Readings 3: The DoS Attack problem

Simon Foucher 260 223 197

 Dos Attacks exploit the bandwidth unbalance that can exist between a victim’s network and compromised machines exploited maliciously to launch an attack. By using this bandwidth to flood the victim’s servers, an attacker can effectively make it difficult for legitimate traffic to go through, and, in the extreme cases, make the victim’s servers crash. It is important to note that DoS attacks do not exploit any network weakness, other than the ones that let the attacker compromise his zombies in order to launch the attack.

 The popularity of DoS attacks have greatly increased after the highly publicized attacks performed against major sites like Yahoo, CNN and Amazon by a 15 year old Montrealer going by the alias ‘Mafiaboy’. In an attempt to test an internet weapon of mass destruction intended to be used in IRC wars, Michael Calse manage to take down some of the biggest sites of the internet in Febuary 2000 using an army of compromised servers [1]. The media publicity generated by the fact that a 15 year old boy was able to destroy these internet giants proved the effectiveness of DoS attacks, and launched a popular wave of interest in DoS attacks, such that we have observed an increase in the availability of a number of user-friendly attack tools. To this day, there is still a lack of effective solutions to defend sites against DoS attacks.

 Generally, there are 2 types of DoS attacks: direct attacks and reflector attacks. In a direct attack, an attacker arranges to send out a large number of attack packets directly toward a victim, using for example TCP SYN packets sent to a victim. The victim responds with SYN-ACK packets to spoofed random source address, then repeat SYN-ACK until giving up. UDP packets may also be used to achieve the same result, being sent to closed ports, in which cases, the victim usually responds with the corresponding ICMP reply error messages. In a reflected attack, the attacker send request packets to innocent intermediary nodes like routers and servers, and spoofs the source address of his requests by the victim’s address. These attacks are a lot more difficult to trace and avoid, since the flooding packets are actually ‘legitimate’ packets sent by non-compromised machines. Because of the legitimate nature of the flooding traffic, these types of attacks cannot be filtered based on address spoofing or other route-based mechanisms.

 There are three lines of defense against DOS attacks: attack prevention, attack detection and filtering, and attack source traceback. Because of the statelessness of the internet, tracing the source of an attacker can be a hard/impossible task. As far as attack prevention goes, one author suggests having ‘cyber-spies’ intercept attack plans, but this is obviously not a complete nor effective way of solving the issue. When developing attacks detection and filtering, one author proposes ratios to quantify the quality of tools: False Positive Ratio, false negative ratio, and normal packet survival ratio. The final line of defense, which seems to be the most promising one, is IP traceback. The challenge lies on identifying the source of packets without relying on the source information provided.

 The best IP traceback solution would be to implement ingress filters on all ISP that are part of the internet. Whenever a packet is detected to have a spoofed return address, it would be dropped, and therefore, the DoS problem would be solved. Of course, as much as this seems like a good idea, ensuring that all ISPs in the Internet install ingress packet filtering is impossible.

 Another solution would be to implement a global internet firewall that would attempt to detect DoS attack behaviors in traffic and automatically drop those packets. This could be done in 2 ways. First, in a route based filtering approach, routers would ensure that each received packet comes from a correct link according to the inscribed source and destination addresses, and the BGP routing information. An issue with this is that dropped packet could have been legitimate due to a recent route change. Of course, one could easily imagine how such a system would fail in case a small website gets big publicity, or an unpredictable external even occurs that drives considerable amount of traffic to the same places (ex: Michal Jackson’s death flooded CNN to a point where DoS flags were send to network admins [2]). Also, it does not provide any help to solve reflected attacks, as the packets in this case are legitimate.

 Another approach is to implement distributed detection devices that would monitor internet traffic. When enough units detect abnormalities, a flag is raised and filters are turned on. There are many issues with implementing this approach, mostly the fact that the devices need to be in constant communication with each other to work. An attacker could easily launch a DoS attack on the DD devices simultaneously to stall them while performing its attack. Or better yes, a new breed of attackers could start sending false alert messages to DDs and have them trigger a false filter alert that would drop legitimate packets. Also, flash crowds on the Internet can trigger false alarms in the detection systems.

 Current approaches include link testing, where upstream links are successively tested to determine where the traffic came from. This approach can fail as soon an upstream ISP admin is unwilling to help. We can also use link flooding; which involves flooding upstream links successively to see if we observe a reduction of the DoS traffic. Even if this solves the problem of unwillingness to help, it is not practical and constituted in itself a DoS attack on router links. We could also implement router logging, but the required overhead would dramatically reduce router performance. An alternate solution is for the router to send low probability ICMP traceback message including information about the adjacent routers along the path to the destination. The main issue with this is that it relies on an input debugging capability which not all fast routers have built in.

 The most promising solution to the DoS problem is to probabilistically mark packets with partial path information as they arrive at routers. Since DoS attacks are composed of a large number of packets, combining the stamped packet would reveal the source path of the attack. Traceback could even be made after an attack has completed. The main design issue would be to add little or no overhead to the router's critical forwarding path. Of course, we could not have all routers append their node ID to a packet as it traverses, because the size of the packets would get arbitrarily large, and could cause fragmentation and trigger unnecessary MTU processes. A better approach would be to reserve a static ‘node’ field on packet headers, and routers would stamp packets with their IP with a given probability. With a large enough sample space, the route between the victim and the attacker could be discovered. This is efficient to implement because it only requires the addition of a write and checksum update to the forwarding path and high-speed routers already must perform these operations efficiently to update the time-to-live field on each hop. An alternative would be to do edge stapling, where we would reserve 2 IP stamp field in packet headers. By doing so, stamped packets would mark network edges that could be reassembled to trace paths.

 Overall, there have not been any significantly good solutions implemented to solve the DoS attack problem. The statelessness of the internet makes it very difficult to trace packets and the frequency of flash crowds makes it virtually impossible to implement effective automated filters. The best approaches would require dramatic changes to the internet infrastructure, and therefore are probably not going to be implemented any time soon.

Additional references used:

*[1] Calce, M & Silverman, C.* **“Mafiaboy: How I Cracked the Internet and Why It's Still Broken**”, Published by Penguin Books Lte, ISBN 0670067482, 277 pages.

*[2] Germino, J.,* **“How Michael Jackson's Death Affected the Internet”,** June 27, 2009, <http://www.associatedcontent.com/article/1886252/how\_michael\_jacksons\_death\_affected.html>