

2. EVALUATION IN TESTBED

In order to analyze the effect of DDoS attacks on web service, we have performed a number of experiments in emulated environment on the DETER (Defense Technology Experimental Research) test-bed using SEER (Security Experimentation EnviRonment) GUI BETA6 environment [8][9]. The test bed is located at the USC Information Sciences Institute and UC Berkeley, and allows security researchers to evaluate attacks and defenses in a controlled environment. To setup a satisfactory experiment for analyzing DDoS effect, we should consider topology, legitimate traffic and attack traffic.

2.1 Topology

We have used dumb-bell topology (Shown in Figure 2) for creating traffic in our experiments in which R1, R2, R3 and R4 are routers, node S is server and L1-L20 are clients. They send legitimate requests to server S via router R1 and R2. The bandwidth of all links is set to be 100Mbps, and the bandwidth of bottleneck link (R1-R2) is 1.5Mbps. Node A1 in topology acts as attacking node and it sends attack traffic to server S via router R1 and R2. The link between R1 and R2 is called bottleneck link. The purpose of attack node is to consume/congest the bandwidth of bottleneck link so that legitimate traffic could not get accessed by the server S.

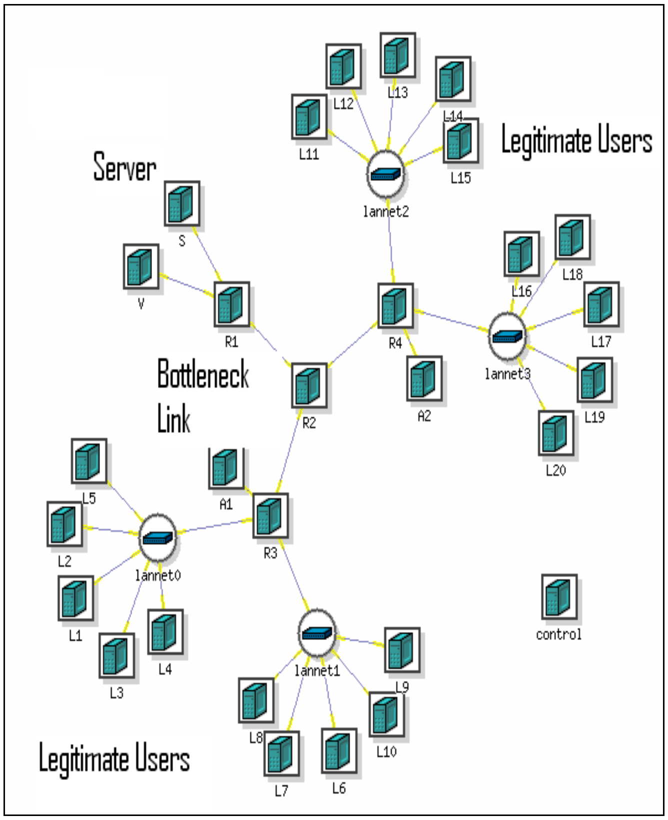

Fig -2: DETER Experiment Topology

We have generated a random network consist of HTTP clients, servers and attack source. In our emulated network, multiple legitimate clients connected with server and one attack source is used as DDoS flooding attacker. This emulates the real situation of DDoS flooding attack.

2.2 Legitimate Traffic

The typical traffic in current Internet is HTTP, FTP and stream traffic. In our experiments HTTP traffics is used. We have used L1-L20 legitimate client nodes which send requests to the server S for 30 seconds with thinking time Minmax (0.01-0.1).

The basic emulation parameters used in experiments are outlined in Table 1. The emulation time is 90 seconds. The total no. of legitimate clients are 20, and all of them send requests to server S at time 1 - 30 second. The attacking nodes A1 sends attack traffic at time 31st- 60th second and again send legitimate requests at time 61st to 90th .

```
set ns [new Simulator]
source tb_compat.tcl
#Create the topology nodes
foreach node { V S R1 R2 R3 R4 L1 L2 L3 L4 L5 L6 L7 L8
L9 L10 L11 L12 L13 L14 L15 L16 L17 L18 L19 L20 A1 A2
control }
 {
  #Create new node
  set $node [$ns node]
  #Define the OS image
  tb-set-node-os [set $node] FC4-STD
  #Have SEER install itself and startup when the node is ready
  tb-set-node-startcmd [set $node] "sudo python
/share/seer/v160/experiment-setup.py Basic"
}
#Create the topology links
set linkRV [$ns duplex-link $V $R1 100Mb 3ms DropTail]
set linkRS [$ns duplex-link $S $R1 100Mb 3ms DropTail]
set linkRA1 [$ns duplex-link $A1 $R3 100Mb 3ms DropTail]
set linkRA2 [$ns duplex-link $A2 $R4 100Mb 3ms DropTail]
set linkRR3 [$ns duplex-link $R2 $R3 100Mb 3ms DropTail]
set linkRR4 [$ns duplex-link $R2 $R4 100Mb 3ms DropTail]
set linkRR2 [$ns duplex-link $R2 $R1 1.5Mb 0ms DropTail]
set lannet0 [$ns make-lan "$L1 $L2 $L3 $L4 $L5 $R3"
100Mb 0ms]
set lannet1 [$ns make-lan "$L6 $L7 $L8 $L9 $L10 $R3"
100Mb 0ms]
set lannet2 [$ns make-lan"$L11 $L12 $L13 $L14 $L15 $R4"
100Mb 0ms]
set lannet3[$ns make-lan "$L16 $L17 $L18 $L19 $L20 $R4"
100Mb 0ms]
$ns rtproto Static
$ns run
```

Fig -3: DETER Experiment Topology Definition

Parameters	**Value**
Emulation Time	90 seconds
No. of legitimate Web clients	20
No. of attack source	01
Access bandwidth	100,50 Mbps per client
Bottleneck bandwidth	1.5Mbps
Legitimate request time	1-30 and 61-90

	seconds
Attack period	31st -60th second
Attack type	DDoS packet flooding

Table -1: Basic Parameters Used During Emulation

2.3 Attack Traffic

We have used packet flooding attack to generate DDoS attack. Node A1 launches attack towards S and thus consumes bandwidth of bottleneck in link R1-R2. UDP protocol is used for launching attacks. Further attack types used are Flat, Ramp-up, Pulse and Ramp-pulse shown in following figures. In our experiments attack traffic from A1 starts at 31st second and stops at 60th second, then we have analyzed effect of DDoS attack on web service. In our emulation experiment, we have generated following flooding attack types:

- **Flat Attack:** The high rate is achieved and maintained till the attack is stopped.
- **Ramp-up Attack:** The high rate is achieved gradually within the rise time specified and is maintained until the attack is stopped.
- **Pulse Attack:** The attack oscillates between high rate and low rate. It remains at high rate for high time specified and then falls to low rate specified for the low tie specified and so on.
- **Ramp-pulse Attack:** It is a mixture of Ramp-up, Ramp-down and Pulse attack.

Attack Type	Flooding	Flooding	Flooding	Flooding
Attack Source	A1	A1	A1	A1
Attack Target	S	S	S	S
Protocol	UDP	UDP	UDP	UDP
Length Min	50	50	100	50
Length Max	100	50	150	50
Flood Type	Flat	Ramp-up	Pulse	Ramp-pulse
High Rate	500	300	200	200
High Time	100	5000	5000	5000
Low Rate	300	100	50	50
Low Time	0	7000	4000	7000
Rise Shape	0	1.0	0	1.0
Rise Time	0	10000	0	10000
Fall Shape	0	0	0	1.0
Fall Time	0	0	0	10000
Sport Min	57	57	57	57
Sport Max	57	57	57	57
Dport Min	1000	1000	1000	1000
Dport Max	2000	2000	2000	2000
TCPFlags	SYN	SYN	SYN	SYN

Table -2: Attack Traffic Parameters Used in Experiment

3. RESULTS AND DISCUSSIONS

During a DDoS attack, bottleneck link (R1-R2) is attacked to force the edge router at the ISP of victim end to drop most legitimate packets. Now, packet flooding attack is being launched by A1 node towards victim S. The attack has been launched by using UDP protocol. And further classification of attack may be flat, rampup, pulse, ramppulse as demonstrated by Table 2. We have created following emulation scenarios and the performance of web server in terms of throughput is analyzed without attack in Figure 4 and with attack in Figure 5 - Figure 8.

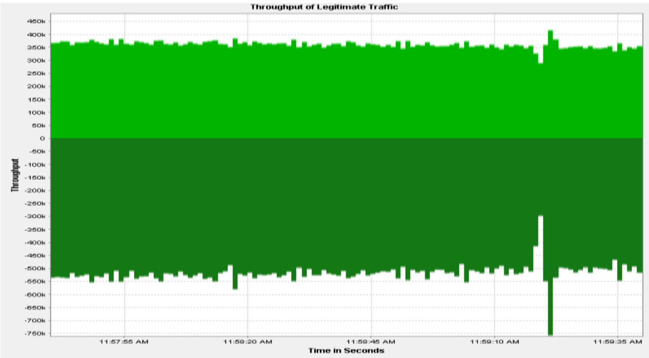

Fig -4: Throughput of legitimate traffic at node S

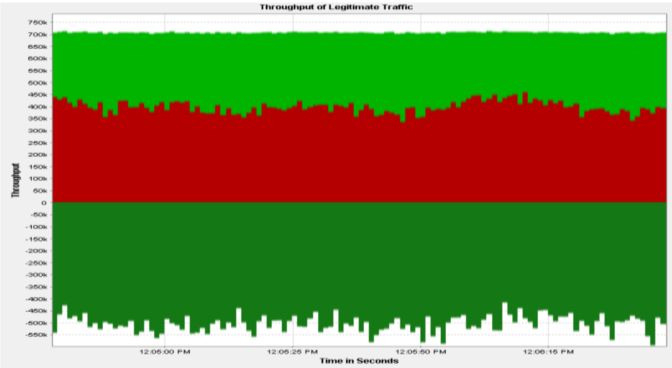

Fig -5: Throughput during UDP flat attack

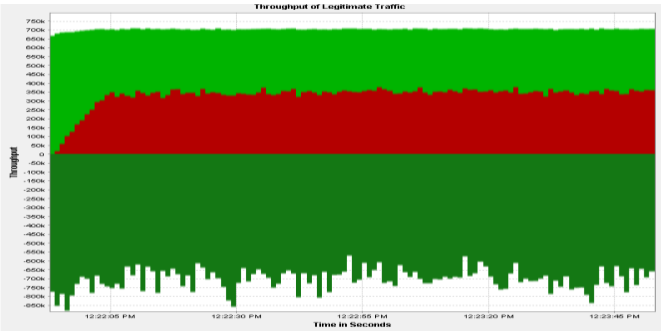

Fig -6: Throughput during UDP Ramp-up attack

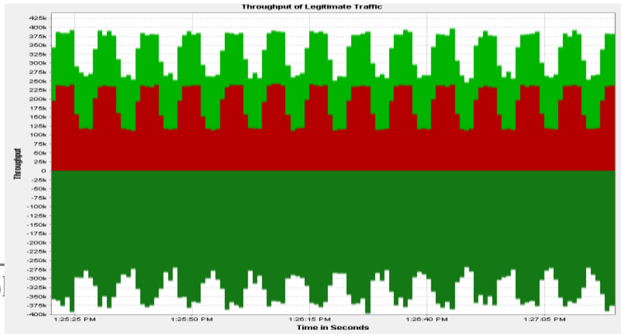

Fig -7: Throughput during UDP Pulse attack

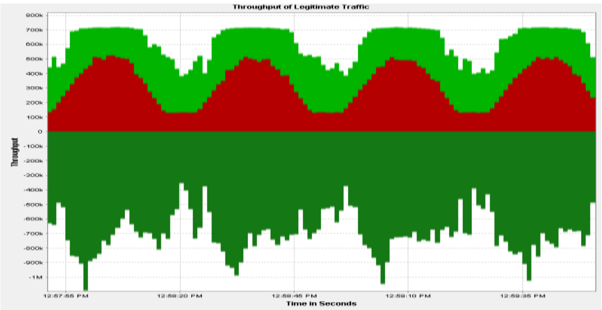

Fig -8: Throughput during UDP Ramp-pulse attack

CONCLUSIONS

There is alarming increase in the number of DDoS attack incidents. Not only, DDoS incidents are growing day by day but the technique to attack, botnet size, and attack traffic are also attaining new heights. Effective mechanisms are needed to elicit the information of attack to develop the potential defense mechanism. DETER testbed allows to carry the DDoS attack experiment in a secure environment. It also allows creating, plan, and iterating through a large range of experimental scenarios with a relative ease. We pointed out the possibility of DDoS attacks on web application by analyzing the characteristics of web application. DDoS attacks are launched on web server and analyzed throughput of legitimate traffic by using different protocols by Emulating attack scenarios.

ACKNOWLEDGEMENTS

We would like to express our gratitude to all those who gave us the possibility to complete this experimental work. We are extremely thankful to all the colleagues and faculty members for their constructive criticism and guidelines.

REFERENCES:

[1] CERT Coordination Center, Denial of Service attacks, Available from <http://www.cert.org/tech_tips/denial_of_service.html>.
[2] Computer Security Institute and Federal Bureau of Investigation, CSI/FBI Computer crime and security survey 2001, CSI, March 2001, Available from <http://www.gocsi.com>.
[3] D. Moore, G. Voelker, S. Savage, Inferring Internet Denial of Service activity, in: Proceedings of the USENIX Security Symposium, Washington, DC, USA, 2001, pp. 9–22.

[4] L.D. Stein, J.N. Stewart, The World Wide Web Security FAQ, version 3.1.2, February 4, 2002, Available from <http://www.w3.org/Security/Faq>.
[5] Kevin, J. and George, M. (2001), "Trends in Denial of Service Attack Technology", http://www.cert.org/archive/pdf/DoS trends.pdf.
[6] CIAC Information Bulletin, L-040: The Ramen Worm, 2001, Available from http://www.ciac.org/ciac/bulletins/ l-040.shtml>.
[7] CERT Coordination Center, CERT Advisory CA-2001-19 _Code Red_ worm exploiting buffer overflow in IIS indexing service DLL, Available from <http://www.cert.org/advisories/ CA-2001-19.html>.
[8] Benzel, T., Braden, R., Kim, D., Neuman, C., Joseph, A., Sklower, K., Ostrenga, R. and Schwab, S. (2006) "Experiences With DETER: A Testbed for Security Research", 2nd IEEE TridentCom Conference.
[9] Mirkovic, J., Hussain, A., Wilson, B., Fahmy, S., Reiher, P., Thomas, R., Yao, W., and Schwab, S. (2007), "Towards User-Centric Metrics for Denial-of-Service Measurement", Workshop on Experimental Computer Science.

BIOGRAPHIES:

Daljeet Kaur has done B.Tech computer science and engineering from Sant Longowal Institute of Engg and Tecnology, Longowal. She finished her M.Tech from Punjab Technical University, Jalandhar in 2012.

Monika Sachdeva has done B.Tech computer science and engineering from National Institute of Technology NIT, Jalandhar in 1997. She finished her MS software systems from BITS Pilani in 2002. In 2012, she finished her PhD from Guru Nanak Dev University, India.