# Defenses Against Distributed Denial of Service Attacks

**Gary C. Kessler**
**November 2000**

**This paper was submitted as the practical exercise in partial fulfillment for the SANS/GIAC Security Essentials Certification (GSEC). A much expanded version will be published as Chapter 11, "Denial of Service Attacks" (by Diane E. Levine and GCK) in the upcoming 4th edition of the *Computer Security Handbook*, edited by M.E. Kabay and S. Bosworth (John Wiley & Sons, in preparation).**

This short paper discusses defenses against Distributed Denial of Service (DDoS) attacks. DoS attacks are of particular interest and concern to the Internet community because they seek to render target systems inoperable and/or target networks inaccessible. "Traditional" DoS attacks, however, typically generate a large amount of traffic from a given host or subnet and it is *possible* for a site to detect such an attack in progress and defend themselves. Distributed DoS attacks are a much more nefarious extension of DoS attacks because they are designed as a coordinated attack from many sources simultaneously against one or more targets.

This paper will focus on DDoS attacks only and assumes some basic familiarity with different DoS attacks. Rather than describe specific DDoS attacks in detail, this paper will define generic DDoS terms and ways in which service providers and user sites can defend themselves against these attacks.

## A Short History of DDoS

Denial-of-service attacks under a number of guises have been around for decades. Distributed DoS attacks are much newer, first being seen in late June and early July of 1999. The first well-documented DDoS attack appears to have occurred in August 1999, when a DDoS tool called Trinoo (described below) was deployed in at least 227 systems, of which at least 114 were on Internet2, to flood a single University of Minnesota computer; this system was knocked off the air for more than two days.

The first well-publicized DDoS attack in the public press was in February 2000. On February 7, *Yahoo!* was the victim of a DDoS during which its Internet portal was inaccessible for three hours. On February 8, Amazon, Buy.com, CNN, and eBay were all hit by DDoS attacks that caused them to either stop functioning completely or slowed them down significantly. And, on February 9, E*Trade and ZDNet both suffered DDoS attacks. Analysts estimated that during the three hours Yahoo was down, it suffered a loss of e-commerce and advertising revenue that amounted to about $500,000. According to book seller Amazon.com, its widely publicized attack resulted in a loss of $600,000 during the 10 hours it was down. During their DDoS attacks, Buy.com went from 100% availability to 9.4%, while CNN.com's users went down to below 5% of normal volume and Zdnet.com and E*Trade.com were virtually unreachable. Schwab.com, the online venue of the discount broker Charles Schwab, was also hit but refused to give out exact figures for losses. One can only assume that to a company that does $2 billion dollars weekly in online trades, the downtime loss was huge.

In a DDoS attack, the attacking packets come from tens or hundreds of addresses rather than just one, as in a "standard" DoS attack. Any DoS defense that is based upon monitoring the volume of packets coming from a single address or single network will then fail since the attacks come from all over. Rather than receiving, for example, a thousand gigantic Pings per second from an attacking site, the victim might receive one Ping per second from 1000 attacking sites.

One of the other disconcerting things about DDoS attacks are that the handler can choose the location of the agents. So, for example, a handler could target several NATO sites as victims and employ agents that are all in countries know to be hostile in NATO. The human attacker, of course, might be sitting in Canada.

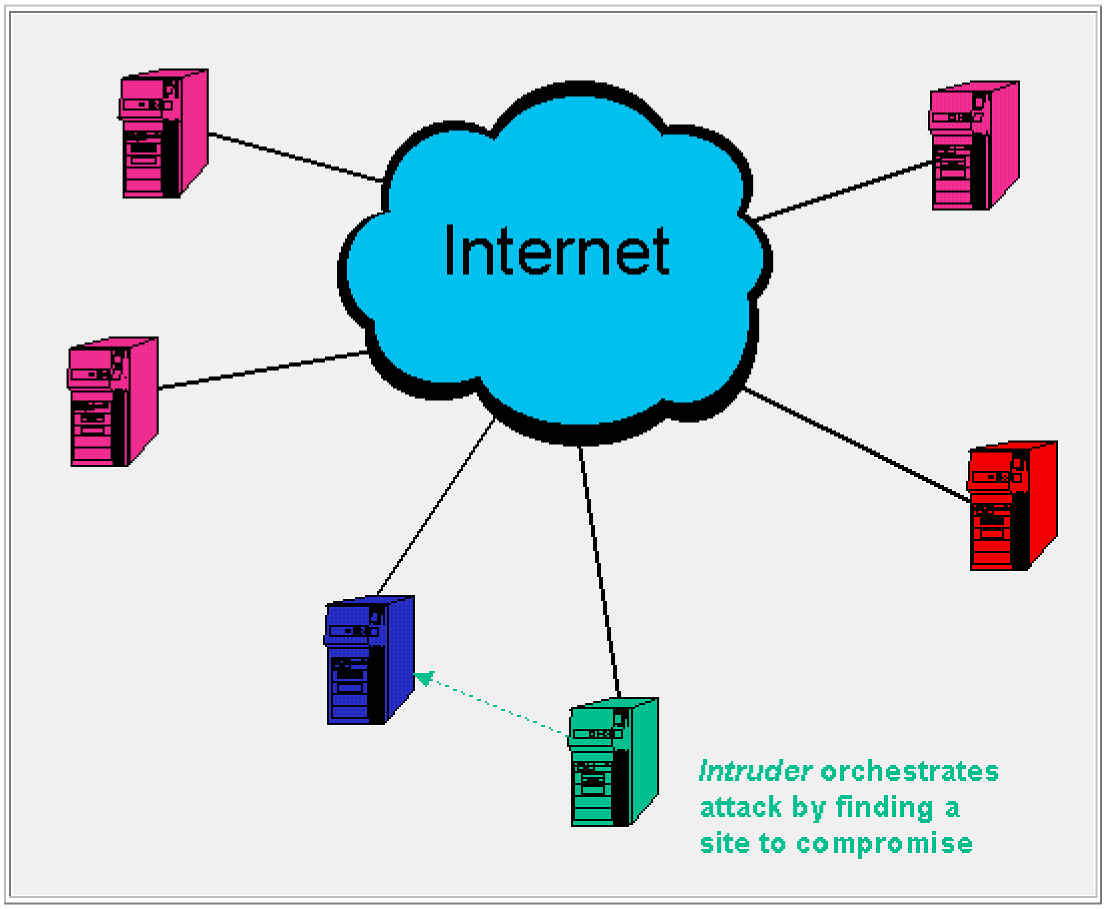
Like DoS attacks, all of the DDoS attacks employ standard TCP/IP messages -- but employ them is some non-standard ways. Common DDoS attacks have such names as Tribe Flood Network (TFN), Trin00, Stacheldraht, and Trinity. Some details about these will be presented in the following sections.

## DDoS Terminology and Overview

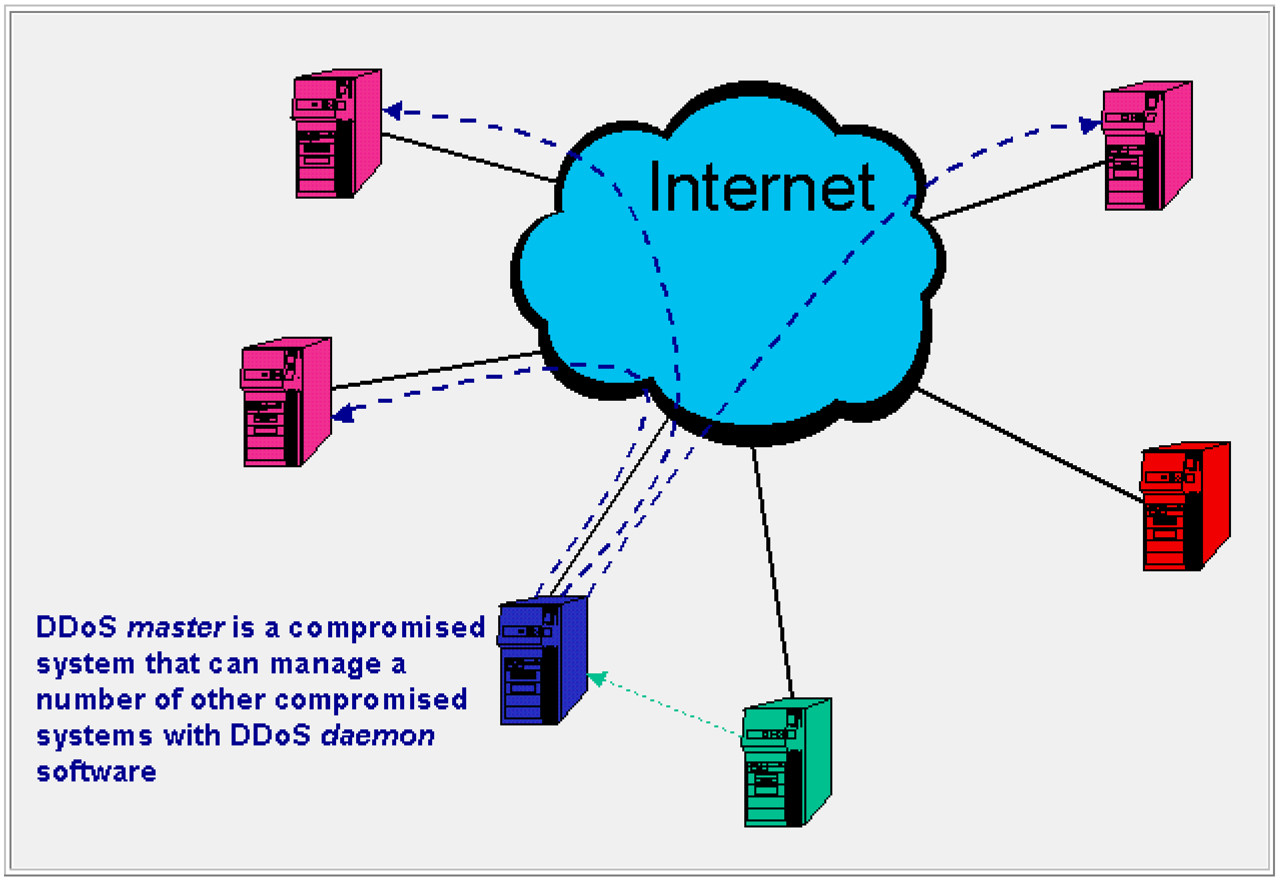
To describe and understand DDoS attacks, it is important to understand the terminology that is used to describe the attacks and the tools. While the industry has more or less settled upon some common terms, that consensus did not come about until well after many DoS/DDoS attacks had already appeared in the hacker and mainstream literature.

DDoS attacks always involve a number of systems. A typical DDoS attack scenario might follow roughly the following steps:

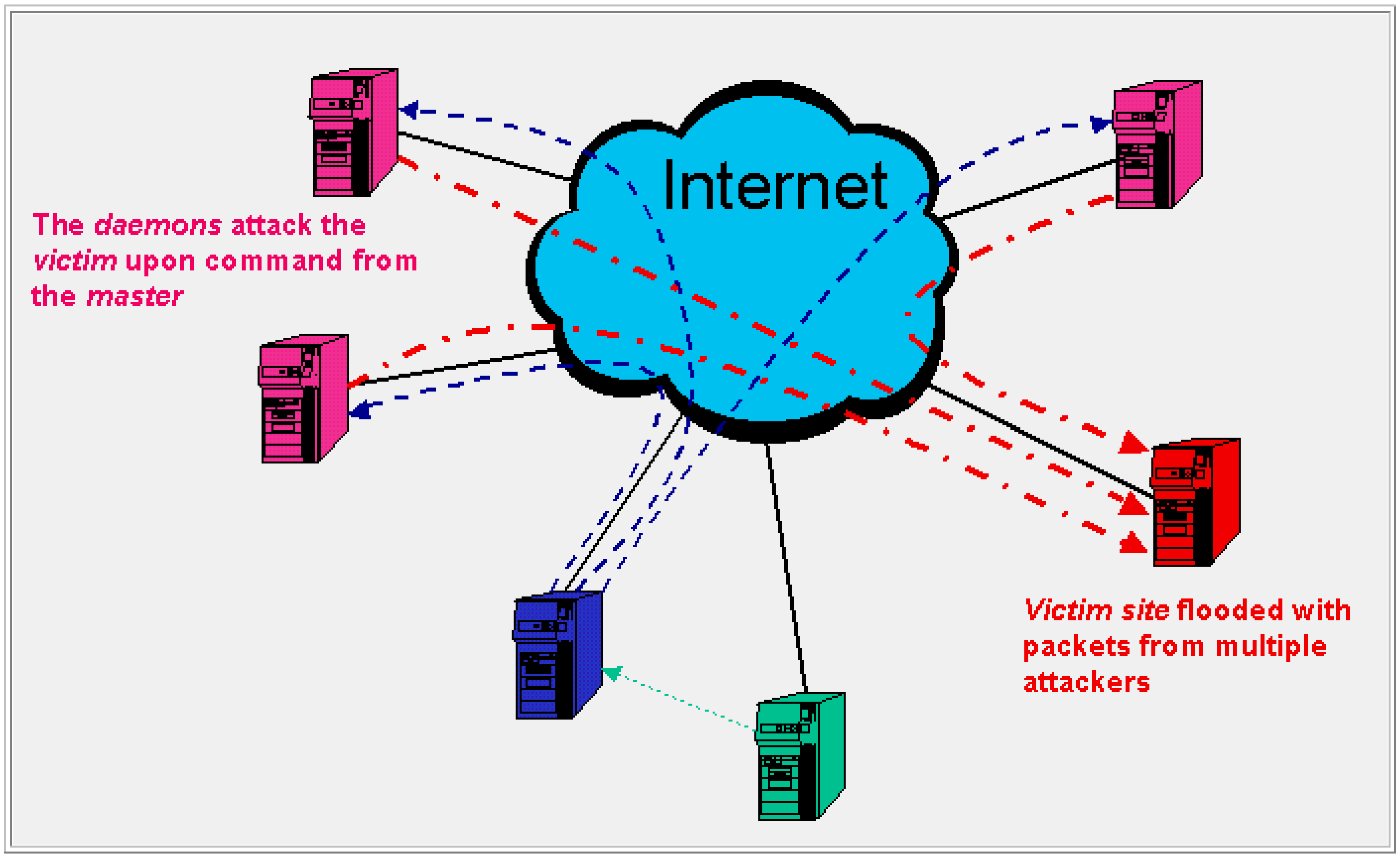
1. An *intruder* finds one or more systems on the Internet that can be compromised and exploited (see figure below). This is generally accomplished using a stolen account on a system with a large number of users and/or inattentive administrators, preferably with a high-bandwidth connection to the Internet (many such systems can be found on college and university campuses).


Internet
*Intruder* orchestrates attack by finding a site to compromise

2. The compromised system is loaded with any number of hacking and cracking tools such as scanners, exploit tools, operating system detectors, root kits, and DoS/DDoS programs. This system becomes the DDoS *master*. The master software allows it to find a number of other systems that can themselves be compromised and exploited. The attacker scans large ranges of IP network address blocks to find systems running services known to have security vulnerabilities. This *initial mass-intrusion phase* employs automated tools to remotely compromise several hundred to several thousand hosts, and installs DDoS agents on those systems. The automated tools to perform this compromise is *not* part of the DDoS toolkit but is exchanged within groups of criminal hackers. These compromised systems are the initial victims of the DDoS attack. These subsequently exploited systems will be loaded with the DDoS *daemons* that carry out the actual attack (see figure below).


Internet
DDoS *master* is a compromised system that can manage a number of other compromised systems with DDoS *daemon* software

3. The intruder maintains a list of *owned systems*, the compromised systems with the DDoS daemon. The actual *denial of service attack phase* occurs when the attacker runs a program at the master system that communicates with the DDoS daemons to launch the attack. Here is where the intended DDoS victim comes into the scenario (see figure below).


Internet
The *daemons* attack the *victim* upon command from the *master*
*Victim site* flooded with packets from multiple attackers

Communication between the master and daemons can be obscured so that it becomes difficult to locate the master computer. Although some evidence may exist on one or more machines in the DDoS network regarding the location of the master, the daemons are normally automated so that it isn't necessary for an ongoing dialogue to take place between the master and the rest of the DDoS network. In fact, techniques are typically employed to deliberately camouflage the identity and location of the master within the DDoS network. These techniques make it difficult to analyze an attack while in progress and also to block attacking traffic and trace it back to its source.

In most cases, the system administrators of the infected systems don't even know that the daemons have been put in place. Even if they do find and eradicate the DDoS software, they can't help anyone determine where else the software may have been placed. Popular systems to exploit are a site's Web, e-mail, name, or other servers since these systems are likely to have a large number of open ports, a large amount of traffic, and are unlikely to be quickly pulled off-line even if an attack can be traced to them.

A final word on terminology is necessary. Early descriptions of DDoS tools used a jumble of terms to describe the various roles of the systems involved in the attack. At the CERT Distributed System Intruder Tools workshop held in November 1999, some standard terminology was introduced and those terms are used in the paragraphs above.

To align those terms and the terms used by the hacker literature as well as early descriptions, we find the following synonyms:

- Intruder: Also called the *attacker* or *client*
- Master: Also called the *handler*
- Daemon: Also called an *agent*, *bcast (broadcast) program*, or *zombie*
- Victim: Always the victim

It should not go without saying that DoS/DDoS attacks actually have two victims, namely the ultimate target as well as the intermediate system(s) that were exploited and loaded with daemon software. Although we tend to refer to the site that is eventually brought down as the victim, the intermediate systems from where the attack is launched have also been victimized. In this chapter, we will focus on the end-of-the line DoS/DDoS victim.

## Some of the DDoS Tools

While this paper focuses on defensive measures against DDoS, it is important to know the names of the major tools to see their commonality -- and how they have already evolved!! By design, this section will be very brief; the reference section will provide a resources for additional information.

In rough chronological order, the DDoS tools commonly seen today include:

- *Trinoo*, also called *Trin00*, was the first known DDoS tool, starting to appear in June or July 1999. Trin00 is a distributed SYN DoS attack, where masters and daemons communicate using the ports shown in the table below.

- The Tribe Flood Network (TFN) started to appear after trinoo. TFN client and daemon programs implement a DDoS network capable of employing a number of attacks, such as ICMP flood, SYN flood, UDP flood, and SMURF style attacks. TFN is noticeably different than trinoo in that all communication between the client (attacker), handlers, and agents use ICMP ECHO and ECHO REPLY packets. Communication from the TFN client to daemons is accomplished via ICMP ECHO REPLY packets. The absence of TCP and UDP traffic sometimes makes these packets difficult to detect because many protocol monitoring tools are not even configured to capture and display the ICMP traffic.

- Stacheldraht (German for "barbed wire") is a DDoS tool that started to appear in the late summer of 1999 and combines features of trinoo and TFN. It also contains some advanced features, such as encrypted attacker-master communication and automated agent updates. The possible attacks are similar to those of TFN; namely, ICMP flood, SYN flood, UDP flood, and SMURF attacks.

- In August 2000, a DDoS attack against Apache Web servers was first detected. The attack took advantage of a vulnerability whereby a URL sent to an Apache Web server containing thousands of forward slashes ("/") would put the server into a state that would consume enormous CPU time. This particular attack was launched by over 500 compromised Windows computers and would, presumably, succeed against Apache Web servers prior to version 1.2.5.

- During the following month, a new DDoS tool called Trinity was reported. Trinity is capable of launching several types of flooding attacks on a victim site, including UDP, fragment, SYN, RST, ACK, and other floods. Communication from the handler or intruder to the agent, however, is accomplished via Internet Relay Chat (IRC) or AOL's ICQ; Trinity appears to use primarily port 6667 and also has a backdoor program that listens on TCP port 33270.
- In November 1999, the Shaft DDoS tool became available. A Shaft network looks conceptually similar to a trinoo; it is a packet flooding attack and the client controls the size of the flooding packets and duration of the attack. One interesting signature of Shaft is that the sequence number for all TCP packets is 0x28374839.
- Tribe Flood Network 2K (TFN2K) was released in December 1999. TFN2K is a complex variant of the original TFN with features designed specifically to make TFN2K traffic difficult to recognize and filter, remotely execute commands, hide the true source of the attack using IP address spoofing, and transport TFN2K traffic over multiple transport protocols including UDP, TCP, and ICMP. TFN2K attacks include flooding (as in TFN) and those designed to crash or introduce instabilities in systems by sending malformed or invalid packets, such as those found in the Teardrop and Land attacks.

| **DDoS Tool** | **Intruder-to-master Communication** | **Master-to-daemon Communication** | **Daemon-to-master Communication** |
|---|---|---|---|
| Trinoo | 27665/tcp | 27444/udp | 31335/udp |
| TFN | ICMP Echo/Echo Reply | ICMP Echo Reply | ICMP Echo/Echo Reply |
| Stacheldraht | 16660/tcp | 65000/tcp | ICMP Echo Reply |
| Trinity | 6667/tcp | 6667/tcp (also 33270/tcp) | |
| Shaft | 20432/tcp | 18753/udp | 20433/udp |

The tools listed here are the best known, and most widely sued, but they are not the only ones and more tools are becoming available. In addition, the distribution of zombie software is not always due to an attacker exploiting a vulnerability of an exposed system. Indeed, the user is very often the "culprit" and Trojan horses are often the mechanism for distributing the zombie code. The SubSeven Defcon8 software, for example, is a backdoor virus that is rapidly spreading. SubSeven gets on a user's system because it is distributed within programs available via Usenet and other Internet sites, such as some game or pornography programs (e.g., SexxxyMovie.mpeg). Computer systems today are frequently scanned for the presence of SubSeven and provides a potential door into users' systems.

## DDoS Defenses

The earlier part of this paper has been the preamble to discussing how sites can protect themselves from these kinds of attacks. The truth is that a site cannot defend itself from DDoS attacks alone. DDoS attacks depend upon the "community" of the Internet and defenses, therefore, depend upon the Internet community acting like a community with a common interest. And defending against attack includes ensuring that our own sites are not the source of attacks and our own networks do not forward attacks. This section will discuss some methods that will help prevent the spread of DDoS attacks, by limiting the distribution of the tools and/or limiting the propagation of the offending "attack" packets.

Although not discussed in detail here, another point needs to be made about DDoS attack responses. If you are the victim of such an attack, maintain detailed logs of all actions you take and events you detect. These logs may prove invaluable in subsequently understanding the attack that occurred, preventing other attacks at your site and others, and aiding law enforcement efforts to track down the perpetrators.

### User and System Administrator Actions

Despite the best intentions and even the best distributed system management tools, the fact is that most computers today are largely managed by the local user. That means, in essence, that the local system has no security protections. But whether the local host computer is a secretary's desktop system or the Web server for a company, there are steps that can and should be taken to minimize the potential that an individual system will be compromised and itself attacked or used as a stepping-stone to attack others:

1. Keep abreast of the security vulnerabilities for all of your site's hardware, operating systems, and application and other software. This sounds like a Herculean task but it is essential to safeguarding your network. Apply patches and updates as soon as possible. Standardize on certain hardware, operating systems, and software where feasible to help manage the problem.

2. Consider using some form of "personal" firewall software to help detect an attack at your systems.

3. Monitor your system periodically to test for known operating system vulnerabilities. Also periodically check to see what TCP/UDP ports are in use using the `netstat -a` command; every open port should be associated with a known application. Turn off all unused applications.

4. Regularly monitor your system logs and look for suspicious activity.

5. Use available tools to periodically audit your systems, particularly servers, to ensure that there have been no unauthorized/unknown changes to the file system, registry, user account database, etc.

6. Do not download software from unknown, untrusted sites. If possible, know the author of the code. Even better is to download source code, review it (where feasible), and compile it on your system, if possible, rather than downloading binaries or executables.

7. Follow CERT, SANS, and TruSecure (ICSA) best practices procedures.

## Local Network Actions

Even if a user locks down their system so that no vulnerability has gone unpatched and no exposure unprotected, the network itself -- including the user community -- can still be at risk. There are a number of steps that local network managers and network administrators can take to protect all of their own users as well as the rest of the Internet community:

1. Every network connected to the Internet should perform egress address filtering at the router. *Egress filtering* means that the router should examine the IP Source Address field of every outgoing packet to the Internet to be sure that the NET_ID matches the NET_ID of the network. The user's firewall has historically been used to protect the user from attacks from the outside world. But those attacks come from somewhere so sites should also use the firewall to protect the outside world.

2. Networks should block incoming packets addressed to the broadcast address (the all-ones HOST_ID). There is no legitimate reason that an external network device should be sending a broadcast message to every host on your network.

3. To prevent your site from being used as a broadcast amplification point, turn off the Directed Broadcast capability at the router unless it is absolutely essential. If you think that it is essential, re-examine your network to see if there is not a better way. Even where Directed Broadcasts are useful, they are typically needed only *within* the enterprise and are not required for hosts on the outside.

4. RFC 1918 defines a set of private IP addresses that are not to be routed on the Internet. These addresses include:

| | | |
|---|---|---|
| 10.0.0.0/8 | 10.0.0.0-10.255.255.255 | One Class A address |
| 172.16.0.0/12 | 172.16.0.0-172.31.255.255 | 16 Class B addresses |
| 192.168.0.0/16 | 192.168.0.0-192.168.255.255 | 256 Class C addresses |

5. 
In addition, there are a number of reserved IP addresses that are never assigned to "public" networks, including:

| | |
|---|---|
| 0.0.0.0/32 | Historical broadcast address |
| 127.0.0.0/8 | Loopback |
| 169.254.0.0/16 | Link-local Networks |
| 192.0.2.0/24 | TEST-NET |
| 224.0.0.0/4 | Class D Multicast address range |
| 240.0.0.0/5 | Class E Experimental address range |
| 248.0.0.0/5 | Unallocated |
| 255.255.255.255/32 | Broadcast |

6. IP address spoofing is commonly employed by attackers and they commonly use one of the RFC 1918 private addresses or one of the other reserved addresses. Any packet that contains an RFC 1918 or reserved IP address in the IP Source Address or Destination Address field should be immediately discarded by the firewall and not ever sent to the Internet.

7. Block all unused application ports at the firewall, particularly such ports as IRC (6665-6669/tcp) and those known to be associated with DDoS tools.

8. Use some form of intrusion detection system (IDS) to protect your network. You might consider providing every system with "personal" firewall software to help detect an attack at individual systems; this is particularly potentially useful at sites (such as colleges) that have a large number of systems in *front* of a firewall (it is no coincidence that so many daemons reside on college and university computers that have been "owned").

9. Regularly monitor network activity so that aberrations in traffic flow can be quickly detected.

10. Educate your users about things to watch for on their systems and how to report any irregularity that might indicate that someone or something has tampered with their system. Educate your help desk and technical support to assist those users who make such reports. Have an intelligence gathering system within your organization so that such reports are centrally known so that trends can be spotted and responses devised.

11. Follow CERT, SANS, and TruSecure (ICSA) best practices procedures.

## ISP Actions

The Internet Service Providers (ISPs) offer the last hope in defeating the spread of a DDoS attack. While the ISP cannot take responsibility for locking down every customers' host systems, the ISPs have -- and should accept -- the responsibility to ensure that their network does not carry packets that contain obviously "bad" packets. Some of the steps that ISPs can take include:

1. As mentioned above, IP address spoofing is commonly employed by attackers using one of the RFC 1918 private addresses or one of the other reserved addresses. Amazingly, many ISPs will route these packets. Indeed, there is no entry in their routing table telling them where to send the packets; they merely forward them to a default upstream ISP. *Any* packet that contains any RFC 1918 or reserved IP address in the IP Source Address or Destination Address field should be immediately discarded.

2. Perform ingress (and egress) address filtering. *Ingress filtering* means that they should examine every incoming packet to their network from a customer's site and examine the IP Source Address field to be sure that the NET_ID matches the NET_ID assigned to that customer. This will require additional configuration at the router and may even result is slight performance degradation but the tradeoff is certainly well worth the effort. The ISPs should also perform egress filtering to check their outbound packets to upstream and peer ISPs.

3. Disable IP directed broadcasts.

4. Pay careful attention to high-profile systems (servers) and customers.

5. Educate customers about security and work with them to help protect themselves.

Most of the ISP community take at least some of these steps. Users should insist that their ISPs provide at least these protections and should not do business with those who don't. The TruSecure (formerly ICSA) ISP Security (ISPSec) community is a good source of information for ISPs.

## Other Tools Under Development or Consideration

Responses to DDoS attacks are not limited to the defensive steps listed above. Indeed, proactive responses to the prevention and detection of DDoS attacks is an active area of research.

One method that is being discussed is to examine the network at the ISP level and build a type of intelligent, distributed network traffic monitor; in some sense, this would be like an IDS for the Internet. The idea is that ISPs, peering points, and/or major host servers would have traffic monitor hardware watching over it. The monitor hardware would itself, naturally, use IP and the Internet for communications, much like today's routing protocols. each box would examine packets and their contents, doing some type of multilayer statistical analysis of traffic to learn the ordinary patterns. These devices would have enough intelligence, presumably, to be able to detect changes in traffic level and determine whether those changes reflected a normal condition or not. The hardware is defensive in nature at the receiving side. As an example, suppose that such hardware at Amazon.com thinks that there is a DoS attack being launched from an ISP in Gondwanaland. The traffic monitoring network would communicate and shut off traffic to Amazon coming from that ISP, as close to the ISP as possible. In this way, the distributed network of monitors can shut traffic off at the source.

The hardware, of course, needs to somehow be informed about traffic level changes that are due to "normal" events, such as a new Super Bowl commercial being posted at the Ad Critic Web site or a new fashion show at Victoria Secret's Web site. And the hardware also needs to prevent the attacker community from operating under the cover of these normal events.

RSA Laboratories, one of the leading vendors of cryptography in the world, has proposed another potential defense to DDoS attacks against Web servers that employs cryptographic methods. Their approach uses a *client puzzle* protocol designed to allow servers to accept connection requests from legitimate clients and block those from attackers. A "client puzzle" is a cryptographic problem that is generated in such a way as to be dependent upon time and information unique to the server and client request.

The scheme, simply, works roughly like this. Under normal conditions, the server accepts any connection request from any client. If an attack is detected, the server selectively accepts connection requests by responding to each request with a puzzle. The server allocates the resources necessary to support a connection only to those clients that respond correctly to the puzzle within some regular TCP timeout period. The idea is that a bona fide client will only experience a modest delay getting a connection during an attack, while the attacker will require an incredible amount of processing power to sustain the number of requests necessary for a noticeable interruption in service, quickly blocking the attack (in effect, a reverse DoS).

A third tool that is under consideration is that of *IP Traceback*. The problem with DoS/DDoS attacks is that packets come from a large number of sources and IP address spoofing masks those sources. Traceback, in concept, is a relatively straight-forward idea. Every packet on the Internet goes through some number of ISP routers. The processing power, memory, and storage are sufficiently available that routers could mark packets with partial path information as they arrive. Since DoS/DDoS attacks generally comprise a large number of packets, the traceback mechanism doesn't need to mark every packet, but only a sample size that statistically likely to include attack packets (e.g., one packet out of every 20,000). The feature allows the victim to locate the approximate source of the attack without the aid of outside agencies *and* even after the attack has ended. Another similar, yet different, traceback proposal would define an ICMP Traceback message that would be sent to the victim site containing partial route information about the sampled packet. While both of these proposals requires a change to tens of thousands of routers in the Internet, it is a solution that can be gradually implemented, is backward compatible, and results in no negative affect to users.

These three proposals are merely samples of some of the R&D for dealing with DDoS attacks; the first adds new hardware to the Internet, the second requires changing Web server and client software (actually, upgrading Web browsers is probably the *easiest* part to fix even though there are millions of copies in distribution; the vast majority come from two vendors and users tend to eventually upgrade), and the third requires incrementally changing software in all of the Internet's routers. And there are even more proposals -- such as ICSA's Host Identity Payload (HIP) protocol -- under way.

### Final Comments

One of the greatest short-comings in many organizations is that the highest levels of management do not truly understand the critical role that computers, networks, information, and the Internet play in the very life of the organization. Without using outright scare tactics, it is difficult to explain that there is an intruder community and that they are actively working on new tools all the time; and history has shown that as the tools mature and become more sophisticated, the technical knowledge necessary of the potential attacker goes down and the number of attacks overall goes up. Too many companies hide their heads in the sand and insist that "no one would bother us" without realizing that *any site* can become a target just by being there.

## Conclusion

If anything proves the intertwingled nature of the Internet, it is the defense against DDoS attacks (even more than the Web!). DDoS attacks require the (unintended) collusion of hundreds or thousands of computers to attack a few victims and defense against DDoS attacks requires the (intended) cooperation of tens of thousands of ISPs and customer networks. DDoS attacks will be with us for some time but there are ways today to minimize them -- but they require continued diligence at locking down all of the hosts connected to the Internet. As that is not likely to happen anytime soon, the disappearance of DDoS attacks is equally unlikely.

## References

- CERT/CC. "CERT Advisory CA-1999-17 Denial-of-Service Tools." 3 March 2000. URL: http://www.cert.org/advisories/CA-1999-17.html (21 November 2000).
- _____. "Results of the Distributed-Systems Intruder Tools Workshop." 2-4 November 1999. URL: http://www.cert.org/reports/dsit_workshop.pdf (21 November 2000).
- Dietrich, S., D. Dittrich, and N. Long. "An Analysis of the "Shaft" Distributed Denial of Service Tool." 13 March 2000. URL: http://www.sans.org/y2k/shaft.htm (21 November 2000).
- Dittrich, D. "The DoS Project's "trinoo" distributed denial of service attack tool." 21 October 1999. URL: http://staff.washington.edu/dittrich/misc/trinoo.analysis (21 November 2000).
- _____. "The "stacheldraht" distributed denial of service attack tool." 31 December 1999. URL: http://staff.washington.edu/dittrich/misc/stacheldraht.analysis (21 November 2000).
- _____. "The "Tribe Flood Network" distributed denial of service attack tool." 21 October 1999. URL: http://staff.washington.edu/dittrich/misc/tfn.analysis (21 November 2000).

- Garber, L. "Denial-of-Service Attacks Rip the Internet." IEEE Computer Magazine, April 2000.
- ICSA.net. "Security Best Practices for Internet Service Providers." 27 September 1999, version 6. URL: http://www.icsa.net/html/communities/downloads/Best_Practices_v6%20rev.rtf (21 November 2000).
- Internet Security Systems. "Trinity v3 Distributed Denial of Service tool." 5 September 2000. URL: http://xforce.iss.net/alerts/advise59.php (21 November 2000).
- Moskowitz, R. "Host Identity Payload." Internet Draft, February 2000. URL: