Distributed Denial of Service

Countermeasures from a System's Perspective

Jonna Bengtsson
jonbe675@student.liu.se

Magnus Falk
magfa938@student.liu.se

Linköping university
TDDC03 – Information Security
May 6, 2003

1 Background

Since the launch of the English-language version of the Al-Jazeera website March 25 it has been unavailable due to various denial of service (DoS) attacks [1]. The attacks have been both in the form of intentional distributed DoS (DDoS) attacks and a barrage of e-mails, a form of "unintentional" denial of service. There is even talk about the American government being a driving force behind the attacks. The power of DoS attacks becomes clearly evident when displayed like this and targets websites that rely heavily on staying available. In this report we will talk about what you as a systems administrator can do to in order to foil, or at least lighten the effects of, a DoS attack. The focus will be on DDoS attacks as when it comes to regular DoS attacks most often you just wait for a patch to come out.

1.1 Motivations

There are numerous of motives for a DoS/DDoS attack. The attacker can be a kid who is curious or wants to impress his/her friends. There can also be political reasons (as likely in the case of the attack on the Al-Jazeera web site) or a rival company trying to damage the image of the other company in order to steal clients. A disgruntled employee may be looking for revenge or the attack is unintentional when too many people are visiting a website at the same time and requesting services.

1.2 Denial of Service

A DoS attack is when a computer or network is in one way or another prevented from providing one or more of its services. This is, of course, very expensive if you are a large company and every minute of downtime means that thousands of customers are unable to buy your products. It also creates a lot of badwill when your website is unavailable and if the DoS continues people might be inclined leave for the, maybe better protected, competitor.

1.3 Definition of Distributed Denial of Service

Distributed denial of service is when a lot of hosts cooperate in order to deliver large volumes of synchronized DoS attacks. There can be several thousands of hosts involved in such an event and naturally they can do much more damage than a single host ever could.
DDoS has become increasingly popular as easy-to-use attack tools has been developed. The attack tools contain means to compromise otherwise innocent hosts and make them into "zombies" ready to DoS attack on commands. This provides the attacker with an efficient way of flooding a victim with enormous amounts of data, often forcing them offline.

1.4 Terminology

In this report we will use a few concepts that might not be familiar to the reader. Here we will explain some of the most common terms:

- **IP spoofing:** When the sending host labels his packets with an IP address different from the one it actually has, thus disguising the actual source of the packet.
- **Zombies:** Hosts that unwittingly has been infected by a DDoS attack tool and can be ordered by the attacker to initiate a DoS attack.
- **Master:** A kind of "middle man"-host that the actual attacker uses to distribute "orders" to the accumulated zombies. These too are actually innocent hosts that has been infected by the attacker. An attacker can have several masters to handle larger number of zombies.

2 Attacks

As denial of service attacks become more and more frequent both the means and ways of attacking become more numerous. There are basically two categories of attacks: Attacks that exploits a bug in the target system and others that simply gathers enough resources to overwhelm the victim. The former is pretty easily done as knowledge of bugs is spread very rapidly in the open environment of the web. Often it doesn't take very long until there are easy-to-use attack tools available that even the so-called "script kiddies" can use. Script kiddies are individuals, often young with little or no in-depth knowledge of security, who otherwise would not pose a significant threat but with the aforementioned tools can cause significant amounts of damage. We are here going to deal with the why, who, what and how of attacking a system. We are also going to talk about a way of measuring the frequency of attacks.

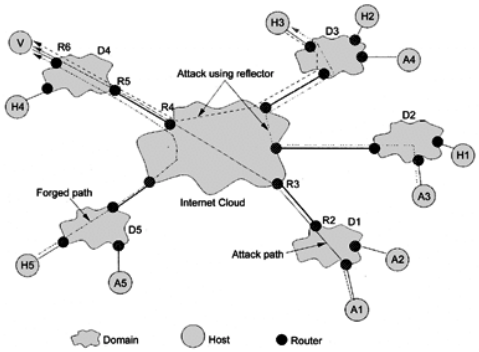

Figure 1. Different scenarios for DoS attacks. Attacker *A1* launches an attack on victim *V*. *A1* spoofs IP address of host *H5* from domain *D5*. Another attacker *A3* uses host *H3* to perform a reflector on attack on *V*. (picture taken from [13])

2.1 What Makes DoS Attacks Possible?

DoS attacks has become pretty famous since the attack that brought down Yahoo!, eBay and several other popular websites in February 2000 [19]. A recent study has shown how popular DoS attacks actually are [4]. With more than 12,000 attacks towards over 5,000 distinct target over a period of three weeks it's pretty safe to say that they are as popular as ever. At the same time the user-base of Internet has literally exploded which also means that there are a lot more people connected that do not know how to protect themselves properly. The attackers know and use this to their advantage by secretly installing software that takes control of the victim's computer and make them unwitting participants of an DDoS attack. This coupled with the relative ignorance on how to protect a system against an actual attack it isn't very surprising that DoS-attacks are very popular.

2.2 Who Are the Victims?

The backscatter study made by Moore et al [4] shows, that a significant part of all attacks are made against home machines, both dialup and broadband. Most of these are made against people running IRC-clients or people running multiplayer gaming services such as battle.net. Another observation is that a significant fraction (maybe 5-6%) of all attacks target network infrastructure such as name servers and routers. The latter is disturbing since disabling a router might deny service for a significant number of hosts.
Finally there is a rather big diversity of commercial targets. While big companies such as Amazon and Hotmail were expected there also was a significant portion of medium and small businesses targeted showing that pretty much anyone can be the victim of an attack.

2.3 What Different Kinds of Attacks Are There?

In this section we will describe different kind of attacks. Later in 2.4 we describe attack tools that relies on these attack techniques. There are two kinds of attack [11], flooding attacks and malformed packet attacks. They are differentiated by what means they use to compromise target systems. We will also mention network worms as a sort of DoS attack.

2.3.1 Flooding

Flooding relies on overwhelming the target or the network leading to the target system by sending too many packets for the target system to handle. The following flooding techniques differ in which kinds of packets they use for flooding and whether they use spoofing or not. The packets they use are SYN-packets, ICMP packets, UDP packets or TCP packets.

2.3.1.1 TCP SYN flooding

The course of action of a TCP SYN flooding attack as described in “Detecting Service Violations and DoS Attacks” [13] is as follows; An attacker sends a TCP SYN segment to a server to open a connection. The server allocates buffer for the connection and replies with a TCP ACK segment. The connection is now half - open. Normally a client would answer ACK here and open the connection. If the server doesn’t get the ACK the buffer space will be deallocated after a while. There is a limit in how many half-open connections a server can have at the same time. If there are too many half-open connections, the server refuses to accept all incoming requests. So, the attacker sends a lot of TCP SYN segments and no ACK, the server refuses all incoming requests (even from non-attacking clients) and the server is hence successfully compromised. The technique uses spoofing to hide the real source address.
Although the TCP SYN flooding technique works, the force of it is less than the power of more recent techniques. The attack can be easily prohibited by shortening the timeout time (the time the system waits until deallocating the buffer space and dropping half-open connections).
In a whitepaper written by WatchGuard Technologies Inc. [14] the SYN-flooding is said to be “more like heavy rainfall than flood” compared to other, more recent flooding techniques.

2.3.1.2 UDP ECHO REQUEST Flooding

UDP echo requests provides an alternative to ping when checking network connectivity over a VPN (Virtual Private Network). A UDP ECHO-REQUEST flooding actually effects two targets. The attacker sends a spoofed UDP packet to the echo service port on one of the two targets. The packet looks as if it is sent from the character generator service port of the other target machine. The machine that got the attack - packet replies to the other target that in turn replies to the reply it got. The two services keep on sending characters to each other (loops). This will make them consume a lot of their CPU resources and keep them from providing other services. More information about the character generator service can be found at Hewlett-Packard’s support page[18].

2.3.1.3 ICMP ECHO REQUEST flooding

An ICMP ECHO REQUEST is an ICMP packet used by the ping tool to determine if a remote system is reachable or not. If the remote system is reached it will answer by sending an ICMP ECHO REPLY - packet. The ICMP ECHO REQUEST flooding works in a similar way as the UDP ECHO REQUEST flooding except for that instead of UDP packets it uses ICMP echo request packets to flood the machines. The articles “Distributed Denial of Service Attacks” [15] and “Distributed Denial of Service Attacks: Threats, Motivations & Management” [16] deals with UDP- and ICMP ECHO-REQUEST flooding.

2.3.1.4 SMURF attack

SMURF is a classical *reflector attack* [9]. A reflector attack is when innocent, intermediary nodes (reflectors) are used in an indirect attack against a victim. In a SMURF attack the attacker sends a large amount of ICMP echo traffic to an IP broadcast address. The packets source address is the same as the victims address (i.e. a spoofed address). The servers that listens to the broadcast replies to the sender (the victim). There can be hundreds of replying servers on the net and the victim is flooded by their replies. Another attack called the Fraggle attack works the same way as the SMURF attack but uses UDP packets instead of ICMP.

2.3.2 Network Worms

A Network worm is a program that utilizes the net to spread itself from system to system. Network worms can be considered a type of DoS attack since network worms can cause network bandwidth saturation when they scan for systems to infect. We will describe two network worms that has been given a lot of attention in the press; Code Red and Nimda.

2.3.2.1 Code Red

Code Red began to infect hosts running unpatched versions of Microsoft's IIS Web server on July 12th, 2001. In less than 14 hours no less than 359,104 hosts were compromised [2]. The attack was unsuccessful in its purpose, to DoS - compromise “www1.whitehouse.gov”. Even though the attack failed in its purpose, it shows how fast systems all over the world can be compromised. There was also some unexpected collateral damage due to the worm; printers, routers, switches, DSL modems, and other web

devices crashed, rebooted or were otherwise damaged.

2.3.2.2 Nimda

The Nimda worm was discovered on September 18th, 2001 [3]. The worm was a “trendsetter” in two ways; it puts itself on existing web sites where it offered people to download the infected files and it used non-server machines when it scanned for vulnerable web sites. By doing so it was able to get behind Intranet firewalls (something Code Red for example was unable to). Nimda uses a security hole to infect IIS web servers. The worm is very complex and behaves differently depending on where it starts from (a server, a workstation, by email, by the hole in IIS or a web site).

2.3.3 Malformed Packets

The malformed packets technique is based on manipulation of packets in ways that for example makes the target get into trouble when trying to handle the packets or use forge source and destination addresses in the packets to make the system loop until it crashes.

2.3.3.1 Ping Of Death

Ping of Death uses ICMP ECHO request packets. A ping is sent that exceeds the allowed maximum ping data size. The effect is a crash or a reboot at the target [11].

2.3.3.2 WinNuke

Win Nuke connects to port 139 of any Windows 95 computer, and sends "junk" into the port. This causes an out of bound (OOB) problem in the victim machine and so the victim machine crashes [20].

2.3.3.3 TearDrop

TearDrop sends multiple fragments that cannot be properly reassembled by the victim. The target either halts or reboots [11].

2.4 Attack Tools

There are several versions of tools that facilitate DDoS attacks publicly available today. Some of the more common ones are called Trinoo, TFN, TFN2K and Stacheldraht and we’ll look them a bit closer below. There are also hybrids and modified versions of the most known tools and there probably exist a number of tools that are kept hidden by their creators so that other people can’t read the source code and figure out how the tools work. Tools can be created for either malicious or non-malicious reasons. A non-malicious reason is for example when a system administrator uses a tool to test the bandwidth capability and reliability of services under heavy traffic in a system. Common for all of them is that they all gather zombie machines first and then later use these to mount a large attack according to some specific scheme.

2.4.1 Trinoo

Trinoo is a simple and relatively benign tool [14]. It makes the zombies send floods of UDP packets to the victim. The tool doesn’t use spoofing so it’s easy to locate the source of the attack if only Trinoo is used.

2.4.2 TFN (Tribe Flood Network) and TFN2K

TFN and TFN2K were written in 1999 by a German hacker under the nickname “Mixter”[14]. Just like Trinoo, TFN uses UDP packet flooding but it also allows SYN- and ICMP flood and SMURF style attacks. Unlike Trinoo, it uses spoofing. TFN includes a backdoor that provides root access to the zombies host system. TFN2K is an updated version of TFN. In this version the communication link is encrypted. Instead of the master sending commands directly to the zombies, commands are sent to the networks that the zombies are in and the zombies sniffs the commands. No responses are sent back to the attacker so to be certain that the zombies get the instructions, each command is sent twenty times(!).

2.4.3 Stacheldraht

Stacheldraht is a hybrid of Trinoo and TFN [14]. It supports ICMP-, SYN- and UDP flooding and SMURF-style attacks. It also uses an encrypted link when sending commands between the attacker and the master. Stacheldraht has a characteristic feature-it can cause the zombies to download, install and execute an updated version of the master.

2.5 Measuring Techniques

The only documented technique for measuring the prevalence of DDoS-attacks as far as we know is called “Backscatter analysis” and is described by D. Moore in the article “Inferring

Internet Denial-of-Service Activity"[4]. The technique relies on the fact that when a system is attacked the attacker almost always uses IP-spoofing as a way of remaining unknown to the victim. When a system is attacked and unable to answer all requests the routers step in and send a ICMP-message back to the source address saying "host unreachable". As these addresses are randomly created the victim will send equi-probably distributed ICMP messages to the spoofed IP's. If a large enough address space is then monitored one can effectively "sample" all DDoS-activity on the Internet. This technique can also be used as a defensive mechanism on router level, but that is outside the scope of this report.

3 Defense

When it comes to defending yourself against denial of service attacks there are many suggested approaches. But so far none of them (maybe with the exception of filtering which has been there since the invention of firewalls) seems to really have caught on. A distributed denial of service attack is basically a war of resources where the victim either has to stop accepting new packets (in which case the DoS attack actually succeeded) or come up with some smart defense strategy as the victim most often is heavily outnumbered. Here we are going to talk about a few of the, in our view, most promising techniques and what makes them "tick".

3.1 How Much Can a System Handle?

When it comes to measuring how much a system can take R. K. C. Chang has made a few calculations in his article "Defending against Flooding-Based Distributed Denial-of-Service Attacks: A Tutorial" [9]. If we take as an example the SYN flooding attack and look at how many half-open TCP connections a normal server can handle. Figure 2 shows the results for three common server types: Microsoft Win2000 Advanced Server, BSD, and Linux kernel 2.2.9-19. All of them have similar retransmission strategies that resend SYN packets that seem lost. BSD retransmits after 6, 24 and 48 s and gives up after 75 s. Linux starts retransmitting at 3 s and then continue retransmitting (while each time doubling the delay) as many as 7 times for a total of 309 s before giving up. Windows on the other hand only tries 2 times at 3 and 6 s and gives up after 9 s. This explains the very good results for the Windows system in the example. If we translate this to an actual attack it means that an ordinary 56 kb/s connection is sufficient to stall a BSD or Linux system if the maximum number of allowed half-open TCP connections is 6000 or less (given a packet size of 84 bytes). Moreover, a 1 Mb/s connection can sink all three servers if the number is 10,000 or less.
When it comes to reflector attack the aim is simply to clog the victim's ling so much that traffic doesn't get through. Reflector attacks are ideally suited for this as each reflector will retransmit the SYN-ACK messages a number of times before giving up (for quite some time in the case of the Linux server). If you want to use a more basic attack, for example a ICMP ping flooding attack, you would need about 5000 zombies to successfully flood a T1 connection if each zombie sends a packet each second. The number naturally goes down if the frequency is increased.

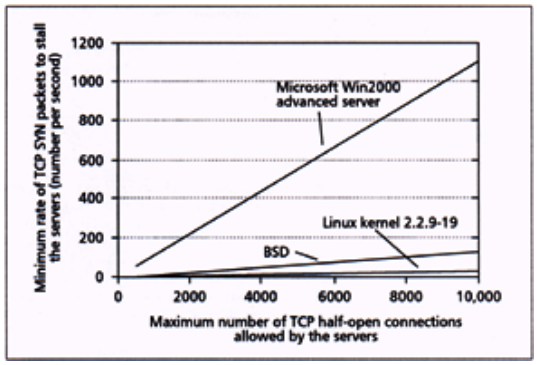

Figure 2. Minimal rates of SYN packets to stall TCP servers in SYN flooding attacks. (picture taken from [9])

3.2 Countermeasures

3.2.1 Honeypots

"The concepts were first introduced by several icons in computer security, specifically Cliff Stoll in the book The Cuckoo's Egg", and Bill Cheswick's paper " An Evening with Berferd." Since then, honeypots have continued to evolve, developing into the powerful security tools they are today." (Lance Spitzner, "Honeypots – Definitions and Value of Honeypots [10]).
A *honeypot* is a tool that deceives the attacker into believing that the system has been compromised into being a slave system that can be used in DDoS attacks. Honeypots are usually single systems that emulate real systems [7].

There are two types of honeypots: production- and research honeypots [10]. A production honeypot is used to protect a specific organization from attacks. A research honeypot uses the information gathered from the attacks to gain information about the "blackhat community" (computer hackers/crackers) in order to see what threats organizations in general face and how to protect them from those threats. A honeypot doesn't normally communicate much with the outside world so whenever a connection is made to or from the honeypot an attack or an unauthorized probing is likely taking place. Let's say you have three web servers and one of them is a honeypot. If the web servers are compromised by an attack, you can take the honeypot offline and conduct a full forensic analysis on it. You cannot do this on the normal web servers since you want them to be up and running as soon as possible after the attack. Those servers are hence usually just cleaned from specific holes and what you learn from the analysis of the honeypot you can later apply on the normal web servers. The honeypot not only records who is doing an attack, it also records how they did it and what they do after they have compromised a system. There are numerous honeypots on the market such as Specter and Mantrap and there are also OpenSource honeypots like Honeyd [10]. They all have different routines, advantages and disadvantages but we will not cover them in this report.

3.2.2 Honeynets

As far as we know, the origin of the concept of a honeynet is the Honeynet Project [7] but since a honeynet is a kind of Honeypot[7][10] we guess you could say that the idea originates from the same persons that thought of honeypots(see beginning of section 3.2.1).
Honeynets are networks of production systems, therefore, unlike honeypots, they don't just *emulate* a system - they really *are* systems. All the systems in the net protect themselves as they usually do so, unlike a honeypot, the honeynet doesn't trick the attacker into attacking by leaving known security holes open. Due to this, the analysis of the honeynet shows an accurate view of which threats the systems in the net are faced with. Honeynets are mainly used for research purposes.
When using honeypots and –nets one must always make sure that once the system is compromised it must not be able to be used in attacks against other systems. The hard part is to enable the attackers to execute whatever they want so that they don't suspect something is the matter and still maintain the demand on not being used in attacks on other, non-honeypot systems. The recording of the attacker actions must not be visible to the attacker and the data cannot be locally stored on the honeypot. If the data is locally stored, the attacker can see it or the data can be lost or destroyed in other ways.

3.2.3 Filtering

Filtering is an old and proven technique that has existed since the introduction of firewalls. A good thing about it is that it doesn't require much modification of the existing system. The general problem with filtering near the victim network though is that a lot of legitimate packets are also dropped. Thereby you've actually inadvertedly helped the attacker, as this kind of denial of service is "as good" as any. Another problem with most filtering technologies is that they require protocol changes and filtering policies to be installed in Internet backbone routers. As that is out of reach for the common systems administrator it will not be covered in this report. We will mention one promising filtering technique though, proposed by Farguson and Seine [5], called ingress filtering. What it does is filter all packets that does not match a domain prefix connected to the ingress router. So if an attacker tries to send a packet with a spoofed IP the ingress router will recognize it as a packet not belonging to that network and drop it immediately. This would at least ensure that DDoS attacks are not initiated from within the own network. This technique can very efficiently reduce DDoS attacks that employ IP-spoofing if it's installed in *all* domains. This is naturally the big drawback of this method as if there remains even a few unchecked entry points, these are the ones the attacks will originate from.
Even though there really aren't any simple filtering techniques that both stop DDoS attacks and at the same time let legitimate packets through there is much to be said for having a soundly configured firewall in the first place. Having a good security policy installed that for instance only allows connections to port 80 (http) goes a long way towards preventing DDoS attacks.

3.2.4 Extrusion detection

If we really want to stop DoS attacks, maybe we should start in the other end. D. Bruschi and E. Rosti [6] proposes a kind of *extrusion detection*.

Extrusion detection is when the host monitors itself in order to discover if it's made to participate in a DoS attack. The goal is to "disarm" computers and they define it as follows:
A disarmed host is a host equipped with tools that turn off the host attacking capabilities and that force the host to be re-installed for it to be subverted.
This is to be achieved through tools that will monitor the host and block it when its behavior doesn't match to the set "good behavior" policy. Or maybe the other way around; when the behavior falls into the "suspicious" category. The choice of policy will depend on what threats that are considered a priority. If shown to work this approach could for instance solve liability issues as a disarmed computer then could be considered legally safe and the owner automatically relieved of liability.
The main perk with this approach, if it's widely implemented, is that DDoS tools will have a really hard time finding hosts to compromise since all disarmed hosts will be worthless as zombies. Another advantage with it is that if "good" packet flows from "safe" hosts can be identified routers and firewalls can spend more time examining "suspicious" flows.
The problems with this approach are mainly deployment but freedom from liability would probably serve as a good incentive. Another problem is that advanced users probably wouldn't have any problems circumventing these mechanisms. But then again these are probably individuals who know what they're doing and also know how to protect themselves. The main point is that most of the aforementioned "script kiddies" would probably be foiled by this approach.

3.2.5 Moving firewall

NTT (Nippon Telegraph and Telephone Corporation) Information Sharing Platform Laboratories have recently developed a system called Moving Firewall (MovingFW)[17]. A prototype has been built and the MovingFW concept was proven to be effective so the future research consists of testing the MovingFW in larger and larger scales. The technique uses other known techniques such as IP- traceback and the main idea is to have several MovingFW devices cooperating to prevent an attack. What separates Moving Firewalls from other DDoS countermeasures is that they don't try to prevent an attack on their own and from a fixed position. Ordinary firewalls can't prevent over-consumption of network bandwidth and therefore they can't protect the user from large-scale DDoS attacks in an effective way. MovingFW's can upgrade themselves automatically to defend against new types of attacks and administrators can configure the device to use their security policies when detecting the attacks. This should enable good packet streams to keep on flowing while stopping malicious streams. The basic idea of a Moving Firewall is as follows: when a client in an ISP network is under attack, a MovingFW device nearby detects the attack. The device automatically launches its defense mechanism and dispatches its defense program code (which includes the attack signatures) hop by hop to MovingFW devices in the upstream of the attack flood. Eventually the code reaches nodes at the uppermost stream (which should mean they are close to the attacker). Since it is such a recent invention, not much literature can be found on the subject. The few articles we found while writing this paper were, sadly enough, very vague when explaining how MovingFW works in detail or in practice. All our information about Moving Firewalls comes from NTT and the information is therefore probably optimistically biased.

4 Conclusions

A big problem with many DDoS countermeasures is that they are rather expensive to implement. In our experience, the way most companies handle the DDoS threat is simply to get more redundancy in the form of additional load-balancers and more bandwidth. None of the described methods contain means to put a definite end to the DDoS threat but a combination of them might. Most of them suffer from deployment problems as in the case of extrusion detection and ingress filters. In order for them to be efficient they pretty much need to be deployed all over the Internet. Honeypots and –nets provide a useful tool for research and in time they might produce the means to end DDoS attacks. But in order to get any real changes we think that the answer probably lies in changing the Internet protocols. There simply isn't that much you can do when a vastly superior enemy decides to pound you into submission if it's all in accordance with the rules. There is a lot to be said for just patching and diligence though. Diligent reading of security mailing lists and system logs helps you stay ahead of the script kiddies and keeping your system safe. Another

good rule of thumb is to not have *any* other services than those you absolutely need running. And those you absolutely need you should know down to the nuts and bolts.

5 References

[1] The Cursor's Al-Jazeera Link. http://www.cursor.org/aljazeera.htm. 2003.

[2] Cooperative Association for Internet Data Analysis (CAIDA): The Spread of the Code-Red Worm (CRv2). http://www.caida.org/analysis/security/code-red/coderedv2_analysis.xml#conclusions. 2003.

[3] F-Secure Virus descriptions - Nimda: http://www.europe.f-secure.com/v-descs/nimda.shtml. 2003.

[4] D. Moore: Inferring Internet Denial-of-Service Activity. http://www.caida.org/outreach/papers/backscatter/usenixsecurity01.ps.gz. 2001.

[5] P. Ferguson and D. Seine: Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing agreements performance monitoring. RFC 2827: http://www.ietf.org/rfc/rfc2827.txt. May 2000.

[6] D. Bruschi and E. Rosti: Disarming offense to facilitate defense. http://homes.dsi.unimi.it/~rose/nspw.ps. 2000.

[7] Honeynet Project: Know Your Enemy: Honeynets – What a Honeynet is, its value, how it works, and risk/issues involved. http://project.honeynet.org/papers/honeynet/. January 2003.

[8] J. Kong, M. Mirza, J. Shu, C. Yoedhana, M. Gerla and S. Lu: Random Flow Network Modeling and Simulations for DDoS Attack Mitigation. http://www.cs.ucla.edu/~jkong/publications/ICC03-jkong.ps.gz. 2003.

[9] R. K. C. Chang: Defending against Flooding-Based Distributed Denial-of-Service Attacks: A Tutorial. IEEE Communications Magazine October 2002.

[10] L. Spitzner: Honeypots – Definitions and Value of Honeypots. http://www.tracking-hackers.com/papers/honeypots.html. May 17, 2002.

[11] A. Brindley: Denial of Service Attacks and the Emergence of "Intrusion Prevention Systems". http://www.sans.org/rr/firewall/prevention.php. November 1, 2002.

[12] F. Kargl, J. Maier and M. Weber: Protecting Web Servers from Distributed Denial of Service Attacks. http://www.www10.org/cdrom/papers/pdf/p409.pdf. May 2001.

[13] A. Habib, M. M. Hefeeda and B. K. Bhargava: Detecting Service Violations and DoS Attacks. http://raidlab.cs.purdue.edu/papers/dos.pdf.

[14] WatchGuard Technologies, Inc.: Distributed Denial of Service. http://ssaintalme.free.fr/ddos_wg.pdf. February 2000.

[15] F. Lau, S. H. Rubin, M. H. Smith and L. Trajkovic: Distributed Denial of Service Attacks. http://www.ensc.sfu.ca/~ljilja/./papers/smc00.pdf. 2000.

[16] T. Simcock: Distributed Denial of Service Attacks: Threats, Motivations & Management. http://www.giac.org/practical/GSEC/Tom_Simcock_GSEC.pdf. November 5, 2002.

[17] NTT Information Sharing Laboratory Group: Development of Moving Firewall, a System that Mitigates DDoS Attacks at Upstream and Defends the Entire Network. www.ntt.co.jp/news/news03e/0302/030218.html. (EXPIRED ARTICLE) February 18, 2003.

[18] Hewlett-Packard Company Support Pages. http://forums.itrc.hp.com/cm/QuestionAnswer/1,,0x7892e3ed7640d71190080090279cd0f9,00.html. February 18, 2003.

[19] M.Williams, IDG News Service: Ebay, amazon, buy.com hit by attacks. http://www.nwfusion.com/news/2000/0209attack.html. February 9, 2000.

[20] D. Finkelstein: The WinNuke Relief Page. http://www.users.nac.net/splat/winnuke/. July 12, 1997.