Denial of Service Definition, Types and Distributed Denial of Service **Lara Ghaddar**

The Internet Worm incident during the first week of November 1988 was the first wide spread denial-of-service attack on the Internet. Service was denied since infected hosts were rendered useless because multiple copies of the worm program absorbed the processing capability of these hosts. Until all copies of the worm were removed, these hosts were not available for their intended use.

1. Denial-of-service Definition and Types

The baseline or essential security that every user needs from a computer system is *availability*. Hardware and software must be kept working efficiently or else they become useless. If computer hardware, software, and data are not kept available, productivity can be degraded, even if nothing has been damaged. Denial-of-service can be conceived to include both intentional and unintentional assaults on a system's availability. An *attack*, however, is an intentional act. A *denial-of-service attack*, therefore, is considered to take place only when access to a computer or network resource is *intentionally* blocked or degraded as a result of malicious action taken by another user. These attacks don't necessarily damage data directly, or permanently (although they could), but they intentionally compromise the *availability* of the resources.

Denial-of-service attacks over the Internet can be directed against three types of targets: a user, a host computer, or a network. An attacker must begin a denial-of-service attack by using tools to exploit vulnerabilities and then either obtain unauthorized access to an appropriate process or group of processes, or to use a process in an unauthorized way. The attacker then completes the attack by using methods to destroy files, degrade processes, degrade storage capability, or cause a shutdown of a process or system.

1.1. Destruction - If an attacker obtains access to a user, host, or network files, the attacker could delete or corrupt some or all of these files. The effect could be to deny the use of these files. At the user level, an attacker could delete some or all of the account's files, rendering the account unusable. At the host level, critical system files could be deleted. On Unix systems, this could be files such as the */etc/passwd* file, or files containing the system's programs. All files on the host's hard disk could also be removed, or the disk itself could be reformatted. This would make the host computer inaccessible or unusable to all users. At the network level, network files could be destroyed. The network or some of its services could then be degraded or made unavailable.

1.2. Process Degradation - Denial-of-service could be also be accomplished through overloading processes on a host computer to such a point that the users' ability to use the resource is degraded either by reduced performance, or by the resource becoming unavailable. This can take place in two ways. First, an attacker could connect to a host across the Internet and spawn *multiple processes* on the host to a point where the host can no longer support any new processes. The targeted user, or users, would then not be able to run processes of their own. Such programs are referred to as *fork bombs*. A second method would be to slow the host computer by spawning many processes that consume large amounts of CPU time, causing a *CPU overload*. An attacker does not need to connect to a command interface on a host to cause process degradation. An attacker could instead direct an attack against network processes.

1.3. Storage Degradation – This method of attack is aimed at consuming disk storage capacity on the target host or network of hosts. Since a disk has finite capacity, if an attacker fills up a user's disk quota, or fills up the space available for all users, then the user's account or the entire host, will not be available for use until the *disk full* condition is changed. An attacker can either create too many files for the system, or a few files that are too large. The same is true for a network, where the files may be distributed across multiple computers.

1.4. Shutdowns - The last two categories of denial-of-service attacks are *process shutdown* and *system shutdown* attacks. In these types of attacks, the attacker aims at halting a process, or all processing, on a host or network. If the attacker has privileged access, this could be accomplished by issuing the appropriate commands to kill a process or shutdown the system completely. The *kill* command in Unix is an example of a command that could be used to terminate a process. Exploiting a software bug that causes

the process or system to halt could cause process or system shutdown. In this case, an attacker has knowledge of a set of commands that will crash the process or system.

2. Distributed Denial of Service Attacks and Prospects – Distributed Denial of Service (DDoS) attacks are a relatively new development; they were first widely discussed a few months ago. Attacks can be sent by an individual or can be set up to be sent automatically by programs known as Zombies that may have been installed in various computers in advance of the attack. With Zombies, the attacker sends a single command and they perform the attack. This method more easily isolates the attacker from those who might want to find him/her as the attack itself is coming from completely unrelated computers. These attacks typically exhaust bandwidth, router processing capacity, or network stack resources, breaking network connectivity to the victims.

The perpetrator starts by breaking into weakly secured computers, using well-known defects in standard network service programs, and common weak configurations in operating systems. First, they install software to conceal the fact of the break-in, and to hide the traces of their subsequent activity. For example, the standard commands for displaying running processes are replaced with versions that fail to display the attacker's processes. Then they install a special process, used to remote control the burgled machine. This process accepts commands from over the Internet, and in response to those commands it launches an attack over the Internet against some designated victim site. And finally, they make a note of the address of the machine they've taken over. All these steps are highly automated. A cautious intruder will begin by breaking into just a few sites, then using them to break into some more, and repeating this cycle for several steps, to reduce the chance they are caught during this, the riskiest part of the operation. By the time they are ready to mount the kind of attacks seen recently they have taken over thousands of machines and assembled them into a DDoS network; this just means they all have the attack software installed on them, and the attacker knows all their addresses (stored in a file on their control system).

Now comes time for the attack. The attacker runs a single command, which sends command packets to all the captured machines, instructing them to launch a particular attack (from a menu of different varieties of flooding attacks) against a specific victim. When the attacker decides to stop the attack, they send another single command. Now to go into details of the attacks, while there are variations, they generally take a common form. The controlled machines being used to mount the attacks send a stream of packets. For most of the attacks, these packets are directed at the victim machine. For one variant the packets are aimed at other networks, provoking multiple echoes all aimed at the victim.

The packets used in today's DDoS attacks use forged source addresses; they are lying about where the packet comes from. The very first router to receive the packet can very easily catch the lie; it has to know what the addresses are on every network attached to it, so that it can correctly route packets to them. If a packet arrives, and the source address doesn't match the network it's coming from, the router should discard the packet. This style of packet checking is called variously Ingress or Egress filtering, depending on the point of view; it is Egress from the customer network, or Ingress to the heart of the Internet. If the packet is allowed past the border, catching the lie is nearly impossible. If you hand a letter to a letter-carrier who delivers to your home, there's a good chance he could notice if the return address is not your own. If you deposit a letter in the corner letter-box, the mail gets handled in sacks, and routed via high-volume automated sorters; it will never again get the close and individual attention required to make any intelligent judgments about the accuracy of the return address.[1] Likewise with forged source addresses on internet packets: let them past the first border router, and they are unlikely to be detected.

From the victim's point of view, the first sign of having problems is when thousands of compromised systems all over the world commence to flood the victim with traffic at once. The first symptom is likely to be a router crash; traffic simply stops flowing between the victim and the Internet. More closely, one or more targeted servers are being overloaded by the small fraction of the traffic that actually gets delivered, but the failures extend much further back. Capturing a sample of the packets flying over the victim’s net, it can be seen that the packet will have the victim’s address as its destination address, and it will have some random number as a source address. There's no trace of the compromised

host that is busy attacking. All that's there is a low-level, hardware address of the last router that forwarded the packet; these low-level addresses are used to handle distribution of packets within a LAN.

2.1. Immediate prospects - What can be done to avoid being part of the problem, what can be done for the victim, what can be done to become a harder target to take down?

First and most important, secure the servers. This is not a complex or difficult procedure. It's easy to prioritize the machines to be secured, to determine which ones need attention most urgently. At the low end, dialup machines are the lowest worry; faster connections are prime targets for people mounting these attacks. Secure the computers.

Second, ensure that packets are being filtered at the point of connection to the Internet, to prevent forged source addresses. This provides protection in both directions; it prevents machines from being used to mount these attacks, if any are broken into, and it prevents some attacks that might help intruders break into.

And a third defensive measure prevents one from being used to mount the smurf attacks that are part of this pattern of DDoS. Smurf attacks send packets to a ``smurf amplifier'' network. This is any network that allows such packets in. These packets come from outside the amplifier net, but are directed to its broadcast address. Such packets aren't used for any legitimate purpose; they are an oversight in the design of the internet protocol. They have a forged source address, to direct all the replies (from all the hosts on the amplifier network) to the victim; each such packet gets repeated by every machine on the network, amplifying the effect of the attack. Packets directed at the broadcast address from outside the net are called IP Directed Broadcast packets, and should be blocked at the border.

The above measures help ensure that systems won't be used to help mount one of these attacks, and they are the place where one can be most effective today. But they do not help in defense against an attack as such; they just ensure that one will not inadvertently assist in one. Today, there are only limited measures one can take to prevent from becoming a victim of DDoS. Such as making oneself harder to target by distributing ones website over multiple server farms, with multiple points of contact to the Internet

2.2. Long-term prospects - Two major developments are currently underway, to help prevent DDoS attacks from remaining unmanageable. The major one is Ingress filtering. If all networks were so well-administered, these attacks could be dealt with relatively quickly; mounting a single attack would deliver to the victim the addresses of all the conquered machines; their owners could be notified, and filters could be put in a place near the machines to block the attacks while the machines are being shut down and fixed. Today some routers can be told to do ingress filtering completely automatically, and nearly all the rest can be manually configured to do it. All routers need to be able to do this filtering. There are still question regarding how realistic this solution is, since it would have to be implemented in every country in the world.

The other half of the solution comes from developing techniques for rapidly tracking these attacks to their source, and notifying the people who need to secure their broken servers, or the providers who need to put blocks in place to shut down an attack at its many sources. Robert Stone of UUNET presented a paper at the October NANOG (North American Network Operators Group) entitled “CenterTrack: An IP Overlay Network for Tracking DoS Floods”. It describes a technique for designing a network, with a handful of extra diagnostic routers, to allow rapid tracking of these floods to their source, even in the face of forged source addresses. UUNET may have pioneered this research, but anyone else who doesn't develop comparable facilities will find their customers fleeing to providers who can. The minimum standard for service has just gone up.[2]

1 – Denial of Service Incidents, www.cert.com

2 - Robert Stone of UUNET presented a paper entitled "CenterTrack: An IP Overlay Network for Tracking DoS Floods", describing a system developed at UUNET to assist in tracking these attacks to their sources quickly, http://www.us.uu.net/gfx/projects/security/centertrack.pdf.

3 - David Dittrich of the University of Washington, http://www.washington.edu/People/dad/.