Network Security and DoS Attacks

0. Document History

Author: Sílvia Farraposo
Laurent Gallon
Philippe Owezarski

Date	Status	Comments
February 2005	Draft	
March 2005	1.0	
April 2005	2.0	

1. Introduction

It was in early 2000, that most people became aware of the dangers of distributed denial of service (DDoS) attacks when a series of them knocked such popular Web sites as Yahoo, CNN and Amazon off the air. More recently, a pair of DDoS attacks nailed The SCO Group's Web site and many people thought that it was a hoax because surely any company today could stop a simple DDoS SYN attack. Wrong.

It has been almost five years now, but DDoS attacks are still difficult to block. Indeed, some DDoS attacks, including SYN, if they are made with enough resources are impossible to stop.

No server, no matter how well it is protected, can be expected to stand up to an attack made by thousands of machines. Indeed, Arbor Networks, a leading anti-DDoS company, reports DDoS zombie (host previously compromised by the attacker, which effectively accomplish the DoS attack) armies of up to 50,000 systems. Fortunately, major DDoS attacks are difficult to make. Unfortunately, minor DDoS attacks are easy to make.

In part, that is because there are so many kinds of DDoS attacks. It may do it by attacking the TCP/IP protocol, it may do it by assaulting server resources, or it could be as simple as too many users demanding too much bandwidth at one time.

Unfortunately, as more and more users add broadband connections without the least idea of how to handle Internet security, these kinds of attacks will only become more common. With this document it is our intention to present how denial of service attacks can occur.

In section 2 of this report data security concepts and a TCP/IP overview are presented. Section 3 describes DoS and DDoS attacks through a presentation of the major attacks of this kind. Section 4 highlights some trend lines that permit to avoid DDoS attacks, or at least to limit their effects in the network being attacked. Finally, section 5 is a conclusion of all this study.

2. A Brief Background

This section is intended to overview some of the main aspects when talking about security and attacks. This includes a broad classification for the attacks, based on how they affect data, and some main protocols of the TCP/IP suite against which the attacks presented are perpetrated or that are used to perpetrate an attack.

2.1. Data Security

Most of the literature about security concepts split the data security into three different parts: integrity, confidentiality and availability.

Integrity is related with the trust that we have about data. So, if someone who is not allowed to manage or to change data is doing that, the integrity is compromised. You can no more trust that your valuable information is true any more.

The information's confidentiality is compromised if a person is able to enter a computer(s) that he is not allowed to. He/she may then get to know information not intended to be available for that person. He/she may even distribute the information.

That latter category - availability - is the one where the attacks discussed in this paper belong. The data's availability is of course important in running a business and huge losses may occur if important information is no longer available as a result of an attack against the computers. Such attacks are often named "Denial-of-Service" (DOS).

2.2. About the TCP/IP protocol

The attacks which are discussed in this paper are all utilizing weaknesses in the implementation of the TCP/IP protocols to make the attacked computer or network stop working as intended. To understand the attacks one has to have a basic knowledge of how these protocols are intended to function.

TCP/IP is the acronym of Transmission Control Protocol/Internet Protocol and is one of several network protocols developed by the United States Department of Defense (DoD) at the end of the 1970s. The reason why such a protocol was designed was the need to build a network of computers being able to connect to other networks of the same kind (routing). This network was named ARPANET (Advanced Research Project Agency Internetwork), and is the predecessor of what we call Internet these days.

TCP/IP is a protocol suite which is used to transfer data through networks. Actually TCP/IP consists of several protocols. The most important are:

- **IP Internet Protocol**
  This protocol mainly takes care of specifying where to send the data. To do that, each IP packet has sender and receiver information [1]. The most common DoS attacks at the IP level exploit the IP packet format.
- **TCP Transmission Control Protocol**
  This protocol handles the secure delivery of data to the address specified in the IP protocol [2].

  Most of the TCP level attacks exploit weaknesses present in the implementations of the TCP finite state machine. By attacking specific weaknesses in applications and implementations of TCP, it is possible for an attacker to make services or systems crash, refuse service, or otherwise become unstable.
- **UDP User Datagram Protocol**
  UDP may be used as an alternative to TCP. The difference is that UDP does not guarantee that data reaches the receiver, since it is connectionless and a protocol

without any packet loss recovery mechanism [3]. On the other hand this protocol has less overhead than the TCP protocol – data transmission is faster. UDP packets are mainly used to perpetrate flooding attacks.

- **ICMP Internet Control Message Protocol**
  ICMP is a subset of the TCP/IP suite of protocols that transmits error and control messages about the network situation between systems [4]. Two specific instances of the ICMP are the ICMP ECHO_REQUEST and ICMP ECHO_RESPONSE datagrams. These two instances can be used by a local host to determine whether a remote system is reachable via the network; this is commonly achieved using the "ping" command. Under certain conditions, flooding ICMP packets might denial services.

A communication through a network using TCP/IP or UDP/IP will typically use several packets. Each of the packets will have a sending and a receiving address, some data and some additional control information. Particularly, the address information is part of the IP protocol – being the other data in the TCP or the UDP part of the packet. ICMP has no separate TCP part – all the necessary information is in the ICMP packet.

In addition to the recipient's address all TCP/IP and UDP/IP communication uses a special port number which it connects to. These port numbers determine the kind of service the sender wants to communicate to the receiver of information.

3. DoS Attacks

DoS attacks today are part of every Internet user’s life. They are happening all the time, and all the Internet users, as a community, have some part in creating them, suffering from them or even loosing time and money because of them. DoS attacks do not have anything to do with breaking into computers, taking control over remote hosts on the Internet or stealing privileged information like credit card numbers. Using the Internet way of speaking DoS is neither a Hack nor a Crack. It is a whole new and different subject.

This section is entirely devoted to denial of service attacks and its variants. Here, we present a broad definition of this kind of network threat, and examples of the most common attacks.

3.1. Definitions

The sole purpose of DoS attacks is to disrupt the services offered by the victim. While the attack is in place, and no action has been taken to fix the problem, the victim would not be able to provide its services on the Internet. DoS attacks are really a form of vandalism against Internet services. DoS attacks take advantage of weaknesses in the IP protocol stack in order to disrupt Internet services.

DoS attacks can take several forms and can be categorized according to several parameters. Particularly, in this study we differentiate denial of service attacks based on where is the origin of the attack being generated at.

"Normal" DoS attacks are being generated by a single host (or small number of hosts at the same location). The only real way for DoS attacks to impose a real threat is to exploit some software or design flaw. Such flaws can include, for example, wrong implementations of the IP stack, which crash the whole host when receiving a non-standard IP packet (for example ping-of-death). Such an attack would generally have lower volumes of data. Unless some exploits exist at the victim hosts, which have not been fixed, a DoS attack should not pose a real threat to high-end services on today's Internet.

DDoS (Distributed Denial of Service) attacks would, usually, be generated by a very large number of hosts. These hosts might be amplifiers[1] or reflectors[2] of some kind, or even might be "zombies" (agent program, which connects back to a pre-defined master hosts) who were planted on remote hosts and have been waiting for the command to "attack" a victim. It is quite common to see attacks generated by hundreds of hosts, generating hundreds of megabits per second floods.

The main tool of DDoS is bulk flooding, where an attacker or attackers flood the victim with as many packets as they can in order to overwhelm the victim. The best way to demonstrate what a DDoS attack does to a web server is to think on what would happen if all the population of a city decided at the same moment to go and stand in the line of the local shop. These are all legitimate requests for service – all the people came to buy something, but there is no chance they would be able to get service, because they have a thousand other people standing in line before them!

DDoS attacks require a large number of hosts attacking together at the same time (see figure 1). This can be accomplished by infecting a large number of Internet hosts with a "zombie". This way, an attacker can be anyone with a certain knowledge and access privilege with the master host (such as the correct password to an Internet Relay Chat (IRC) channel). All he has to do is enter a few commands, and the whole zombie army would wake up and mount a massive attack against the victim of his or hers choice. [5]

The zombie program can be planted on the infected hosts in a variety of ways, such as attachment to spam email, the latest cool flash movie, a crack to a game, or even the game itself. Communication from the zombie to its master can be hidden as well by using standard protocols such as HTTP, IRC, ICMP or even DNS.

[1] System that is able to drastically increase the volume of attacking traffic. This can be accomplished with the use of broadcast addresses as the return destination of packets. Attacks that use amplifiers are also known as magnification attacks.

[2] A reflector is any IP host that will return one or more packets for each packet received. So, for example, all Web servers, DNS servers, and routers are reflectors, since they will return SYN ACKs or RSTs in response to SYN or other TCP packets. The same is true for query replies in response to query requests, and ICMP Time Exceeded or Host Unreachable messages in response to particular IP packets.

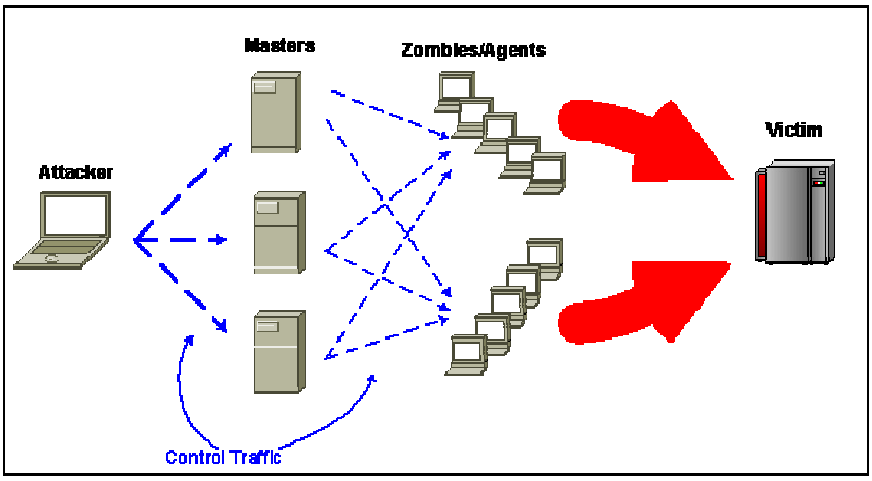

Figure 1: Description of a DDoS attack

DDoS attacks are quite common today, and they pose the main threat to public services because when a distributed attack is being generated against an Internet service, it is quite hard to block thousands of hosts sending flood data. This can be particularly painful if attacking packets are legitimate requests, since they cannot be easily associated to a DDoS attack.

Another aspect of most DDoS is that they consume a vast amount of resources from the network infrastructure, such as ISP networks and network equipment. This fact makes such attacks even more troublesome, because a single attack targeted against a minor web server, might bring the whole ISP's network down, and with it affect service for thousands of users.

3.2. Steps to Perform an Attack

Actually, the majorities of denial of service attacks are of distributed type, and have as basis flooding of large quantities of packets. Moreover, used packets might be changed to increase the harmfulness of the attack. Following, is a description of how to perpetrate a DDoS attack.

Today, in order to be able to generate a real DDoS overload attack it is not enough to have a few computers connected to Internet via a T1 leased line connection. Even a DS3 link might not be enough to bring a major web site down. In order to really be able to generate a massive amount of data in order to overload such a server the attacker would require a large number of hosts, each with a decent connection to the Internet. Summing up flood attacks from hundreds or even thousands of hosts can eventually generate a flood of hundreds of Mbps all directed against a single host or service.

Such a massive stream of data can overwhelm almost any system, including the network equipments, circuits and finally the servers themselves, even if a load balancer is being used. It is quite obvious that such attacks usually hit not only the victim of the attack, but also affect the close-by Internet area, because such amounts of traffic would overload many circuits and might crash the routers providing service to other clients as well.

Most of such attacks operate by building a large “army” of zombies which are spread all over the Internet. This army is composed of compromised Internet hosts which have a “zombie” or a Trojan[3] program installed on them. In order to create such a collection of hosts a way must be found to break into a large number of systems, and install a new program on these hosts.

After installing the software it has to be able to contact some central location in order to receive commands and maybe even software updates. This control channel would enable the attacker to initiate DDoS attacks on any Internet host whenever it is desired [6].

There have been several means of spreading such zombie software on the Internet. The simplest way is to break in into systems using known (or even better – unknown) security holes and manually installing the software. This method works just fine, but it’s slow and cumbersome. There are many other ways to distribute Trojan programs. For example a hacker can write a nice small free game, and put the Trojan’s installer inside. Now all that has to be done is to make sure that the link to the game is famous enough and wait for people to download it. Any person who would download the game and run it would install the zombie Trojan on his or her computer, adding it to the zombie army waiting for commands. The carrier program can take any shape and form, as long as it would attract enough people and make them run it on their computers. It has become quite common to find MP3 songs and other media files to carry viruses and Trojans inside.

Other ways of deployment have been seen on the Internet. Many worms[4] have been introduced into the Internet with a single objective in mind – spread as widely and as quickly as possible. For example, one of these worms was the Code Red worm [7]. Code Red was a malicious Internet worm which was propagating through the Internet, using vulnerabilities found in Microsoft Internet Information Service (IIS) servers.

After a Trojan has been installed on some host, it has to create a channel of communication with its “master” in order to receive commands and maybe even software updates. This functionality is not a must, but it would make a Trojan much more effective and the trouble of spreading it more worth while. A Trojan without such functionality can be used for a pre-defined set of tasks, but after it has finished, it would not serve any purpose.

A Trojan may use a large number of ways to communicate back to its master. The easiest way would be to open a TCP connection to a predefined host, and use this connection to receive instructions. This way of communication makes the Trojan quite vulnerable, because it would leave a very obvious trail behind it. Running the ‘netstat’ command on the infected computer would reveal the connection and with it the presence of the Trojan and the IP address of the master node. In order to hide the connection, the Trojan might use well-used protocols, such as HTTP or IRC.

---

[3] A destructive program that masquerades as a benign application. Unlike viruses, Trojan horses do not replicate themselves but they can be just as destructive. One of the most insidious types of Trojan horse is a program that claims to rid your computer of viruses but instead introduces malicious code onto your computer.

[4] A program or algorithm that replicates itself over a computer network and usually performs malicious actions, such as using up the computer's resources and possibly shutting the system down.

By using HTTP, the Trojan can open an HTTP connection to a pre-defined web server, and use a specialized CGI page to post information to the server, and request new commands and data. Such a connection might be lost between other normal HTTP connections. Also, the connection can be relatively short, and repeat once in a few minutes. This will ensure that the Trojan would be up to date, but stay hidden from the user.

Another common way to create the connection is use the IRC protocol. The Trojan would connect to a pre-defined IRC channel on some public IRC network. In such a way, any person with access to this IRC channel can send text messages to the Trojans, and program and activate attacks in minutes [6]. A famous Trojan who is known to operate in such a way is Sub7. In order to protect the master node, and make it harder for people to break into the system, it is quite common to encrypt the whole sessions using standard encryption software easily found on the Internet.

The type of attack generated by the Trojan can vary, and it depends on which tools the Trojan software is based on. Usually it would enable at least a few types of attacks, including TCP/UDP/ICMP floods. The master can control the type of the attack, the packet lengths, the destination IP and many other parameters. Virtually any kind of known flood attack can be mounted using such zombie Trojans.

Protecting against such attacks is quite difficult because the volume of the attack may be so huge, that it would saturate many circuits and block all available bandwidth at the victim's network area. This kind of attack can hardly be beaten by any technique which can be installed at the victim's facility. The amount of traffic would simply fill the connection to the ISP, and any devices placed behind it would have no way to deal with all the traffic.

3.2.1. Attack Identity

Usually, when someone wants to attack an Internet host, he would like to maintain anonymity in order to avoid prosecution or just to avoid being exposed. One of the means of keeping anonymity is using zombie hosts to do the dirty work. This way is quite secure, but it is possible to back track the original attacker via the zombie hosts or software [6]. Also the zombies themselves are exposed, and could be fixed quite quickly because the victim of the attack can report their real IP addresses to the owner or service provider.

Another way for the attacker to keep his anonymity is to use spoofed IP addresses, forging the source IP address of the attacks.

Using reflectors is another way to accomplish anonymity. The attacker would reflect the attack from other Internet hosts, and by that make it seem to the victim as if they are the attackers. Reflectors can be used by many attack types. A single attacker host as well as many zombie hosts can use reflectors in order to hide the attack sources (see figure 2).

The main problem with reflectors is that they can be very secure Internet hosts, and still be used as reflectors. Almost any host which offers services to the Internet can be used as a reflector because it follows the IP standard. The basic idea is exploiting standard protocols

which have a request-response sequences build into them. The request would be sent from the attacker to the reflector, with the source IP set to the victim's address. The reflector would send the response to the victim, effectively reflecting the attack.

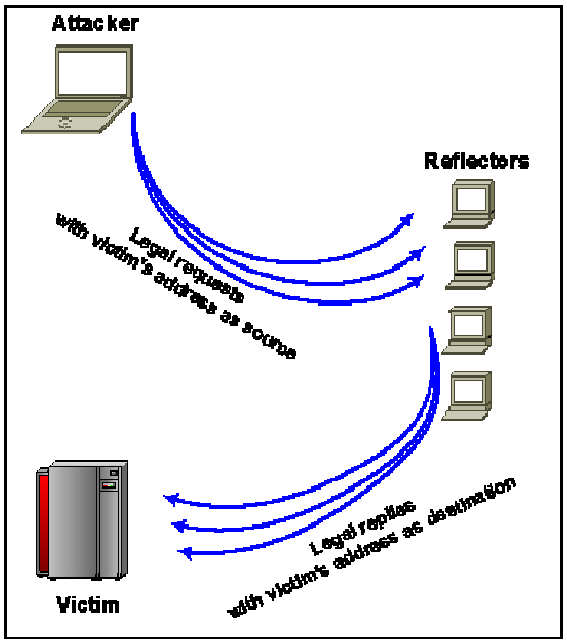

Figure 2: Reflector attack

3.3. Examples of Attacks

The attacks which are described in this document are only some of all those available on the Internet. Their common denominator is that they all use weaknesses or erroneous implementations of the TCP/IP protocol, or they utilize weaknesses in the specification of the TCP/IP protocol itself. Besides these particular exploiting attacks there is the always effective brute force attack that works well with the exploitation of weaknesses in the TCP/IP suite implementations and specifications.

The reminder of this section presents TCP/IP attacks, their impact on the network and whenever possible some solutions to protect the network from their occurrence.

3.3.1. SYN Flood Attack

When a system (called the client) attempts to establish a TCP connection to a system providing a service (the server), the client and server exchange a sequence of messages. This connection technique applies to all TCP connections – telnet, Web, email, etc.

The client system begins by sending a SYN message to the server, asking the server to open a connection. The server then acknowledges the SYN message by sending a SYN-ACK message to the client, meaning it accepts to open the connection from the client (the ACK part) and asking if the client agrees to open the connection in the opposite sense (the SYN

part). The client then finishes establishing the connection by responding with an ACK message to server. The connection between the client and the server is then open, and the service-specific data can be exchanged between the client and the server. Figure 3 presents a view of this message flow.

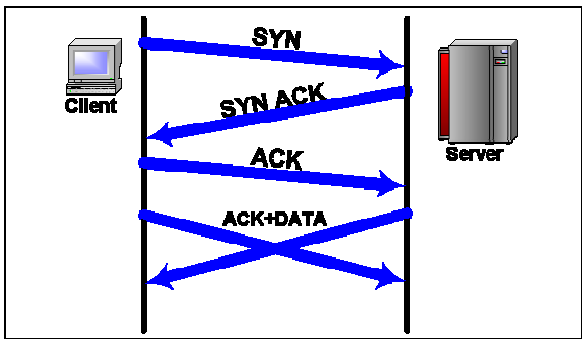

Figure 3: TCP three-way handshake

The potential for abuse arises at the point where the server system has sent an acknowledgment (SYN-ACK) back to client but has not yet received the ACK message. This is what we mean by half-open connection. The server has built in its system memory a data structure describing all pending connections. This data structure is of finite size, and it can be overflowed by intentionally creating too many partially-open connections.

Creating half-open connections is easily accomplished with IP spoofing. The attacking system sends SYN messages to the victim server system; these appear to be legitimate but in fact reference a client system that is unable to respond to the SYN-ACK messages. This means that the final ACK message will never be sent to the victim server system.

The half-open connections data structure on the victim server system will eventually exhaust; then the system will be unable to accept any new incoming connections until the table is emptied out. Normally there is a timeout associated with a pending connection, so the half-open connections will eventually expire and the victim server system will recover. However, the attacking system can simply continue sending IP-spoofed packets requesting new connections faster than the victim system can timeout the pending connections.

In most cases, the victim of such an attack will have difficulty in accepting any new incoming network connection. In these cases, the attack does not affect existing incoming connections nor the ability to originate outgoing network connections.

However, in some cases, the system may exhaust memory, crash, or be rendered otherwise inoperative.

The location of the attacking system is obscured because the source addresses in the SYN packets are often implausible. When the packet arrives at the victim server system, there is no way to determine its true source. Since the network forwards packets based on destination address, the only way to validate the source of a packet is to use input source filtering.

Any system connected to the Internet and providing TCP-based network services (such as a Web server, FTP server, or mail server) is potentially subject to this attack. Note that in addition to attacks launched at specific hosts, these attacks could also be launched against routers or other network server systems if these hosts enable (or turn on) other TCP services (e.g., echo). The consequences of the attack may vary depending on the system; however, the attack itself is fundamental to the TCP protocol used by all systems.

You should note that this type of attack does not depend on the attacker being able to consume your network bandwidth. In this case, the intruder is consuming kernel data structures involved in establishing a network connection. The implication is that an intruder can execute this attack from a dial-up connection against a machine on a very fast network.

A relatively small flood of bogus packets will tie up memory, CPU, and applications, resulting in shutting down a server.

Although some have described TCP SYN Flood as a bug in TCP/IP, it is more correctly a feature of the design. TCP/IP was designed for a friendly Internet, and a limited connection queue (collection of resources reserved per connection) has worked fine for years.

Early fixes have focused on increasing the length of the queues and reducing a timeout value. The timeout value controls how long an entry waits in the queue until an acknowledgement is received. The problem with simply making the queue longer is that there are actually many queues (one for each TCP server on the system – HTTP, FTP, SMTP, etc.), and lengthening the queues to very large values, for example, eight kilobytes, results in an operating system requiring enormous amounts of memory (over 100 megabytes for a system with 25 server applications).

Shortening the timeouts can also help when used with longer queue lengths because the spoofed packets get removed from the queues more quickly. However, shortening the timeouts also affects new outgoing connections, and remote users with slow links which may never get connected to the server.

Actually, several other countermeasures are available. Some of them include:

- To check periodically incomplete connection requests, and randomly clear connections that have not completed a three-way handshake. This will reduce the likelihood of a complete block due to a successful SYN attack, and allow legitimate client connections to proceed.
- To limit TCP SYN traffic rate.
- To install an IDS (Intrusion Detection System) capable of detecting TCP SYN flood attacks.
- To use circuit level firewalls (stateful inspection) to monitor the handshake of each new connection and maintain the state of established TCP connections. The filtering

system must be able to distinguish harmful uses of a network service from legitimate uses.

- To set a firewall to block all incoming packets with bad external IP addresses like 10.0.0.0 to 10.255.255.255, 127.0.0.0 to 127.255.255.255, 172.16.0.0 to 172.31.255.255, and 192.168.0.0 to 192.168.255.255 and all internal addresses.
- To modify the TCP implementation to reduce the amount of information stored for each in-progress connection.
- To verify the return route of each new connection. If it is different than the received packets, which is normal during this attack, connection should be dropped.

However, as some attacks have proved, if enough SYN packets are thrown at a site, any site can still be SYNed off the net.

3.3.2. TCP Flooding

Probably, whenever we have heard about TCP flooding attacks, we were talking about a TCP SYN flood attack. However, it is possible to experience a TCP flooding attack that is not taking advantage of the TCP three-way handshake. It is also possible to perpetrate a TCP flooding attack taking advantage of other TCP's finite state or TCP's flags.

TCP ACK flood

In this attack, a lot of TCP ACK packets are sent to victim to utilize its system and network resources. Depending on the OS, an open port or closed port might reply a TCP RESET packet, causing more traffics and workload on the victim and victim's network.

An evolution of this attack consists in flooding the victim with TCP ACK packets with spoofed source IP, random sequence number and random port number in the packet.

NULL flood

This TCP flooding attack is accomplished with TCP packet's TCP flag all set to 0. This is where the 'NULL' means. The victim might ignore it, consume system resource or crash completely depending on the operating system implementation.

RST Attack

There is also a quite similar DoS attack called RST. The TCP Reset flag is used to abort TCP connections, usually to signify an irrecoverable error. When receiving such a packet, the host deletes the connection and frees data structures. To prevent a bad utilization of this flag, only RST messages that fit in the sequence number window are accepted.

By sending RST packets with correct sequence numbers (packets can be sniffed from the network) and with a spoofed IP address, an active TCP connection can be torn down quite effectively. The purpose and effect of this kind of attack is similar to SYN flood DoS attack, however it is not necessary to send large volumes of information to accomplish an effective cutting of services at the attacked machine.

3.3.3. UDP Flooding

The UDP flooding attack belongs to the class of brute force attacks, and it is perpetrated by programs that launch denial-of-service attacks by creating a "UDP packet storm" either on a system or between two systems.

This attack is possible when an attacker sends an UDP packet to a random port on the victim system. When the victim system receives an UDP packet, it will determine what application is waiting on the destination port. When it realizes that there is no application that is waiting on the port, it will generate an ICMP packet of destination unreachable to the forged source address. If enough UDP packets are delivered to ports on victim, the system will go down.

An attack on one host causes that host to perform poorly. An attack between two hosts can cause extreme network congestion in addition to adversely affecting host performance.

Anyone with network connectivity can cause a denial of service. This attack does not enable them to gain additional access.

To dam up UDP floods it is just necessary to block all UDP services request that will not be used. Programs that need UDP will still work, unless of course, the sheer volume of the attack mauls the Internet connection.

3.3.4. ICMP Flood

Like the other flooding attacks, this one is accomplished by broadcasting a bunch of ICMP packets, usually ping packets. The idea is to send so much data to the system that it slows down so much and gets disconnected due to timeouts.

Particularly, Ping flood attacks attempt to saturate a network by sending a continuous series of ICMP echo requests over a high-bandwidth connection to a target host on a lower-bandwidth connection. The receiver must send back an ICMP echo reply for each request.

3.3.5. Smurf

Smurf attacks send ICMP echo requests (pings) to the broadcast addresses of well-populated "intermediate" networks. The source IP addresses of attack packets are spoofed to match the address of an attacked host on a target network. After receiving a copy of the echo request, each host in the intermediate network responds with an echo reply to the attacked host, flooding both the host and its network.

These attacks can result in large amounts of ICMP echo reply packets being sent from an intermediary site to a victim, which rapidly exhausts the bandwidth available to the target, effectively denying its services to legitimate users and causing network congestion or outages. These attacks have been referred to as "smurf" attacks because the name of one of the exploit programs attackers use to execute this attack is called "smurf."

The two main components to the smurf denial-of-service attack are the use of forged ICMP echo request packets and the direction of packets to IP broadcast addresses.

In the "smurf" attack, attackers are using ICMP echo request packets directed to IP broadcast addresses from remote locations to generate denial-of-service attacks. There are three parties in these attacks: the attacker, the intermediary, and the victim (note that the intermediary can also be a victim).

The intermediary receives an ICMP echo request packet directed to the IP broadcast address of their network, which will be an amplifier. If the intermediary does not filter ICMP traffic directed to IP broadcast addresses, many of the machines on the network will receive this ICMP echo request packet and send an ICMP echo reply packet back. When (potentially) all the machines on a network respond to this ICMP echo request, the result can be severe network congestion or outages. Figure 4 represents how smurf is executed.

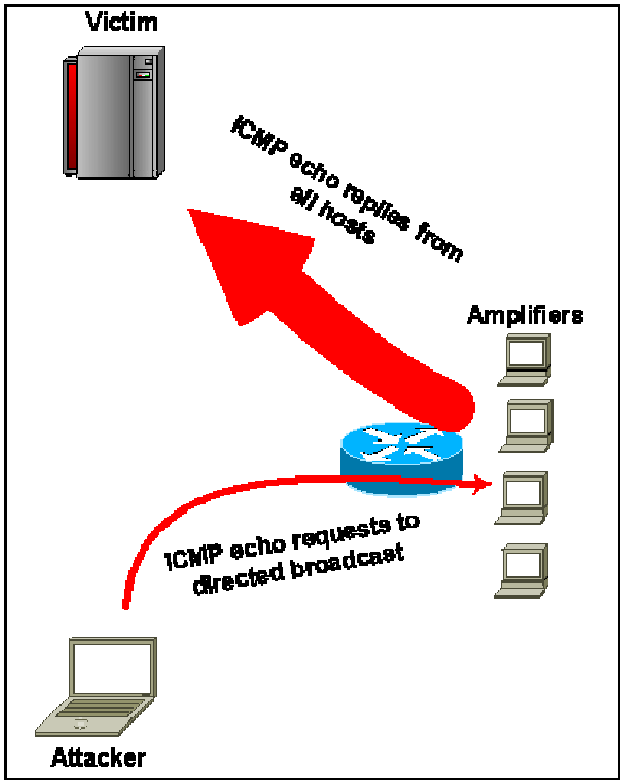

Figure 4: Representation of Smurf attack

When the attackers create these packets, they do not use the IP address of their own machine as the source address. Instead, they create forged packets that contain the spoofed source address of the attacker's intended victim. The result is that when all the machines at the intermediary's site respond to the ICMP echo requests, they send replies to the victim's machine. The victim is subjected to network congestion that could potentially make the network unusable. Even though the intermediary was not labeled as a "victim," the intermediary can be victimized by suffering the same types of problem that the "real victim" does in these attacks.

Attackers have developed automated tools that enable them to send these attacks to multiple intermediaries at the same time, causing all of the intermediaries to direct their responses to the same victim. Attackers have also developed tools to look for network routers that do not filter broadcast traffic and networks where multiple hosts respond. These networks can then subsequently be used as intermediaries in attacks.

Both the intermediary and victim of this attack may suffer degraded network performance, both on their internal networks or on their connection to the Internet. Performance may be degraded to the point that the network cannot be used, since all the available bandwidth is being consumed.

A significant enough stream of traffic can cause serious performance degradation for small and mid-level ISPs that supply service to the intermediaries or victims. Larger ISPs may see backbone degradation and peering saturation.

Fortunately, this type of attack can be blocked by just setting the router to ignore broadcast addressing and setting the firewall to ignore ICMP requests.

3.3.5. Ping of Death

The TCP/IP specification (the basis for many protocols used on the Internet) allows for a maximum packet size of up to 65536 octets (1 byte = 8 bits of data), containing a minimum of 20 bytes of IP header information and 0 or more bytes of optional information, with the rest of the packet being data. It is known that some systems will react in an unpredictable fashion when receiving oversized IP packets. Reports indicate a range of reactions including crashing, freezing, and rebooting.

In particular, some reports indicate that Internet Control Message Protocol (ICMP) packets issued via the "ping" command have been used to trigger this behavior. The "ping" command can be used to construct oversized ICMP datagrams (which are encapsulated within an IP packet), taking advantage that many ping implementations by default send ICMP datagrams consisting only of the 8 bytes of ICMP header information, but allow the user to specify a larger packet size if desired.

An attacker sends an ICMP ECHO request packet that is much larger than the maximum IP packet size to victim. Since the received ICMP echo request packet is bigger than the normal IP packet size, the victim cannot reassemble the packets. The OS may be crashed or rebooted as a result.

The Ping of Death is a typical TCP/IP implementation attack. In this assault, the DDoS attacker creates an IP packet that exceeds the IP standard's maximum 65,536-byte size. When this large packet arrives, it crashes systems that are using a vulnerable TCP/IP stack. No modern operating system or stack is vulnerable to the simple Ping of Death, but it was a long-standing problem with Unix systems.

3.3.6. Teardrop

The Teardrop, though, is an old attack that relies on poor TCP/IP implementation that is still around. It works by interfering with how stacks reassemble IP packet fragments. The trick here is that as IP packet are sometimes broken up into smaller chunks, each fragment still has the original IP packet's header, and field that tells the TCP/IP stack what bytes it contains. When it works right, this information is used to put the packet back together again. What happens with Teardrop though is that your stack is buried with IP fragments that have overlapping fields. When the stack tries to reassemble them, it cannot do it, and if it does not know to toss these trash packet fragments out, it can quickly fail. Most systems know how to deal with Teardrops now and a firewall can block Teardrop packets in return for a bit more latency on network connections since this makes it disregard all broken packets. Of course, if you throw a ton of Teardrop busted packets at a system, it can still crash

Many other variants such as Targa, SynDrop, Boink, Nestea Bonk, TearDrop2 and NewTear are available to accomplish this kind of attack.

3.3.7. Land

A LAND attack consists of a stream of TCP SYN packets that have the source IP address and TCP port number set to the same value as the destination address and port number (i.e., that of the attacked host). Some implementations of TCP/IP cannot handle this theoretically impossible condition, causing the operating system to go into a loop as it tries to resolve repeated connections to itself.

Service providers can block LAND attacks that originate behind aggregation points by installing filters on the ingress ports of their edge routers to check the source IP addresses of all incoming packets. If the address is within the range of advertised prefixes, the packet is forwarded; otherwise it is dropped.

3.3.8. Echo/Chargen

The character generator (chargen) service is designed to simply generate a stream of characters. It is primarily used for testing purposes. Remote users/intruders can abuse this service by exhausting system resources. Spoofed network sessions that appear to come from that local system's echo service can be pointed at the chargen service to form a "loop." This session will cause huge amounts of data to be passed in an endless loop that causes heavy load to the system. When this spoofed session is pointed at a remote system's echo service, this denial of service attack will cause heavy network traffic/overhead that considerably slows down the network. It should be noted that an attacker does not need to be on your subnet to perform this attack as he/she can forge the source addresses to these services with relative ease.

3.3.9. Naptha Attack

The number and type of resources that an attacker can target for a denial-of-service attack are many and varied. The Naptha work highlights a set of them for which some specific defenses exist.

In general, any system that allows critical resources to be consumed without bound is subject to denial-of-service attacks. Naptha and similar network attacks are more dangerous for several reasons:

- They can be done "asymmetrically" – that is, the attacker can consume vast amounts of a victim's limited resource without a commensurate resource expenditure.
- In combination with other vulnerabilities or weaknesses, they can be done anonymously.
- They can be included in distributed denial-of-service tools.

Naptha attacks mainly exploit weaknesses in the way some TCP stacks and applications handle large numbers of connections, creating a resource starvation attack. Particularly with Naptha attacks, instead of actually creating connections to the victim, the Naptha tool fools the victim's operating system into thinking it sees valid network connections. The victim's server application waits for a valid request, or receives a request and attempts to send the data, which instead languishes in the operating system because it cannot be sent. Because the connections are simulated instead of real, an attacker can launch a devastating attack with little in the way of resources.

If an unusual number of connections in a particular state are noticed, it may be an indication of this type of attack. The definition of "unusual" in this case depends largely on the types of services offered by the attacked machine. For example, a large number of connections in the ESTABLISHED state on a web server may simply be an indication of a busy web server. Understanding the normal usage patterns of services offered may help to distinguish an attack from ordinary activity. Many operating systems offer a netstat utility that is useful for examining the state of connections.

3.3. Some Solutions to DoS Attacks

The way DoS and DDoS attacks are perpetrated, by exploiting limitations of protocols and applications, is one of the main factors why they are continuously evolving, and because of that presenting new challenges on how to combat or limit their effects.

Even if all of these attacks cannot be completely avoided, some basic rules can be followed, to protect the network against some, and to limit the extent of the attack [3][4] :

- Make sure the network has a firewall up that aggressively keeps everything out except legal traffic.
- Implement router filters. This will lessen the exposure to certain denial-of-service attacks. Additionally, it will aid in preventing users on network from effectively launching certain denial-of-service attacks.
- Install patches to guard against TCP/IP attacks. This will substantially reduce the exposure to these attacks but may not eliminate the risk entirely.

- Disable any unused or unneeded network services. This can limit the ability of an intruder to take advantage of those services to execute a denial-of-service attack.
- Observe the system performance and establish baselines for ordinary activity. Use the baseline to gauge unusual levels of disk activity, CPU usage, or network traffic.
- Keep the anti-viral software up to date. This will prevent the site becoming a home for DDoS agents like TFN.
- Invest in redundant and fault-tolerant network configurations.

Besides the rules listed above, it is important for a network administrator, or even a machine administrator, to keep current on the latest DDoS developments.

Also, since there is no silver bullet for DDoS attacks several companies offer program and services that can help a network administrator to manage DDoS assaults. Essentially, these corporate approaches consist of intense real-time monitoring of the network looking for telltale signs of incoming DDoS attacks.

Conclusion

Denial of Service and Distributed Denial of Service attacks, which exploit inherent weaknesses in the design and organization of the Internet, are rapidly becoming the weapon of choice for hackers around the globe. Easily launched using readily available tools (see annex 2 with some of the major tools) against individual Web sites, chat servers, email servers or network components such as aggregation routers, core routers or DNS servers, DDoS attacks and similar assaults are designed to paralyze businesses by flooding sites with bogus traffic, preventing legitimate transactions from completing.

The growing dependence on the Internet makes the impact of these attacks increasingly painful for service providers, enterprises, hosting centers and government agencies alike. DDoS attacks are already among the most difficult to defend against, and newer, more powerful tools promise to unleash even more destructive attacks in the months and years to come.

Responding to and defeating these attacks in a timely and effective manner is the primary challenge confronting Internet–dependent organizations today. Traditional perimeter security technologies such as firewalls and intrusion detection systems do not provide adequate DDoS protection, while filtering solutions such as router–based access control lists (ACLs) simply cannot separate good traffic from bad for most attacks, resulting in legitimate transactions being blocked.

To protect them, businesses require a next–generation architecture, purpose–built specifically to detect and defeat increasingly sophisticated, complex and deceptive attacks without impacting ongoing business operations.

Bibliography

[1] “RFC 791 – Internet Protocol: Protocol Specification”, Defense Advanced Research Projects Agency, September 1981

[2] “RFC 793 – Transmission Control Protocol: Protocol Specification”, Defense Advanced Research Projects Agency, September 1981

[3] “RFC 768 – User Datagram Protocol”, J. Postel, ISI, August 1980

[4] “RFC 792 – Internet Control Message Protocol”, J. Postel, ISI, September 1981

[5] “The Strange Tale of the Denial of Service Attacks Against GRC.COM”, S. Gibson http://www.grc.com

[6] “Results of the Distributed-Systems Intruder Tools Workshop” http://www.cert.org/reports/dsit_workshop.pdf

[7] “New Order Newsletter 002 – Full Analysis of the "Code Red" Worm” http://neworder.box.sk/newsletter2.php

Annex I – DoS Attacks

DoS attacks “industry” is continuously evolving, and most probably at the time of reading this document, new kinds of attacks will be ready to use, which makes the task of presenting all of them in the same document difficult. In that sense, section 3.2 only presents the more usual, or those who are being important.

To complete this document, table 1 presents a brief description of the most popular DoS attacks.

Name of Attack	Protocol	Description
Land	TCP SYN	Source and destination IP addresses are the same causing the response to loop.
SYN Flooding	TCP	Sending large numbers of TCP connection initiation requests to the target. The target system must consume resources to keep track of these partially open connections.
Teardrop	TCP fragments	Sends overlapping IP fragments.
Smurf	ICMP	ICMP ping requests to a directed broadcast address. The forged source address of the request is the target of the attack. The recipients of the directed broadcast ping request respond to the request and flood the target's network.
Ping of Death	ICMP	ICMP packets greater than 65536 bytes can shut down a system.
Open/Close	TCP/UDP	The open/close attack opens and closes connections at a high rate to any port serviced by an external service through inetd. The number of connections allowed is hard coded inside inetd.
ICMP Unreachable	ICMP	The attacker sends ICMP unreachable packets from a spoofed address to a host. This causes all legitimate TCP connections on the host to be torn down to the spoofed address. This causes the TCP session to retry and as more “ICMP unreachable” messages are sent, a DoS condition occurs.
ICMP Redirect	ICMP	ICMP redirects can cause data overload to the system being targeted.

IRDP	ICMP	ICMP Router Discovery Protocol can be spoofed and cause fake routing entries to be entered into a Windows machine. IRDP has no authentication. Upon startup, a system running MS Windows95/98 will always send 3 ICMP Router Solicitation packets to the 224.0.0.2 multicast address. If the machine is not configured as a DHCP client, it ignores any Router Advertisements sent back to the host. However, if the Windows machine is configured as a DHCP client, any Router Advertisements sent to the machine will be accepted and processed.
ARP Redirect	ARP	Only local subnet can be attacked. ARP tables do not have the correct layer 2/layer 3 address pair, and a system can start sending information to an incorrect destination.
Looping UDP Ports	UDP	The attack uses 2 UDP services. Chargen (port 19) and echo (port 7), can be spoofed into sending data to each other.
Fraggle	UDP	Same as Smurf, but rather than ICMP uses UDP to broadcast address for amplification.
UDP Flood	UDP	Sending large numbers of UDP (User Datagram Protocol) packets to the target system, thus tying up network resources.
TCP Flood	TCP NUL, TCP RST, TCP ACK	When TCPs communicate, each TCP allocates some resources to each connection. By repeatedly establishing a TCP connection and then abandoning it, a malicious host can tie up significant resources on a server.
UDP Reflectors	UDP	All Web servers, DNS servers, and routers are reflectors, since they will return SYN acks or RSTs in response to SYN or other TCP packets; query replies in response to query requests; or ICMP Time Exceeded or Host Unreachable in response to particular IP packets. By spoofing IP addresses from slaves — a massive DDoS attack can be arranged.
URL Attacks	TCP	URL attacks attempt to overload an http server via various methods: http bombing — continuous requests for the same homepage or large web page; requesting the page with REFRESH so as to bypass any proxy server. Many of these attacks are not zombie attacks but rather human executed — by hundreds simultaneously.
VPN attacks	TCP	Using specially crafted GRE or IPIP packets to attack the destination address of a VPN.

Table 1: Description of some DoS Attacks

Annex II – DoS Attack Tools

DoS attack tools are continuously evolving, from simple applications used to attack directly one site/machine, leaving some clues behind, like the IP address of the attacking machine, to more sophisticated applications, that use zombies to perform the attacks, and other techniques to make difficult its detection.

Table 2 presents a summary of the most common attacking tools, with the main protocols that are used to accomplish the DoS attack. These tools include some worms that became famous, since they were used to attack some popular sites, not long ago.

Name of Tool	Protocol	Description
Trinoo	UDP	Flooding of UDP packets. Only initiates UDP attacks to random ports. Communication between master and slave is via unencrypted TCP and UDP. No IP spoofing. Uses UDP ports 27444 and 31335.
TFN (Tribal Flood Network)	UDP, ICMP Echo, TCP SYN, Smurf	Flood of different types of packets. Uses IP spoofing. Uses ICMP Echo reply packets to communicate between zombie and master.
Stacheldracht v4	UDP, ICMP, TCP SYN, Smurf	Flood of packets. Uses encryption for communications (but not for ICMP heartbeat packets that zombie sends to master) and has an auto-update feature (via rcp). Has ability to test (via ICMP Echo) if it can use spoofed IP addresses.
Stacheldracht v2.666	UDP, ICMP, TCP SYN, Smurf, TCP ACK, TCP NUL	Flooding of different kinds of packets. Uses encryption for communications (but not for ICMP heartbeat packets that zombie sends to master) and has an auto-update feature (via rcp). Has ability to test (via ICMP Echo) if it can use spoofed IP addresses.
TFN 2K	UDP, ICMP Echo, TCP SYN, Smurf	Same as TFN – but the slave is silent so difficult to spot. No return of info from the slave. Zombie to master communication is encrypted.
FAPI	UDP, TCP SYN, TCP ACK, ICMP	Flooding of UDP, TCP (SYN and ACK) or ICMP packets. With smurf extensions. Can spoof IP addresses. Not designed for large networks.
Carko (Stacheldraht v1.666 + antigl + yps)	UDP, ICMP, TCP SYN, Smurf, TCP ACK, TCP NUL	Mix of 3 other attack tools with some minor modifications, which flood an end system with packets. Uses the backdoor hole of snmpXdmid (exploitation of buffer overflow vulnerability) and uses UDP port 530. Used in February 2000 to overwhelm Yahoo!, eBay and Amazon servers.
Freak88	ICMP	NT specific zombie. No spoofing capabilities. Sends ICMP 1500 octet packets marked as fragments.
Shaft	UDP, ICMP, TCP SYN	Flooding of UDP, TCP SYN or ICMP Echo packets. Can randomize all three attacks. Uses UDP ports 18753 and 20433. Has optional IP spoofing capabilities (needs root). Can set ICMP/UDP packet size.

Mstream	TCP ACK	Flooding of TCP ACK packets. Usually uses TCP port 12754 but can use any port. Master connects via telnet to zombie. Communication between zombie and controller is not encrypted. The target gets hit by ACK packets and sends TCP RST to non-existent IP addresses. Routers will return ICMP unreachable causing more bandwidth starvation. Can randomize all 32 bits of source address.
Blitznet	TCP SYN	Launches a distributed SYN flood attack with spoofed source IP, without logging.
Ramen	Multicast	Ramen is a worm that propagates by using a newly compromised system to scan Class B (/16) wide address spaces, searching for port 21 (FTP) and looking for new vulnerable hosts. SYN scanning performed by Ramen can disrupt network traffic when scanning the multicast network range.
Targa	Any	Works by sending malformed IP packets known to slow down or hangup many TCP/IP network stacks.
Spank	Multicast	Will only work on a multicast enabled network. Similar to Mstream.
Stick	Any	Stick uses the straightforward technique of firing numerous attacks at random, from random source IP addresses to purposely trigger IDS events. Stick is a DoS tool against IDS systems.
Omega	TCP ACK, UDP, ICMP, IGMP	Flooding of packets. Can spoof IPs and has a chat option between attackers.
NAPHTA	TCP	Naptha attacks exploit weaknesses in the way some TCP stacks and applications handle large numbers of connections in states other than "SYN RECVD," including "ESTABLISHED" and "FIN WAIT-1."
Trinity (also called My Server and Plague)	UDP, TCP Fragment, TCP SYN, TCP RST, TCP Random Flag, TCP ACK, Establish, NULL	Attack tool that preys on Linux servers and uses IRC channels to unleash IP packet floods on targeted host machines. Listens to TCP port 33270. When idle it connects to Internet IRC server on port 6667.
IRC bots	ICMP, UDP	Flooding attack, perpetrated by zombies, with ICMP and UDP packets. Zombie systems controlled via a central IRC channel. Sub7Server trojan used to maintain control over the zombie.
HTTP Smurf	TCP HTTP	Using public IIS servers as unsuspecting zombies, it sends a string of data to multiple web servers and they reflect the data to the target.

Code Red	TCP HTTP	Worm used in 2001 to attempt a DDoS attack against www.whitehouse.gov. This worm exploits a known vulnerability in IIS servers to infect other servers. After gaining administrator-level access to the first server, the Code Red worm defaces the Web server's pages and begins spreading to other servers. Code Red made use of a known vulnerability in Microsoft's IIS Web server: an unchecked buffer in the Indexing Server ISAPI that could be overrun and thereby give administrator-level access to the server. Actually IIS servers are patched against Code Red worm.
Power worm	TCP HTTP	A worm, known by the name of "Power" that compromises systems vulnerable to the IIS Unicode vulnerability. It uses the IRC as a control channel for coordinating compromised machines in DDoS attacks. Based on reports, over 10,000 machines have already been compromised by this worm.
Cisco	ICMP	Use a Cisco router as a zombie for an ICMP based ping attack. By using the directed broadcast address, the attacker's ICMP packets will be routed across the Internet to the victim's network. Once in the target network, most IP stacks will respond to a broadcast address, by replying. To avoid this attack, routers only have to be correctly configured.
Nimda	SMTP, MIME, HTTP, TFTP, TCP/IP	Nimda can infect PCs and servers in any of four ways: through an e-mail attachment, by scanning for vulnerable servers running Microsoft's Internet Information Server software and then exploiting a flaw in the software, through shared hard drives, and by fooling browsers into uploading the worm from infected Web servers. This worm slows network performance.

Table 2: DoS attack tools