DENIAL OF SERVICE ATTACKS

Alexandru Enaceanu, acid@rau.ro

Abstract
This paper describes the most common types of DoS, including the latest one, named Distributed Reflection Denial of Service.
The operation of the Internet's TCP protocol is followed by complete explanation on how several types of DoS work.
Bandwidth and CPU load are very important aspects on how the resources are delivered by the servers. Therefore an attack that produces load on any of the two resources – bandwidth and processing power – can cause valid traffic not to obtain useful service, because of the malicious attack.
The crucial fact is that the world is changing rapidly and the world's Internet of today and tomorrow is not the Internet of yesterday. Therefore we must be one step behind (if not forward) any attacker, in order to be prepared and make our servers stay live on the Internet.

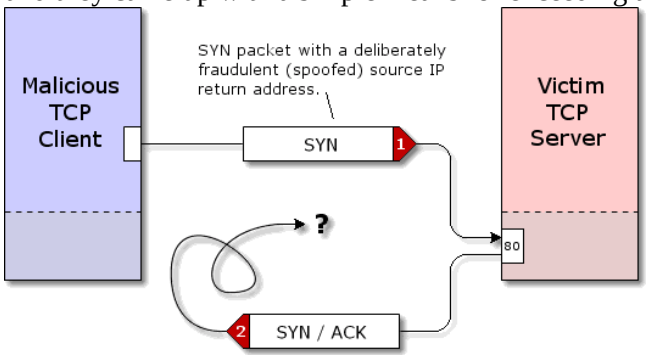
Keywords : DoS, Denial of Service, hijack, DRDoS, Internet attack, vulnerability, TCP/IP, TCP, crack, sniff, routing, router

From the various types of DoS Attacks, we can remember the traditional ones based on the TCP protocol. The easiest method is known as SYN Flood :

The Traditional SYN Flood
Several years ago, a weakness in the TCP connection handling of many operating systems was discovered and exploited by malicious Internet hackers.

As shown in the TCP transaction diagram above, the server's receipt of a client's SYN packet causes the server to prepare for a connection. It typically allocates memory buffers for sending and receiving the connection's data, and it records the various details of the client's connection including the client's remote IP and connection port number. In this way, the server will be prepared to accept the client's final connection-opening ACK packet. Also, if the client's ACK packet should fail to arrive, the server will be able to resend its SYN/ACK packet, presuming that it might have been lost or dropped by an intermediate Internet router.

But think about that for a minute. This means that memory and other significant server "connection resources" are allocated as a consequence of the receipt of a single Internet "SYN" packet. Clever but malicious Internet hackers figured that there had to be a limit to the number of "half open" connections a TCP server could handle, and they came up with a simple means for exceeding those limits:



The packet's "return address" (source IP) can be overridden and falsified. When a SYN packet with a spoofed source IP arrives at the server, it appears as any other valid connection request. The server will

allocate the required memory buffers, record the information about the new connection, and send an answering SYN/ACK packet back to the client.

But since the source IP contained in the SYN packet was deliberately falsified (it is often a random number), the SYN/ACK will be sent to a random IP address on the Internet. If the packet were addressed to a valid IP, the machine at that address might reply with a "RST" (reset) packet to let the server know that it did not request a connection. But with over 4 billion Internet addresses, the chances are that there will be no machine at the address and the packet will be discarded.

The problem is, the server has no way of knowing that the malicious client's connection request was fraudulent, so it needs to treat it like any other valid pending connection. It needs to wait for some time for the client to complete the three-way handshake. If the ACK is not received, the server needs to resend the SYN/ACK in the belief that it might have been lost on its way back to the client.

As you can imagine, all of this connection management consumes valuable and limited resources in the server. Meanwhile, the attacking TCP client continues firing additional fraudulent SYN packets at the server, forcing it to accumulate a continuously growing pool of incomplete connections. At some point, the server will be unable to accommodate any more "half-open" connections and even **valid** connections will fail, since the server's ability to accept **any** connections

It is important to understand that these early spoofed-source-IP SYN attacks were **not** bandwidth consumption attacks. Due to the susceptible nature of most operating systems' TCP/IP protocol handlers, very little inbound bandwidth was required to completely tie-up a TCP server. Rather than consuming the network's "bandwidth resource", the server's "connection resources" were consumed.

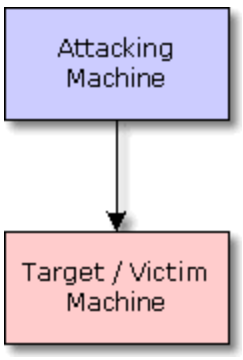
Also notice that this Denial of Service (DoS) attack was not "Distributed." It was a "DoS" attack, not any form of "DDoS" attack. A single, malicious, SYN-generating machine, hiding its Internet address and identity behind falsified source IP SYN packets, could tie-up and bring down a large web site.

Solving the SYN spoofing problem

Operating system vendors responded to spoofed SYN packet DoS attacks by strengthening their TCP "protocol stacks" in various ways. Most of these were quantitative improvements to make their systems less vulnerable, but they did not eliminate the problem. The Unix way to deal with this kind of attack was TCP syncookies (included in the kernel).

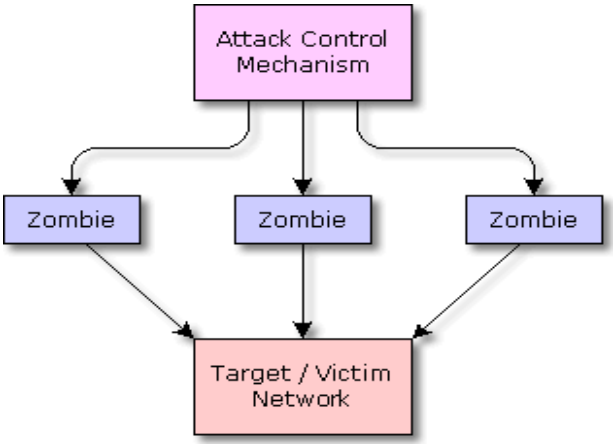
DoS versus DDoS

DoS: The traditional DoS style attack, where a single machine attacks another, can be depicted by this diagram **. . .**



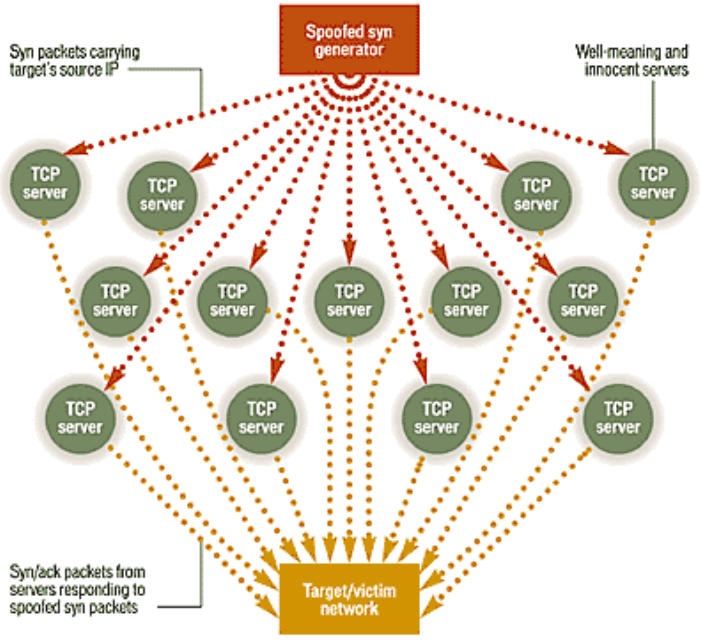
As we saw in the routing diagram above, if the attacking machine enjoys a significantly higher speed Internet connection than the target/victim machine, it could successfully swamp the target's connection bandwidth. Thus, even one well-connected attacking machine can flood a less well-connected target to deny its access to the Internet.

DDoS: Much higher levels of flooding traffic can be generated by focusing the combined bandwidth of multiple machines onto a single target machine or network **. . .**



The diagram above shows the architecture commonly used in distributed denial of service attacks. The operation of a network of compromised machines, containing remotely controlled "Zombie" attack programs, is directed and coordinated by a "Zombie Master" central control agency. When the network of Zombies receives instructions from its Master, each individual Zombie begins generating a flood of malicious traffic aimed at a single target/victim machine or network.

Now, syn floods are getting a whole lot nastier. A new form of syn, called a distributed reflection denial-of-service (**DRDOS**) attack is giving headaches to many network administrators around the world.



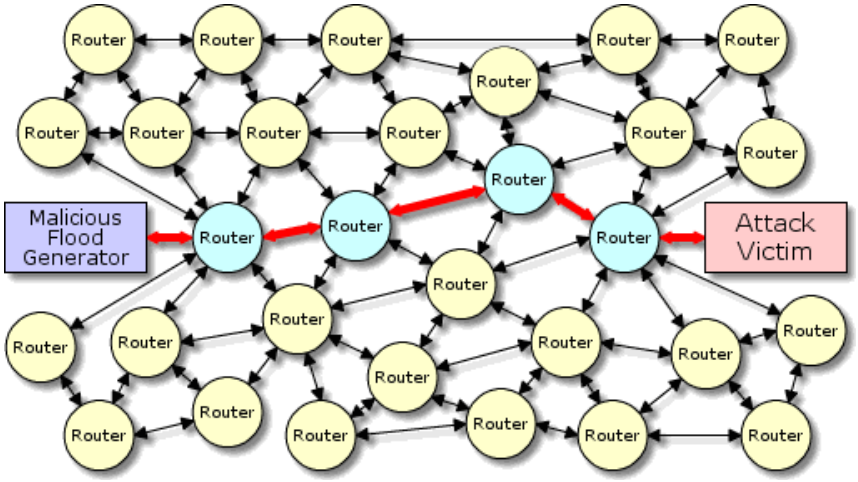
As we can understand from the picture above, DRDoS is the worst type DoS Attack, because it uses valid TCP connections from valid hosts.

Why should we worry?

Why is a reflection attack superior to simply having the malicious hosts directly flood and attack their victim?

Packet path diffusion

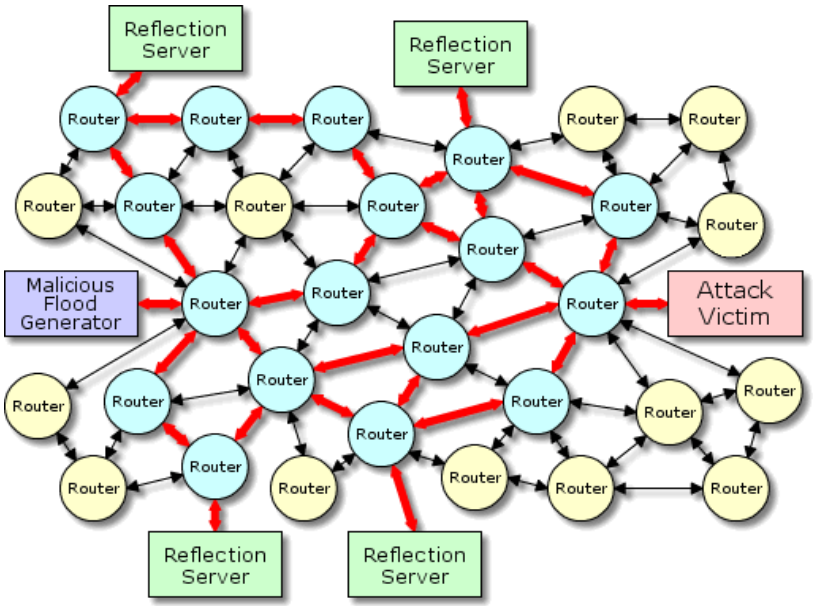
The big win for the attacker is the extreme degree of "packet path diffusion" made possible when attack traffic can be bounced off a large number of intermediate TCP servers. This diagram is a representation of the path of traffic between a single attacker and victim:



The Internet is essentially a large network of individual, interconnected, packet routers. The specific path taken by individual packets moving between any two Internet endpoints may change as a result of local network outages, congested routers, and other factors. However, over a short period of time, most packets will travel along the "best path" — which rarely changes from one packet to the next.

Since Internet routers do not retain any record of previously routed packets, "backtracking" an attack from the victim to the attacker, relies upon the feasibility of manually following a packet flood "upstream" from one router to the previous.

Diffusing the path: Even in the presence of a solid and powerful packet flow, packet path backtracking is a difficult and time consuming manual process. So, imagine what happens when a large number of widely spread packet refection servers are added to the system **. . .**



As this second traffic-routing diagram demonstrates, the addition of innocent reflection servers substantially transforms the attack. Upon leaving an attacking machine, the malicious SYN packets immediately fan out. No longer aimed at the victim, these attack packets are instead being sent to widely spread TCP servers. As we know, these servers are potentially located throughout the entire Internet. Just a few "router hops" away from the attacker, the heavy packet flow will no longer be discernible because it will have diffused into neighboring routers rather than following a single path.

The situation on the receiving end is also significantly changed. Rather than being assaulted by discrete packet flows, one from each attacking machine, the victim is now under a SYN/ACK flood from hundreds, thousands, or even millions of individual, innocent, servers. Although each of those packet flows would be individually harmless to the victim (just as each one is harmless to its individual reflection server), the convergence of these packets arriving from everywhere across the entire Internet, creates an attack that will swamp the victim.

Conclusion : Having a permanent Internet connection gives us a great responsibility of protecting our servers. The DoS Attackers are getting smarter and we should take care not only not to be one of their targets but not to be used for attacking other targets. Therefore, using an active monitoring system in correlation with the latest patches and technologies is a MUST for every network administrator.