A Survey of DDoS attacks and some DDoS defense mechanisms

Advanced Information Assurance (CS 626)

Submitted by
Puneet Zaroo

1 Introduction

A Denial of Service attack is an attempt by a person or a group of persons to cripple an online service. This can have serious consequences, especially for companies like Amazon and eBay which rely on their online availability to do business. In the not so distant past there have been some large scale attacks targeting high profile internet sites[3,4,5,6]. Consequently, there are currently a lot of efforts being made to come up with mechanisms to detect and mitigate such attacks.

Even though the first denial of service attacks did not take place a long time ago (tools that automate setting up of an attack network and launching of attacks, started appearing in 1998), there are a multitude of denial of service attacks that have been used. Broadly speaking the attacks can be of three forms. a) Attacks exploiting some vulnerability or implementation bug in the software implementation of a service to bring that down . b) Attacks that use up all the available resources at the target machine. c) Attacks that consume all the bandwidth available to the victim machine.

The third type of attacks are called **bandwidth attacks**. A distributed framework becomes especially suited for such attacks as a reasonable amount of data directed from a number of hosts can generate a lot of traffic at and near the target machine, clogging all the routes to the victim. Protection against such large scale distributed bandwidth attacks is one of the most difficult (and urgent) problem to address in today's internet. CERT reports bandwidth attacks as increasingly being the most common form of Denial of Service attacks seen in the internet today.

The rest of the paper is organized as follows. In section 1, I provide a brief history of denial of service attacks. The original internet design envisaged a group of co-operating interconnected hosts. The reality of today's internet differs from the design assumptions made by the inventors of the internet. Section 2 discusses these and other problems with the internet design that make denial of service attacks possible. Section 3 desdcribes a brief history of DDoS attacks along with a few details about the various techniques used. Section 4 discusses some recent and some not so recent proposals made to defend against bandwidth denial of service attacks. Finally, I conclude in section 5 by insight into what the future might hold with respect to DoS attacks.

2 What makes DDoS attacks possible?

Internet was designed with functionality and not security in mind. The TCP/IP protocol suite, the most widely used protocol suite for data communication assumes that all the hosts participating in the communication have no malicious intent. There is no security built into the internet infrastructure to protect hosts from other hosts not regulating their own behavior. For example, the TCP protocol assumes that hosts will reduce the rate of packet transmission on detecting packet losses due to congestion. If a particular host instead does not respond to the congestion conditions, it can easily overwhelm the intermediate links to the destination.

Such design opens up the internet to many opportunities for denial of service attacks[1]. Some features of the internet that make DoS attacks possible are :

- **Internet Security is highly dependent** :DDoS attacks are launched from hosts whose security has been subverted. No matter how secure a particular host is, it

opens itself to the possibility of a DDoS attacks if there are other insecure hosts in the internet which can be used to launch such attacks.

- **Difficulty in tracing back the attack to the source** : Most (if not all) of the internet runs on top of the TCP/IP protocol. The underlying protocol (IP) is basically connectionless in nature. At each intermediate step from the source to the destination, the decision about the next host to forward the packet is made. All such routing decisions are made on the basis of the destination address. It is thus possible to generate packets with incorrect source IP addresses and use them to launch Denial of Service attacks. This technique is known as IP spoofing. Users with sufficient privileges on a system have the ability to fabricate such fake packets. E.g in Linux , raw sockets can be created which enable users (with super user access)to construct all the packet contents and headers for a given packet. This makes the task of determining the true source of attack very difficult. Apart from the source IP address, the attackers nowadays even randomly change all the headers in an IP datagram, keeping just the destination address constant. This makes dropping of packets based on certain characteristics very difficult, as distinguishing attack packets from legitimate packets becomes difficult. There are also some attacks that rely on illegitimate source addresses to launch a denial of service attack on the hosts whose IP address was used. The Smurf attack is one such example. If on detection of an attack, packets are dropped solely on the basis of the IP source addresses, then the hosts whose IP addresses were used for the spoofing will suffer from a denial of service.
- **Limited Resources** : The infrastructure of the interconnected hosts and networks is comprised of limited resources. Bandwidth processing power and storage capacities are all targets of Denial of Service attacks. If these resources are increased by substantial investments, it just raises the bar on the degree an attack must reach to be effective. Even if the attack is not able to shut down the victim completely, it may waste its resources, reducing the level of quality as seen by the end users, and making the service provider incur heavy financial losses.
- **A target rich environment**: If the people in the military were to describe the internet today, they would describe it as a target rich environment. There are thousands and thousands of hosts and networks in the internet with vulnerabilities that can be exploited to get access to the machines there. It is therefore easy to gain control of a large number of hosts that can be then used as a spring board to launch DDoS attacks.
- **Easier to break systems than to make them** : Just as it is easier to destroy a car , than to make a good car, (all you need is a good sledgehammer), it is easier to break the networking infrastructure / protocols than to develop them. All the hosts in the internet, including the intermediate routers expect certain packet formats and traffic behavior. Since, at the time of the design of these software, no one foresaw the system being used for malicious purposes. This can lead to unexpected behavior of the network systems as response to unexpected packets. E.g all routers and hosts allocate buffers in memory while waiting for all the fragments constituting a datagram to arrive. If such a router is sent badly formed fragments indicating a very large datagram, the router will keep on allocating huge amounts of memory till it runs out of memory and cannot process more

datagrams. Similar results can be achieved with fragments that indicate negative offsets within its datagram.

3 History and Trends in DDoS attacks.

The DDoS attacks gained very widespread notoriety and media exposure with the three days of DoS attacks[3,4,5,6] (Feb 7-11, 2000) that were launched against major internet sites like CNN, Yahoo, EBay and Datek. Multiple attack tools like Trinoo, TFN, StachleDraht, TFN2K were used in these attacks. Ironically, these attacks came just a day after Steve Bellovin's talk on Distributed Denial of Service at NANOG (North American Network Operator's Group) in San Jose. But, Denial of Service attacks had been observed, studied and some attack tools like Trinoo and TFN even analyzed much before the infamous week.

The sophistication of the DDoS attack tools has kept on improving with time. Therefore a historical study of DDoS attacks also gives a good overview of the various techniques that are used in orchestrating such attacks. This section starts off with an explanation of the various steps involved in orchestrating a DDoS attack. We then move on to look at the various attack tools available and study how these tools evolved over time.

3.1 Steps in orchestrating a DDoS attack

A DDoS attack is carried out by a group of machines (refered to as "zombies" or "agents") that start sending packets to a victim host on receiving commands from a machine (refered to as a "handler" or a "master") under the control of the attacker. Therefore, an attacker has to take the following steps in order to launch a DDoS attack[15].

a) Compromise a number of hosts and deploy software on them that converts them into agents or handlers in the attack network: An attacker first scans the network looking for vulnerable hosts. The next step is to try an exploit a known vulnerability for that host to compromise the system. After compromising a system, software that converts the system into an agent or a handler is deployed on that and the attacker then leaves after trying to cover his tracks. Some tools also carry out one more step called the propagation step, in which all the compromised hosts recursively start the whole process of scanning, exploitation and deployment.
   In the earlier days of DDoS technology development, all the 4 steps (scanning, compromise, deployment and propagation) would be done manually by the attacker. Over time, more and more automation has been brought into each of these steps. The T0rnkit[17] toolkit, essentially a rootkit with some DoS attack capabilities was probably the first DDoS toolkit to automate the process of scanning, exploitation and deployment. Manual intervention was needed to carry out propagation. With the advent of internet worms, starting with the ramen worm[18], the propagation step has also been automated. Internet worms are a malicious piece of software that automatically scan and compromise a host after which they deploy a copy of themselves in the compromised host. Therefore the number of hosts that are compromised by them increases at an exponential rate. These worms can then be

used to launch denial of service attacks , while at the same time carrying out the scanning to detect other vulnerable hosts. Sometimes, these worms generate such high packet rates just due to aggressive scanning that, that itself leads to disruption in the normal operation of the networks being scanned.

b) Method of Propagation : DDoS attack tools are using increasingly sophisticated methods of self-propagation. The evolution in propagation methodologies has gone through the following steps.
   i) Central Chain propagation : After the compromise of a host, the mechanism used to carry out the compromise copies the attack toolkit from a central server to the new machine. Transfer protocols like HTTP, FTP or RCP are used in this transfer. In this method, discovery of the central server disables further propagation.
   ii) Back chaining propagation : The attack toolkit is copied from the attacking host on to the compromised host.
   iii) Autonomous propagation : Some worms like the Morris worm of 1988[19] and the Code Red[20] worm hard code the attack instructions into the code itself. This makes the propagation extremely fast and no external file transfer has to be initiated.

c) Establish the communication channels between the attacker , handlers and the agents. As the DDoS network starts building, it becomes hard for the attacker to keep track of all the agents which can be signaled to start a DDoS attack. The handlers facilitate this task by performing the communication with the agents. Therefore most agents will listen to handlers for commands at some well known ports. Handlers in turn need to keep track of all the agents under their control and they can do this by maintaining the list of all agents in local files. The need for agents and handlers to communicate also makes it easier to detect the presence of DDoS tools in a network. E.g if it is known that trinoo agents listen to handlers at the UDP port 31335, then the presence of an open port with the same number can alert the system administrators to the presence of the trinoo attack tool in the network. A DDoS network can also be disabled relatively easily by simply filtering out packets to that particular port. Similarly, a system integrity checker like Tripwire can signal the presence of suspicious files that may be being used to maintain agent lists.
On getting hold of a handler's agent list all the agents can be identified and disabled As a result, there has been a constant evolution and increased sophistication in the techniques used for agent/handler communication to avoid detection. These include using encryption and IRC (Internet Relay Chat) channels as the communication backbone for the DDoS network. IRC driven DDoS networks are sometimes called "botnets" because of the fact that they are software driven participants rather than human participants. Agents can then connect to well known IRC servers and be issued commands by the attacker or the handlers which also first connect to the same IRC servers. Since it is hard to distinguish these connections to the server from other legitimate connections, it becomes harder to detect the DDoS networks, Furthermore, identification of an agent or a handler will take one no further than identifying a couple of IRC servers being used by this DDoS network. To make matters worse, recent DDoS tools also come with the ability to send configuration

commands from the handlers to agents, which change the IRC server or channel being used. Therefore the DDoS network can hop from server to server and make detection even harder.

d) Launching a DDoS attack : Once the DDoS network has been set up and the infrastructure for communication between the agents and the handlers established, all that an attacker needs to do is to issue commands to the agents to start sending packets to the victim host. The agents try to send unusual data packets (all TCP flags set, repeated TCP SYN packets, Large ICMP packets) to maximize the possibility of causing disruption at the victim and the intermediate nodes.
There are certain basic packet attack types which are favorites of the attack tool designers. All the attack tools use a combination of these packet attack types to launch a DDoS attack. The basic attack types are

i) TCP floods : A stream of packets with various flags (SYN,RST, ACK) are sent to the victim machine. The TCP SYN flood works by exhausting the TCP connection queue of the host and thus denying legitimate connection requests. TCP ACK floods can cause disruption at the nodes corresponding to the host addresses of the floods as well. Also the one known tool that uses TCP ACK flooding (mstream [21]) has been known to cause disruptions in a router even with a moderate packet rate. Both TCP SYN flooding and the mstream attack constitute a group of attacks known as asymmetric attacks(Attacks where a less powerful system can render a much more powerful system useless). An interesting variation of the TCP flood is a flashcrowd[22]. A flashcrowd occurs when a lot of users simultaneously try to access a popular website leading to temporary unavailability of the site. Another term for this is 'slashdotted' as a lot of web servers suffered from such flashcrowds after being featured on Slashdot (www.slashdot.org).

ii) ICMP floods (e.g ping floods): A stream of ICMP packets is sent to the victim host. A variant of the ICMP floods is the Smurf attack in which a spoofed IP packet consisting of an ICMP ECHO_REQUEST is sent to a directed broadcast address. The rfc for ICMP specifies that no ECHO_REPLY packets should be generated for broadcast addresses, but unfortunately many operating systems and router vendors have failed to incorporate this into their implementations. As a result, the victim host (in this case the machine whose IP address was spoofed by the attacker) receives ICMP ECHO_REPLY packets from all the hosts on the network and can easily crash under such loads. Such networks are known as amplifier networks and thousands of such networks have been documented.

iii) UDP floods : A huge amount of UDP packets are sent to the victim host. Trinoo is a popular DDoS tool that uses UDP floods as one of its attack payloads.

Over time, the evolution of DDoS tools has mainly been in the increased sophistication in scanning, exploitation , propagation and the agent-handler communication. The attack traffic which is generated has remained largely the same (i.e TCP,ICMP, UDP floods as

discussed above). One of the reasons for this could be that there is no real need to evolve the attack payload as the existing ones are very effective.

3.2 Timeline and Evolution of DDoS attack tools.

As mentioned earlier, the DDoS tools have a very recent history with the first primitive tools starting emerging in 1998. Below is a brief timeline that shows the evolution of the DDoS tools over the past 5 years.

- 1998 : The initial DoS tools start to appear. They are still not truly distributed in nature , but allow the attacker to use a combination of attacks (TCP, UDP, ICMP floods).
- June 1998 : The first primitive DDoS tools start to appear. Fapi[2] is one such tool. They also allow a combination attack consisting of UDP, TCP(SYN and ACK) and ICMP floods. The tools were still hard to set up and control and were designed to be used on a DDoS network of less than 10 hosts.
- July 1999 : Wide spread deployment of DDoS networks based on tools like trinoo[2] ,TFN[2]and TFN2K[2] take place. The historyof these tools also gives some insight into the motives behind the design of such tools. Trinoo was the direct result of an IRC channel take over. A small group of users were denied access to a particular IRC channel. As a retaliatory step, this small group of users decided to take on the larger group by launching an automated DDoS attack on the IRC server. Trinoo was used to do just that. It is interesting to note that trinoo was the first attempt at client-server programming by its author and was designed and implemented in a period of months[16]. That it could cause such wide spread havoc is a grave point of concern. The attack payload of trinoo consists of UDP floods while TFN and TFN2K generate a mixture of TCP, ICMP and UDP floods. Trinoo does not do the forging of the source IP address, and so the attacking hosts can be easily detected. TFN and TFN2K make detection more difficult by using a random sequence of source addresses.

  The agent/handler communication is via hard coded handler ids inside the agents and local files with the agent names at the handler. Agents listen to inbound commands from the handler at well known ports and thus can be detected by network scanners. TFN and TFN2K start using ICMP packets (ECHO_REPLY) for handler and agent communication. ICMP ECHO_REPLY packets are allowed to pass through many firewalls since it is assumed that the communication was initiated by a host behind the firewall. These tools also start using integer ids instead of actual text commands. This makes detection by network scanners even harder. In addition TFN2K also encrypted the packets containing the commands from the handlers to the agents. The deployment of attack toolkit was still not fully automated with the attackers carefully hand selecting the hosts to be made part of the DDoS network.

- August 1999 : The Stachledraht attack toolkit starts to appear on some compromised networks. StachleDraht is a much harder toolkit to detect. It uses the same attack payloads as TFN/TFN2K but adds more control features to enable more comprehensive communication between the agents and the handlers. The communication is encrypted. The hosts to comprise the DDoS

network are still carefully selected by the attackers based on their packet generation capability.

- February 2000 – The now famous DDoS attacks take place on high profile internet sites.
- April 2000 - DDoS tool mstream is detected in the wild. This usesTCP ACK floods. As discussed earlier, the tool could bring down a much more powerful router system.
- August 2000 :The Trinity DDoS tool is developed and this is one of the first tools to start using IRC channels for the control communication.
- November 2000 : There is a shift with attackers targeting windows based agents to be used as DDoS agents[15]. Since many of the windows machines are commonly used at homes, by not so technically conscious or proficient user base, these machines pose an attractive target for attackers.
- January 2001: The ramen[18] worm is introduced. This improved attack tool distribution across hosts using automatic propagation based on the back chaining model.
- July 2001 : Sophisticated worms like the Code Red[20] worm start appearing. Code Red targets the windows operating system and also hasthe capability of turning the compromised machines into DDoS agents launching TCP SYN attacks against the victim. In some cases networks observe denial of service due to the aggressive scanning done by the worm.
- August 2001: Code Red II worm starts propagating much like the earlier Code Red worm. At the same time various IRC based agents start gaining widespread use.
- September 2001: Nimda worm outbreak. Nimda [23] combines attacks via email attachments, SMB networking, backdoors from previous attacks, exploitation of an Internet Explorer vulnerability, and exploitation of an IIS vulnerability to propagate widely. Like Code Red, propagation causes isolated DoS conditions.
- January 2003 : The SQL slammer worm[7] exploits a vulnerability in Microsoft's SQL server and is the fastest worm in history. It doubles every 8.5 seconds and is able to infect 90% of vulnerable hosts within minutes. It also causes wide spread network outages and unforeseen consequences like cancelled airline flights and ATM failures.

As we can see, the complexity and scale of DDoS attacks has kept on increasing with time and are expected to do so in the future as well. Consequently there is an urgent need to come up with DDoS detection and mitigation techniques.

4 Defense against DDoS attacks

DDoS attacks pose the most potent threat to the network infrastructure today. Sadly, there are no really effective mechanisms in place today to defend against an on going DDoS attack. There has been active research going on in this field, but most of the solutions proposed are either in the initial stages of development or assume more functionality from the Internet Protocol, which is difficult to gain in reality. Some of these ideas will be discussed later on in the section.

Since DDoS attack mitigation is poses such a challenge, more stress should be laid on prevention of such attacks. This would require more conscious effort to be put into the security of an organization and its internal networks. The first step that any organization should take is to come up with a security policy with part of it dedicated to DDoS attack prevention and mitigation.
There should be explicit mention of the steps that are to be taken before, during and after a DDoS attack.

- **Before the attack** : The first step to prevent a DDoS attack is to prevent compromise and use of hosts as agents. To achieve this, the whole network should be guarded by the firewall. The time between the posting of an exploit and its use by attackers is continuously diminishing. System and network administrators need to keep up with this pace and be diligent in applying patches supplied by the various vendors. I believe that this is one of the most important aspects that should have explicit mention in the security policy of any company. Responsibilities for keeping track of various patches should be divided among a group of people (if the size of the organization allows that) on the basis of operating system and major software applications. This brings up the related issue of the level of training and motivation of system administrators. Organizations should consider spending more on the training and compensation of system administrators. They should be able to track down the onset of a DDoS attack; the type of attack tool used and after the incident be able to aid the law agencies in tracking down the hosts belonging to the DDoS network and possibly the attacker himself. There should be improved intrusion detection systems like (Snort[24]) in place. Hosts and networks should be constantly audited for DDoS tools by using DDoS detectors like RID[25].
- **During the attack** :During a DoS attack, it becomes difficult (if not impossible) for the sysadmins to get access to the routers and servers of their network. Therefore there should be some understanding between the organization and its upstream service provider. The upstream provider is in a better position to throttle down the attack packets and to also be able to gather information to aid in forensic analysis. Furthermore, with some re-engineering, the whole network should be built in such a way that there is a separate network operations network isolated from the underlying data transfer network. This network could even be in terms of serial cables to the various routers, such that these can be used to turn off packet rates and maybe transfer data to be used later on in forensic analysis.
- **After the attack** : It is absolutely imperative that there should be an intrusion response team in place. This team should be able to gather data after an attack to be able to identify the type of attack being carried out. This analysis can aid in tracking down the hosts that form the DDoS network so that they can be shut down. There should also be closer co-operation between law enforcement agencies to be able to gather and submit evidence that can be used to prosecute an attacker.

Denial of Service is made possible due to the basic deficiencies in internet design and level of security of various internet hosts. Consequently, unless those deficiencies are addressed , we will have to live with DDoS attacks. Even if a particular network is able to secure its own assets, it does not secure itself against DDoS attacks as other compromised hosts can still be used to launch attacks on it. Therefore, since such a time that internet is secure and DDoS attacks hard to generate, there will be a need for (and a huge market for) solutions that mitigate the effect of a DDoS attack on a network. In the next section, I review some of the mechanisms for DDoS defense that have been proposed in the recent past.
It is important to note that there is no single solution that can solve all kinds of DDoS attacks, but that in practice a combination of the defense mechanisms proposed below will have to be used in tandem to be effective.
Generally, DDoS defense mechanisms take 3 response steps. The first step is to detect an attack. In addition to detecting the attack the mechanism should be able to pin point the exact packet characteristics of the attack payload that characterizes the attack. This classification can then be fed to an attack mitigation scheme that rate limits or filters out the malicious packets. Therefore there is a close interaction between attack detection and rate-limiting strategies and this paper discusses such strategies below.
The final step or a step that can be taken concurrently while attack mitigation is taking place is the problem of IP traceback. In IP traceback schemes, the true source of the packets is discovered. E.g if an attacker uses host A to launch an attack using the spoofed source address of host B, the problem of IP Traceback involves finding out host A. This can be accomplished if there is a way of traversing all the routers from A to the victim in the reverse order, hop by hop and finally reaching host A. We will discuss some mechanisms that achieve this.
The next section gives more detailed information about DDoS detection, mitigation and traceback techniques.

4.1 Defense Mechanisms

1) Ingress/Egress Filtering[11,12] :

Ingress/Egress filtering makes it difficult for attackers to launch attacks using spoofed IP addresses. As we have seen that IP spoofing is required for some attacks like the Smurf (ping flood) to work. Furthermore, IP spoofing makes it difficult to trace back the attack to the actual originating host. If on detection of a DDoS attack, the traffic is dropped based on just the IP source address, then the network whose source address was spoofed is also denied access. This in itself is a denial of service for the end-users on that network.
Any firewall, connecting a network to the internet will have some interfaces connected to the internal network and some interfaces connected to the internet. The firewall should apply ingress filtering on the external interfaces and drop all packets that have the source address which belongs to its internal network, since such packets have been clearly spoofed. If such packets are allowed into the network, then the attacker can masquerade as a host within the same network. There is generally a higher level of trust between hosts on the same network and this can lead to security compromises. Egress filtering is applied on the internal interface on packets that are heading out of the network. The

firewall drops all the packets that have source addresses that do not belong to their local network. This stops an attacker from using hosts within that network as DDoS agents. If these two solutions are widely deployed all over the internet, then they will go a long way in stopping all attacks that rely on IP spoofing to be effective. Furthermore, they will enable easy traceback of the attacks to the true origin as the attacking hosts are forced to use their true IP addresses. Ingress/Egress filtering do not provide protection against bandwidth based DDoS attacks though.
Ingress and Egress filtering depend on their widespread use for their efficacy. Unfortunately, not many ISPs today do enforce Ingress filtering, either due to lack of awareness, the administrative burden or to allow applications like Mobile IP to work. There are therefore some proposals [27] that enhance the routing protocol on the internet to automatically drop packets that can potentially have spoofed IP source address. Routers in the internet already exchange routing information regarding the reachability to various destinations. These protocols can be enhanced to also incorporate information about valid source address prefixes that can be observed on a particular router interface. The router's forwarding function normally just observes the destination IP address on a packet and decides the next hop to which to forward the packet to. This function can now be enhanced to drop packets which have been determined to have invalid prefixes based on the routing information exchange.

2) IP traceback. : IP traceback is the process of tracing back the forged IP packets to their true sources rather than the spoofed IP address that was used in the attack. At a very basic level, you can think of this as a manual process in which the administrator of the network under attack places a call to his ISP (whose router is just one hop away) asking for the direction from which the attack packets are coming. The upstream ISP finds out that information, but has to call its own upstream router to find out the previous hop and so on. Therefore co-operation between different networks is required to be able to traceback attack packets to their true sources. Since the manual traceback is very tedious there have been various proposals in the recent past to automate this process. The three main ways of doing a packet traceback are given below.
i) Link testing schemes[30] – In this scheme the victim tests each of its incoming links as a probable input link for the DDoS traffic . Burch and Cheswick[30] propose a scheme called controlled flooding to determine the loaded link. The victim generates some load on each of the links coming into it and observers the perturbation observed in the input packet rate. The idea being that the loaded link will suffer from the most perturbation. This process is recursively followed as the victim generates loads across links farther and farther upstream till the source is reached, This scheme assumes that the victim has the complete map of all the paths to all its possible internet sources. Furthermore, this analysis can only be done during a DDoS attack and has no post-mortem value. It can also be argued that it would be hard for the victim to be able to generate the packets for flooding while it is under a DDoS attack, Some people have argued that controlled flooding of various links might in itself constitute a denial of service attack. Link testing mechanisms work best when there is a single attacking source and give bad results under a distributed denial of service attack.
ii) Packet Marking schemes[30,13] : In packet marking schemes, each router in addition to forwarding a packet also inserts a mark in the packet. This mark is a unique identifier

corresponding to this particular router. As a result the victim can determine all the intermediate hops for each packet by observing the inserted marks . There are 2 variants to this marking scheme. First is the deterministic packet marking scheme[13,30] in which each router marks all the packets passing through it with its unique identifier. This scheme is thus similar to the IP record-route option. This makes the reconstruction of the attack path at the victim trivial. But the downside to this scheme is that routers are slowed down as they have to perform additional functionality. Furthermore, the packet headers can grow up to an arbitrary size and so provisions have to be made for this. Note, that we cannot have an upper bound on the header size (and consequently the number of routers that can mark a packet) as the attacker can then generate bogus packets that already have false information filled in the limited number of entries available.
To overcome the deficiencies of the deterministic packet marking scheme, a probabilistic packet marking scheme has been proposed. In this scheme there is just a single entry in the IP header to store the markings. Each router on the path from the source to the destination writes down its unique identifier in the entry in the packet header with some probability. By writing into the entry, routers overwrite any previous entry that was present there. The victim can reconstruct the path from the source to itself on receiving a large number of packets. It is even possible to reconstruct the order of routers from the source to the victim based on the relative frequencies of the router markings in the packets. E.g if a packet follows the path A->B->C->D->E->F; where B,C,D,E are intermediate routers, then the relative frequencies of the packet markings found at F would be in the decreasing sequence E>D>C>B as E would overwrite the markings made by the previous routers.
A downside of this scheme is that some packets will not be overwritten by any of the routers. The attacker can therefore write bogus information in all the packets knowing that some of these packets will get through and confuse the victim[31]. This method also does not work well for denial of service attacks that can work without a lot of packets as it requires a large number of packets to converge.

iii) ICMP traceback messages[14] - This method is similar to the probabilistic packet marking technique, but instead of marking the packets, routers send newly proposed ICMP messages to the destination, with the information about the previous hop. The scheme proposes sending an ICMP message for every 20,000 packets forwarded.
Thus the overhead for the scheme is minimal, but this scheme also gives complete path information after only after forwarding multiple packets. In this scheme the attacker can also generate bogus ICMP traceback messages to confuse the victim. Therefore there has to be some provision for verifying the authenticity of an ICMP message, without incurring too much of an overhead.

3) Rate Limiting mechanisms – Rate limiting mechanisms limit the rate of packet arrivals which match the criteria for DDoS attacks, It is important that rate limiting mechanisms only limit the rate of malicious packets and do not harm legitimate flows. Furthermore, these rate limiting mechanisms should not incur a lot of extra overhead and they shouldn't become a source of denial of service attacks themselves.
Rate limiting attacks can also be thought of as a less severe form of packet filtering.If it is known that the attack detection mechanism can come up with many false positives, it is

better to go for rate limiting rather than packet filtering. Rate limiting can also be thought of as packet shaping using which all the internet sources are forced to respect some constraints imposed on their forwarding rate.

A novel aspect is to rate limit aggregates[26] rather than IP source addresses. Aggregates are defined as a subset of traffic defined by some characteristic like a particular destination address. They can be classified on the basis of diverse criteria such as UDP/TCP source ports or IP destination addresses. Routers detect aggregates overwhelming it by using samples of packet drops in the queues. They then send a pushback message [32] to the upstream router along with the information about the aggregate to rate limit and the value of the rate limit. If the aggregate packet respects the rate limit, it is allowed to sail through, otherwise the packets are dropped to conform to the rate limit and pushback messages are recursively propagated to upstream routers.

There are also mechanisms designed for the protection of servers (web servers in particular) from high traffic rates. One such approach [29] involves a server under stress installing rate throttles at a subset of its upstream routers. On installing such throttles all the traffic passing through the router to the source S is rate limited to the throttle rate. This scheme can shown to distribute the total capacity of the server in a max-min fair way among the routers servicing it. This means that only aggressive flows which do not respect their rate shares are punished and not the other flows.

There are other rate limiting schemes [8,28] that detect bandwidth attacks by noticing an asymmetry between the packets traveling to and from a network. If a host is not replying with as many packets as are being sent to it, this could be an indication of the host being attacked by DoS traffic.

The DWARD system [28] is meant to be installed at the edge routers for a network. The system monitors the traffic being sent to and from the hosts in its interior. If it notices asymmetry in the packet rates generated by an internal host, it rate limits the packet rate. Thus, this is a solution that does DDoS attack detection at the source (much like egress packet filtering). The downside here is that there is a possibility of numerous false positives while detecting DDoS conditions near the source. This is because there might be asymmetry in the packet rates for a short duration. Furthermore, some legitimate flows like real time UDP flows do exhibit asymmetry.

MULTOPS[8] is a multilevel data structure that can be used to keep track of asymmetric flows passing through a router. It stores packet rate statistics for flows between hosts (or subnets) A and B using either A's or B's IP address. When it stores the statistics based on source addresses, it is said to operate in attack oriented mode, otherwise in the victim oriented mode. A MULTOPS data structure can thus be used for keeping track of attacking hosts or hosts under attack. When the packet rate to or from a subnet reaches a certain threshold, a new sub-node is created to keep track of more fine – grained packet rates. This process can go till finally per IP address packet rates are being maintained. Therefore, starting from a coarse granularity one can detect with increasingly finer accuracy, the exact attack source or destination addresses. The IP source addresses that are obtained are spoofed addresses, but can still be valuable in applying rate limits.

As we can see, there is no dearth of novel DDoS detection and mitigation schemes being invented. Though none of them single handedly can provide protection against the problem, they can provide useful and powerful tools in alleviating the problem.

5 What does the future hold?

As discussed previously, DDoS attacks are made possible by inherent flaws in the internet design and the lack of proper security mechanisms in numerous computer systems. The problem is only going to be made more severe in the future. There are millions of computers being added to the internet every year. We can be sure that there are not going to be millions of new system administrators for these new hosts.

Many of these systems will be used by home users with permanent IP addresses on a broadband connection. This enriches an already highly target rich environment for attackers to scout for systems that can be used as DDoS attack agents. It will also be difficult to gather useful forensic evidence from systems being run by people who might not even under stand the concept of a TCP/IP stack, let alone gather useful forensic evidence.

The newer variants of worms like the Code Red Worm and the SQL Slammer worm are extremely aggressive and can lead to an increased collateral damage. Systems not directly attacked but connected to the attacked systems can also freeze up. This is what happened when many ATM machines stopped working during the time of the slammer worm attack. It is scary to think of the consequences if critical military or medical systems are brought down due to DDoS attacks launched. The situation looks even more grave when one considers the possibility that DDoS attacks could become tools of warfare used to target critical infrastructure of other countries.

Though "super worms"[33] that could propagate through the whole internet within minutes had been proposed in the literature, the spread of the Slammer worm showed it to be a reality. These worms spread at such a fast rate that system administrators cannot be expected to respond in time. Therefore there is a need to do more research into tools to scan the network for possible intrusions automatically.

Finally, I feel that there is a very urgent need for the software industry to mature and take security more conscientiously. The stress till now has been to build things that are fast and convenient to use rather than secure. The general perception among the public is also partly responsible for this. The public in many cases is not willing to spend the extra amount needed to design secure software. To take an example, an airline whose aircrafts crash once every month is bound to see a huge loss in revenue because of people refusing to use them, but at the same time systems that crash much more frequently enjoy a huge market. I believe that such perceptions will change over time with security increasingly becoming more important to customers as well. One just hopes that it does not take too long a time.

It will also not be long before major corporations start holding the software companies responsible for major losses suffered due to security incidents. I believe that software developers will not be able to escape accountability in at least some cases, forcing them to invest more in the security of the software to escape lawsuits and loss of market.

I also feel that government will increasingly take more interest in regulating the quality and security in the software industry. Just like other industries like the automobile industry have to comply with minimum standards of safety, so will the software industry. I think this will be a welcome move and go a long way in making the computer industry more mature and increase user confidence.

References

[1] A Taxonomy of DDoS attacks and DDoS defense Mechanisms. Jelena Mirkovic, Janice Martin and Peter Reiher. Computer Science Department. University of California , Los Angeles, Technical Report #020018.

[2] Distributed Denial of Service (DDoS) Attacks/tools. http://staff.washington.edu/dittrich/misc/ddos

[3] CNN. Cyber-attacks batter Web heavyweights, February 2000. Available at http://www.cnn.com/2000/TECH/computing/02/09/cyber.attacks.01/index.html.

[4] CNN "Immense" network assault takes down Yahoo, February 2000. Available at http://www.cnn.com/2000/TECH/computing/02/08/yahoo.assault.idg/index.html.

[5] Netscape. Leading web sites under attack, February 2000. Available at http://technews.netscape.com/news/0-1007-200-1545348.html.

[6] CERT coordination center. Denial of Service attacks. http://www.cert.org/tech_tips/denial_of_service.html

[7] David Moore, Vern Paxson, Stefen Savage, Colleen Shannon, Stuart Staniford, Nicholas Weaver.The spread of the Sapphire Slammer worm. http://www/silicondefense.com/sapphire.

[8] MULTOPS: a data-structure for bandwidth attack detection**.** Thomer M. Gil, Massimiliano Poletto. In the Proceedings of the 10th USENIX Security Symposium, Washington D.C., August 2001.

[11] P. Ferguson et. al. RFC 2267. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. Technical report, The Internet Society, 1998.

[12] SANS Institute. Egress filtering v 0.2, 2000. http://www.sans.org/y2k/egress.htm.

[13] Stefan Savage, DavidWetherall, Anna Karlin, and Tom Anderson. Practical Network Support for IP Traceback. Technical report, Department of Computer Science and Engineering, University of Washington, 2000.

[14] Bellovin. ICMP Traceback Messages. Technical report, AT&T, 2000. http://www.ietf.org/internet-drafts/draft-bellovin-itrace-00.txt

[15] Trends in Denial of Service Technology . http://www.cert.org/archive/pdf/DoS_trends.pdf

[16] Usenix Security Symposium, 2000 talk by Dave Dittrich. ftp://ftp.ddj.comT/technetcast/mp3/tnc-0417-24.mp3 .

[17] Analysis of the t0rn rootkit. http://www.securityfocus.com/infocus/1230

[18]Analysis of the ramen worm. http://www.whitehats.com/library/worms/ramen

[19] Eugene H. Spafford. The internet worm program: An analysis. Computer Communication Review, 19(1):17-57, January 1989.

[20] CERT Advisory CA-2001-19 "Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL. http://www.cert.org/advisories/CA-2001-19.html.

[21] The mstream distributed denial of service attack tool. http://staff.washington.edu/dittrich/misc/mstream.analysis.txt

[22] Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites. Jaeyeon Jung, Balachander Krishnamurthy. Proceedings of the International World Wide Web Conference, pages 252--262. IEEE, May 2002

[23] CERT® Advisory CA-2001-26. Nimda Worm.http://www.cert.org/advisories/CA-2001-26.html.
[24] Network Intrusion detection using snort.
http://www.linuxsecurity.com/feature_stories/feature_story-49.html
[25] Remote Intrusion Detector. http://www.theorygroup.com/Software/RID/
[26] Vern Paxson, Steve Bellovin, Sally Floyd and Ratul Mahajan. Controlling high bandwidth aggregates in the network. Technical report
[27] K. Park and H. Lee. On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets. ACM SIGCOMM 2001.
[28]Jelena Mirkovic, Peter Reiher, Gregory Prier. Attacking DDoS at the source. International Conference on Network protocols, 2002.
[29] David Yau, John C. S. Lui, Feng Liang. Defending against distributed denial of service attacks using max-min fair server centric router throttles. IEEE international conference on Quality of Service. 2002.
[30] Hal Burch and Bill Cheswick. Tracing anonymous packets to their approximate source. In Proceedings of the USENIX Large Installation Systems Administration Conference, pages 319--327, New Orleans, USA, Decemeber 2000.h
[31]K. Park and H. Lee. On the effectiveness of probabilistic packet marking for IP Traceback under a denial of service attack. In Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies, 2001.
[32] Vern Paxson, Steve Bellovin, John Ioannidis, Kireete Kompella, Sally Floyd and Ratul Mahajan. Pushback messages for controlling high bandwidth aggregates in the network. Internet Draft, work in progress.
[33] Vern Paxson, Stuart Staniford, and Nicholas Weaver, How to 0wn the Internet in Your Spare Time, Proceedings of the 11th USENIX Security Symposium (Security '02).