RESEARCH ARTICLE OPEN ACCESS

Distributed Denial Of Service Attack Techniques: Analysis, Implementation And Comparison

Prof Tushar D. Kolhe, Prof. Minal T. Kolhe
Assistant Professor Department of Computer Engineering K.C.E.S'S C.O.E.I.T. Jalgaon (MH)
Assistant Professor Department of Computer Engineering K.C.E.S'S C.O.E.I.T. Jalgaon (MH)

Abstract
A denial of service attack (DOS) is any type of attack on a networking structure to disable a server from servicing its clients. Attacks range from sending millions of requests to a server in an attempt to slow it down, flooding a server with large packets of invalid data, to sending requests with an invalid or spoofed IP address. In this paper we show the comparative analysis of various types of attacks: namely Ping of Death, Connection Flood, TCP SYN Flood, Distributed DOS and others. This paper will demonstrate the potential damage from DOS attacks and analyze the ramifications of the damage. The paper concludes with suggested mitigation methods for some of the discussed attacks.

Keywords- DDOS attack, Spoof attack, TCP-SYN, Flooding, PUSH-ACK.

I. INTRODUCTION

Denial of services attacks (DOS) is a constant danger to modern day Servers. DOS has received increased attention as it can lead to a severe lost of revenue if sites are taken offline for a substantial amount of time. In a denial-of-service (DoS) attack, an attacker attempts to prevent legitimate users from accessing information or services[1][2]. By targeting computers and its network connection, or the computers and network of the sites, an attacker may be able to block accessing of emails, websites, online accounts (banking, etc.), or other services that rely on the affected computer. In a distributed denial-of-service (DDoS) attack, an attacker may use a particular computer to attack another computer. By taking advantage of security vulnerabilities or weaknesses, an attacker could take control of that particular computer. He or she could then force that computer to send huge amounts of data to a website or send spam to particular email addresses. The attack is "distributed" because the attacker is using multiple computers, to launch the denial-of-service attack. Many types of DDOS attacks are prevalent. The paper discusses the methodologies, signs and possible preventions of the existing known attacks.

II. DDOS ATTACK MECHANISM

The DDoS attack operates through a client machine by hacking into weakly secured computers. This is done by searching and finding well-known defects in standard network service programs and commonly weak configurations in known operating systems. But before that attacker can start, the attacker scans these systems looking for vulnerabilities. Unfortunately, this phase very much favours the attackers. The attacker uses computer systems and network port openings to gain access. The more ports that are open, the more points of

vulnerability To determine which ports are open on a given system, a program called port scanner is used[2]. A port scanner runs through a series of ports to see which ones are open. Usually a machine in TCP/IP stack has 65,535 TCP ports and 65,535 UDP ports. The number of ports combined has a potential doorway into the system. Normally, major services listen on fixed port number with the list of open ports on a target system. Using this information, the attacker can get an idea of which services are in use by checking RFC 1700, "Assigned numbers". In the Windows environment, one good scanner is called Scan port. This is a fairly basic port scanner but it enables the attacker to specify both the range of addresses and range of ports to scan. On the Unix side, the best scanner is Nmap. This program scans for open ports by sending packets to the target system to interact with each port. What type of packets is sent and how does interaction happen depend on type of scan being conducted. Some of the types of scan are as follows.

- TCP Connect: Completes the three-way handshake with each scanned port.
- TCP Syn: Only sends the initial SYN and awaits SYN-ACK response to determine if the port is open.
- UDP scan: Sends a UDP packet to target ports to determine if a UDP service is listening.
- Ping: Sends ICMP Echo request to every machine on the target network, for locating live hosts. After the vulnerability scan is done on the target system, a list of vulnerabilities is given to the attacker could exploit. The reason

behind the scan is to automate the process of connecting to a target system and checking to see if the vulnerabilities are present. Another scan tool called Nessus scans random IP addresses to find a known vulnerability. After the scan, a list of victim systems is created that shares the same common vulnerability.

After the scan, the attacker chooses a number of machines to be involved in the attack. These systems are also known as handlers or masters[2]. Now the attacker can find a way to gain access and have significant control over these machines. Most common method is using Stack Based buffer overflow attack. Any application or operating system component that is poorly written could have this problem. A buffer overflow attack happens when an attacker tries to store too much information in an undersized receptacle. Buffer overflow takes advantage of the way in which data is stored by computer programs. When a program calls a subroutine, the function variable and the subroutine returns address pointers stored in a logical data structure known as stack. A stack is a portion of memory, which stores information about the current program needs and contains the address where the program returns after the subroutine has completed execution.

When the buffer is overflowed, the data placed there goes into neighboring variable space and eventually into the pointers space. To cause the attacker's code to be executed, the attacker precisely tunes the amount and content of data to cause buffer overflow and stack to crash. The data the attacker sends usually consists of machine specific byte code to execute a command plus a new address for return pointer. This address points back into the address space of stack, causing the program to run the attacker's instruction when it returns from the subroutine. To help improve the odds that the return pointer will jump to a good place to begin executing the attacker's code, attackers will often prepend a series of NOP (no processing) instruction to their machine level code. A key point is that attacker code will run at whatever privileges the software that is exploited is running at. In most cases, attacker tries to exploit program running as root or administrator privilege. So attacker can easily install backdoor on a system in this way. The captured machines are now instructed to control another set of captured machines. These are called the agents or daemons. By doing this, it ensures a measure of cautiousness on the part of the attacker. Now it is very difficult and impossible to track and find the actual attacker on the Internet. The attacker comprises more systems until the risk of being captured is almost impossible. At the end, the attacker knows the addresses of all the nodes and stores them in a file on his control system. This is later used to attack the victim.

After the attacker breaks into the system, they want to be able to get back into victim's system whenever they want. They could achieve this by installing a backdoor entry as in step 2 or by installing a rootkit (very common in Unix operating system). A rootkit is like a trojan key system files on an operating system. The attacker can replace the login program by overwriting it, but it would be obvious someone messed up the system so a legitimate user could not gain access. To avoid this, the attacker could add some feature into existing login program like allowing someone to have root access without prompting for a password; it would be hard for the administrator to detect their system has been comprised. In general, rootkit provide false information or lie to the administrator to hide what the attacker is doing. The rootkit masks attack activity going on the background.

So finally the actual attack takes place. The attacker on his computer using client software sends instructions to the handlers or nodes to launch a particular attack. These attacks come from variety of different flooding attacks against specific victim.

III. EXISTING ATTACK MECHANISMS

A.1 SYN Flood

A SYN flood is a form of denial-of-service attack in which an attacker sends a succession of SYN requests to a target's system in an attempt to consume enough server resources to make the system unresponsive to legitimate traffic[3]. Normally when a client attempts to start a TCP connection to a server, the client and server exchange a series of messages which normally runs like this:

- The client requests a connection by sending a SYN (synchronize) message to the server.
- The server acknowledges this request by sending SYN-ACK back to the client.
- The client responds with an ACK, and the connection is established.

This is called the TCP three-way handshake, and is the foundation for every connection established using the TCP protocol. A SYN flood attack works by not responding to the server with the expected ACK code. The malicious client can either simply not send the expected ACK, or by spoofing the source IP address in the SYN, causing the server to send the SYN-ACK to a falsified IP address - which will not send an ACK because it "knows" that it never sent a SYN. The server will wait for the acknowledgement for some time, as simple network congestion could also be the cause of the missing ACK, but in an attack increasingly large numbers of half-open connections will bind resources on the server until no new connections can be made, resulting in a denial of service to legitimate traffic. Some systems may also malfunction badly or even crash if other operating system functions are starved of resources in this way.

A.2 ICMP flood, ping flood, smurf attack

In a smurfing attack, a network amplifier is used create a flood of traffic to target a victim system. The attack begins with a ping packet sent to some system,

which supports direct broadcast messages known as a network amplifier[4]. A network amplifier is usually a system on the Internet with an incorrect configured network. The source address of the packet is spoofed to be that of the victim system. Spoofing is a way for the attacker to send messages to IP address, which says that the message was from a trusted host. By doing this, all the ping responses are sent to the victim system. Using the network amplifier with 50 hosts, 50 packets can be sent to the victim by just sending one packet. Network amplifier will receive packet by packet until the maximum amount of traffic is sent. This is because the network amplifier itself has a fixed bandwidth connection to the Internet. At the end, the attack will be traced back to the network amplifier and not the attacker.

Smurf attacks rely on a directed broadcast to create a flood of traffic for a victim on a particular IP address. An IP address is made of host address and network address. If the host part of address is all 1's then the packet is destined for broadcast address of the network. For example, if the network IP address of the network were 10.1.0.0 with net mask of 255.255.0.0, the broadcast IP address for the network would be 10.1.255.255. Using 255 consecutively means there is a message for network IP address because host contains 16 consecutive 1s. This in turn will cause every machine on destination LAN to read the packet and send a response.

The packets sent by the attacker are ICMP ECHO REQUESTS. Normally if the packet's destination network router allows direct broadcasts, all destination LANs will receive the packet. Once received, these machines will then send a ping response. By sending 1 packet, thousands of response packets can be sent. If the first ping response were from spoofed address then all ping responses from the network would be sent to the spoofed address. The number of response packets will increase with more machines on the network that allow direct broadcasting. Using this idea an attacker can conduct a smurfing attack.

A similar attack to smurfing is the fraggle attack. Fraggle is similar that the attacker sends packets through network amplifier but differ by using UDP ECHO packets rather than ICMP ECHO packets. The attack begins with packets sent to IP broadcast address. The destination is UDP port set to a service, which can send the response. The service that receives the packet just sends the packet back exactly as received. By doing this, all machines will echo UDP traffic back causing a flood of the victim's system expresses the probability of X item sets and Y item sets appear in D affair at the same time.

A.3 UDP Flooding:

User Datagram Protocol (UDP) is a connectionless protocol. When sending data packets through UDP, no handshake is required between the sender and receiver. The receiving party will receive packets to process. If a large number of UDP packets are sent, this could cause the victim system to be saturated. This in turn would reduce the bandwidth amount available for legitimate users on the system[5].

When the attacker uses UDP flood attack, UDP packets are sent to either random or specified ports on a victim system. Most of the time they are sent to random ports. When the packets are sent, it causes the victim system to process the incoming data. The system then has to determine which application sent the request. If no applications were running on targeted port, the victim system would send out ICMP packet indicating the destination port is unreachable. As with smurfing, UDP flooding uses spoofed IP address when sending the attacking packet. By doing this, the return packets are sent to another system with spoofed address and not sent back to zombie systems. Another side effect of UDP flood attacks is that these attacks can fill the bandwidth connection around the victim system causing those systems to experience problems with their connectivity.

A.4 Push –Ack Attack

In the TCP protocol, packets that are sent to a destination are buffered within the TCP stack and when the stack is full, the packets get sent on to the receiving system. However, the sender can request the receiving system to unload the contents of the buffer before the buffer becomes full by sending a packet with the PUSH bit set to one. PUSH is a one-bit flag within the TCP header. TCP stores incoming data in large blocks for passage on to the receiving system in order to minimize the processing overhead required by the receiving system each time it must unload a non-empty buffer. The PUSH + ACK attack is similar to a TCP SYN attack in that its goal is to deplete the resources of the victim system. The attacking agents send TCP packets with the PUSH and ACK bits set to one. These packets instruct the victim system to unload all data in the TCP buffer (regardless of whether or not the buffer is full) and send an acknowledgement when complete. If this process is repeated with multiple agents, the receiving system cannot process the large volume of incoming packets and it will crash.

A.5 Low-bandwidth HTTP denial of service attacks

An undefended modern web server is a surprisingly vulnerable target for very simple HTTP attacks such as the Slowloris script. Slowloris works by opening connections to a web server and then sending just enough data in an HTTP header (typically 5 bytes or so) every 299 seconds to keep the connections open, eventually filling up the web server's connection table. Because of its slow approach, it can be a devious attack, remaining under the radar of many traffic-spike attack detection mechanisms[6]. Against a single, typical web server running Apache 2, Slowloris achieves denial-of-service with just 394 open connections2. Like Slowloris, the Slowpost attack client uses a slow, low-bandwidth approach.

Instead of sending an HTTP header, it begins an HTTP POST command and then feeds in the payload of the POST data very, very slowly. Because the attack is so simple, it could infect an online Java-based game, for instance, with millions of user, then becoming unwitting participants in an effective, difficult-to-trace, low bandwidth DDoS attack.

A third low-bandwidth attack is the HashDos attack. In 2011, this extremely powerful DoS technique was shown to be effective against all major web server platforms, including ASP.NET, Apache, Ruby, PHP, Java, and Python3. The attack works by computing form variable names that will hash to the same value and then posting a request containing thousands of the colliding names. The web server's hash table becomes overwhelmed, and its CPU spends all its time managing the collisions. The security professionals exploring this attack demonstrated that a single client with a 30 Kbps connection could tie up an Intel i7 core for an hour. They extrapolated that a group of attackers with only a 1 Gbps connection could tie up 10,000 i7 cores indefinitely. If a web server is terminating SSL connections, it can be vulnerable to the SSL renegotiation attack. This attack capitalizes on the SSL protocol's asymmetry between the client and server. Since the server must do an order of magnitude more cryptographic computation than the client to establish the session, a single SSL client can attack and overwhelm a web server with a CPU of the same class. Rounding out the category of low-bandwidth attacks are simple HTTP requests that retrieve expensive URLs. For example, an attacker can use automated reconnaissance to retrieve metrics on download times and determine which URLs take the most time to fetch. These URLs can be then be distributed to a small number of attacking clients.

Such attacks are very difficult to detect and mitigate, turning any weak points in an application into a new attack vector

IV. DDOS DETECTION AND POSSIBLE MITIGATION METHODS

Distributed denial-of-service (DDoS) attack types have moved up the OSI network model over time, climbing from network attacks in the 1990s to session attacks and application layer attacks today. Network attacks include DDoS variants such as SYN floods, connection floods, or ICMP fragmentation. Session attacks, which target layers 5 and 6, include DNS and SSL attacks. Application attacks at layer 7 represent approximately half of all attacks today. Finally, though layer 7 tops the OSI model, attacks are now moving into business logic, which often exists as a layer above the OSI model. But even with these changes in the current threat spectrum, organizations must continue to defend against network and session attacks, too.

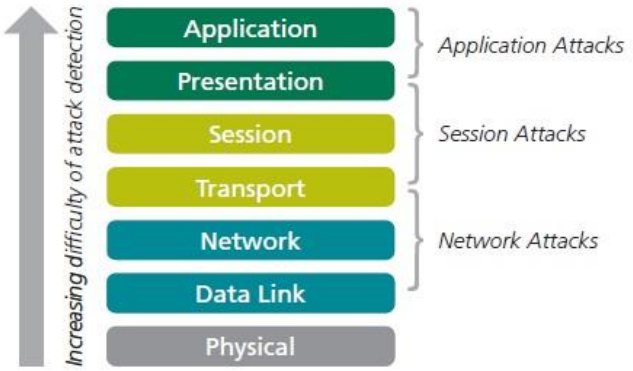

Fig 1. DDoS attacks target many layers of the OSI network model.

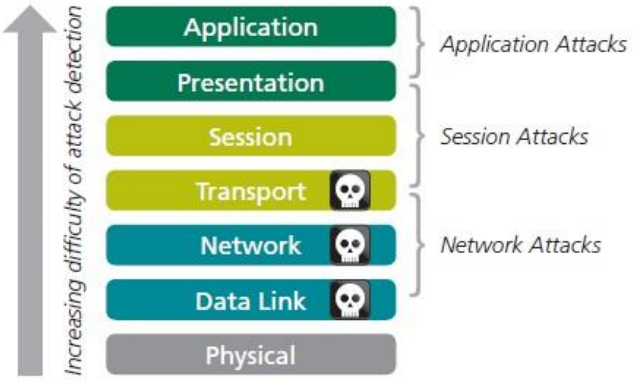

Fig 2. Network attacks target layers 2 through 4.

The most basic network attacks attempt to overwhelm a defensive device with sheer volumes of traffic. Sometimes these volumetric attacks are designed to overload the connections-per-second (CPS) capacity (e.g., the ramp-up rate). Another, slightly more sophisticated attack method is to establish many legitimate connections (a connection flood) to overwhelm the memory of any stateful defensive devices so they lose the ability to accept legitimate connections. Listed below are some of the methodologies which can be instrumental in detecting and mitigating a suspected DDOS attack

A.1 Counter measure against SYN Flood:

One of the more well-known countermeasures against a SYN flood is the use of "SYN cookies", typically used in DDoS engines and load balancers to create another level of protocol security for Denial of Service attacks. A SYN cookie is a specific choice of initial TCP sequence number by TCP software and is used as a defence against SYN Flood attacks.[7]

In normal operation, a Client sends a SYN and the Server responds with a SYN+ACK message, the server will then hold state information in the TCP stack while waiting for Client ACK message. A simple SYN flood (using suitable software) will generate SYN packets which would consume all available TCP memory as the server must maintain state for all half-open connections. And since this state table is finite the server will no longer accept new TCP connections and thus fail or deny service to

the user. This is highly leveraged attack since a very small amount of bandwidth and CPU can exhaust the resources on a large number of servers. The TCP sequence number at the commencement of a TCP sequence is normally a randomised choice. The TCP sequence is what NMAP uses to identify the OS since it 'knows' the some OS's do not have high quality randomisation and NMAP uses algorithms to analyse the ISN to 'guess' the OS. This is part of the functions of a PIX/ASA firewall, it will improve the randomness of the ISN to ensure If the ACK response is not correct the TCP session is not created. The effect is that SYN floods will no longer consume resources on servers or load balancers/ This is especially true in high bandwidth environments such as Data Centres.

By specifically calculating the TCP sequence number with a specific, secret math function in the SYN-ACK response, the server does not need to maintain this state table. On receipt of the ACK from the Client, the TCP sequence number is checked against the function to determine if this is a legitimate reply. If the check is successful, then the server will create the TCP session and the user connection will proceed as normal

A.2 Countermeasures against HTTP/UDP Flood

An effective defense against an HTTP flood can be the deployment of a reverse proxy[8]. In particular a collection of reverse proxies spread across multiple hosting locations, deciding which packets are allowed to where the real web server is. By deploying many proxies, the crush of incoming traffic is split into fractions, lessening the possibility of the network becoming overwhelmed. Deploying this type of architecture can be done in the scramble after an attack has begun, or baked into the network architecture of a web site as a preventative defense. The key to fast denial of UDP floods historically has been the default-deny security posture. Any packets that do not match a defined virtual server are dropped as quickly as possible, thus mitigating UDP floods. No UDP packets ever reach HTTP-based applications.

A.3 Countermeasures against PUSH –ACK Attack

Solutions built atop a full-proxy architecture can be active security agents because their architecture makes them part of the flow of traffic, not simply devices sampling that traffic. Products that are full proxies provide inherently better security because they actively terminate the flow of data, essentially creating an "air gap" security model inside the product, thereby preventing attacks like PUSH-ACK attacks. With full proxies, traffic coming from the client can be examined before it is sent on its way to the application tier, ensuring that malicious traffic never passes the proxy barrier. Traffic returning from the server can be fully examined before it is deemed acceptable to pass back to the client, thereby ensuring that sensitive data such as credit card or Social Security numbers are never passed across the proxy barrier. A full-proxy can mitigate PUSH and ACK floods. Because it will be a part of every conversation between every client and every server, it can recognize packets that do not belong to any valid flow, such as typical PUSH and ACK flood packets. These are dropped quickly and never pass beyond the ADC.

A.4 Countermeasures against Smurf and Tear Drop attacks

One of the few layer 3 attacks still in use today is the ICMP flood. Often these floods are triggered by amplifying ICMP echo replies from a separate network to a target host. This can be mitigated by limiting the rates of all ICMP traffic and then dropping all ICMP packets beyond the limit. The limit is adjustable by the operator. [9][10][11]

A teardrop attack exploits an overlapping IP fragment problem in some common operating systems. It causes the TCP reassembly code to improperly handle overlapping IP fragments. It can be handled by correctly checking frame alignment and discarding improperly aligned fragments. Teardrop packets then are dropped and the attacks are mitigated before the packets can pass into the data center.

V. CONCLUSION

In this review research, we have given a comparative analysis of various DDOS attack mechanisms. We have also discussed possible methods of detection and also some possible methods of mitigation of some of these attacks. However, these attacks are becoming sophisticated by the day and low bandwidth attacks like Slowloris and Pyloris are becoming a serious threat to systems all over the world. Advanced and Strong architectures need to be built in servers with strict protocol validation rules to make sure the modern day systems are protected.

.

REFERENCES

[1] Zhang Chao-yang, Huanggang, *"DOS Attack Analysis and Study of New Measures to Prevent"* International Conference on Intelligence Science and Information Engineering (ISIE), Page(s): 426 – 429, 2011

[2] Douligeris, C. Mitrokotsa, "*A DDoS attacks and defense mechanisms: a classification"* Proceedings of the 3rd IEEE International Symposium on Signal Processing and Information Technology(. ISSPIT), Dec. 2003 *Page(s):190 - 193*

[3] Kavisankar, L. Chellappan, C. *"A mitigation model for TCP SYN flooding with IP spoofing."* International Conference on Recent Trends in Information Technology (ICRTIT), 2011, *Page(s):251 - 256*

[4] Kumar, S. *"Smurf-based Distributed Denial of Service (DDoS) Attack Amplification in Internet*" Second International Conference on Internet Monitoring and Protection, 2007

[5] Xu Rui ; Ma Wen-Li ; Zheng Wen-Ling *"Defending against UDP Flooding by Negative*

Selection Algorithm Based on Eigenvalue Sets".. Fifth International Conference on Information Assurance and Security, 2009 Page(s):342 - 345

[6] Saman Taghavi Zargar, James Joshi, David Tipper "A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks" IEEE Communications Surveys & Tutorials,2013

[7] Bo Hang ;Ruimin Hu *"A novel SYN Cookie method for TCP layer DDoS attack"* International Conference on Future BioMedical Information Engineering, 2009 Page(s): 445 – 448

[8] Martin Mailloux, Hesham Naim and Travis Wayne *"Application Layer and Operating System Collaboration to Improve QoS against DDoS Attack."2008* https://wiki.engr.illinois.edu

[9] *CERT Advisory CA-1997-28 IP Denial-of-Service Attacks"*. CERT. 1998. Retrieved May 2, 2008.

[10] *Windows 7, Vista exposed to 'teardrop attack*'". ZDNet. Retrieved 2011-12-02.

[11] "Microsoft Security Advisory (975497): *Vulnerabilities in SMB Could Allow Remote Code Execution"*. Microsoft.com. Retrieved 2011-12-02.