Reading Notes: (D)DoS Attacks

Panos Papadimitratos
Networked Systems Security Group
www.ee.kth.se/nss

November 9, 2015

Contents

1 Introduction (Slides 1-2)

Denial of Service (DoS) attacks and Distributed Denial of Service (DDoS) Attacks are hard to defeat and commonly used against networked systems. Recall the introductory lecture, the Mafia Boy, and the story of extortion from the New Yorker. Beyond those, many incidents, notably of DDoS attacks, have been observed these past years. The targets of such attacks can be international companies,[1] national authorities, [2,3] or even countries[4,5].

2 DoS Attacks Overview (Slide 3 - 5)

DoS attacks, distributed or not, have the same purpose, to harm system availability. They differ in how they are launched. A DoS is launched essentially by one attacking machine while a Distributed DoS is launched by multiple attacking machines at the same time. Let's consider first DoS attacks and why they are possible, even easy to mount.

- Security agnostic design: Early Internet protocols were designed with functionality and efficiency in mind, not security. The same is true for other networking technologies in their first stages of development; for example, Mobile Ad Hoc Networks (MANETs) and notably routing protocols were initially proposed circa 2000 assuming a benign, collaborative environment.

- Design Problems: In various systems and for many technologies, design aspects can be used by an attacker; especially, if there is no clear understanding of vulnerabilities and what security means.

- System defects or *Bugs*: In many cases, attackers take advantage of technical problems, e.g., because of fast development and roll-out of new products, or complexity of the software, they craft specific attacks (often called in literature attack vectors).

Various types of DoS attacks can take place in all layers of communication networks. For example, Jamming attacks are a type of DoS attacks launched at the *Physical Layer*, the lowest layer of the ISO/OSI reference Model (slide 4).

In all cases, a successfully mounted DoS attack degrades performance or even denies service, i.e., prevents access to resources or brings down the system.

2.1 802.11b De-authentication (Slides 6-8)

The 802.11b version of the 802.11 family of standards for Wireless Local Area Networks (LANs) [6] makes use of *Management Frames* for managing the wireless

[1] http://www.computerworld.com/s/article/9200521/Update_MasterCard_Visa_others_hit_by_DDoS_attacks_over_WikiLeaks

[2] http://www.sydsvenskan.se/sverige/article1321200/Aklagarmyndighetens-hemsida-nere.html

[3] http://nakedsecurity.sophos.com/2011/06/15/cia-website-down-hackers-lulzsec/

[4] http://arstechnica.com/security/news/2007/05/massive-ddos-attacks-target-estonia-russia-accused.ars

[5] http://ddos.arbornetworks.com/2008/08/georgia-ddos-attacks-a-quick-summary-of-observations/

[6] http://standards.ieee.org/about/get/802/802.11.html

access. One such frame is the de-authentication frame used by wireless devices to terminate their connection to the Access Point (AP). As such frames are not authenticated, an attacker can send de-authentication frames on behalf of another mobile device. This will result in the targeted device being disconnected from the wireless network (the AP). Such an attack can render the wireless network unavailable for a specific mobile host.[7].

- This is a DoS attack easy to launch. An attacker can disconnect a wireless device with a single frame.
- It is an example of bad design: the 802.11b protocol did not include authentication of Management Frames.

Solution: IEEE 802.11w - see lecture on wireless security.

2.2 The Apache Killer Bug (Slide 9)

An example of DoS attack that makes use of system bugs is presented here. An attacker can attack earlier versions (e.g., 2.2 or 1.3) of the *Apache Server*, with multiple *HTTP* requests with overlapping byte ranges that can crash the web server. This DoS attack exploited a software bug that resulted in an unexpected behavior of the server [8].

The essence of the attack: HTTP offers the ability to optimize bandwidth, requesting a range, a subset of the bytes of a large file. Back in 2007, it was first realized that several requests with overlapping ranges could consume a lot of memory on the server side and make it send large amounts of data. Then, automated tools to do this against any server (notably, un-patched) became available. Then, few years later, advisories were issued, and the problem was fixed (a patch released).[9]

3 SYN-Flooding (Slides 10-12)

Next, we examine is SYN-Flooding[10]. This attack takes advantage of the TCP transport protocol and more specifically the 3-way Handshake used to establish a TCP connection between the two communicating entities[11].

The *TCP Handshake* is an exchange of three messages:

- **SYN Message** Client to server message: request for synchronization. Upon receipt, the server consumes resources to keep track of all the required fields (sequence number, ports, addresses, etc.) of the TCP connection that has just been initiated.
- **SYN-ACK Message** Server to client message: response to the *SYN* message. Sequence number stored by the server are sent to the client via this message.

[7] http://sysnet.ucsd.edu/~bellardo/pubs/usenix-sec03-80211dos-html/aio.html

[8] http://www.theregister.co.uk/2011/08/24/devastating_apache_vuln/

[9] http://seclists.org/fulldisclosure/2011/Aug/301

[10] http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_9-4/syn_flooding_attacks.html

[11] http://tools.ietf.org/html/rfc793

- **ACK Message** Client to server message: acknowledgement of the *SYN-ACK* message. At this point, the connection between the client and the server is considered established.

As the server has to keep state for half-open TCP connections, it consumes resources. The attacker can try to initiate as many half-open TCP connections in an effort to exhaust the resources of the targeted server. Additionally, given the attacker does not intent to establish an actual TCP session, the IP address of the IP header can be/is spoofed (not the attacker's actual IP address).

A reading reference you may find useful: `http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_9-4/syn_flooding_attacks.html`.

3.1 Protecting against SYN-Flooding: The SCTP Protocol (Slides 13-14)

The problem with TCP flooding can be partially mitigated by using the newer Transport Protocol: SCTP makes use of a 4-way handshake protocol [12]. The difference between the two handshake phases is that in SCTP, the server sends a cookie to the client before it consumes any resources for the state of the initiated connection. The client has to replay the cookie as proof that its IP is a real one (not spoofed). Once the client sends the $COOKIE-ECHO$ message, the connection is be established.

More precisely, the server gets information from the $INIT$ packet ("chunk" in SCTP parlance) and its $INITACK$ response. These define the connection to be established. In addition, the cookie calculation inputs include the current time (a timestamp) and the life span of the cookie (note: it is called the "State Cookie"). The two end points (client and server) are assumed to have a shared symmetric key. This is used in the calculation of the cookie, that is, in the authentication of the cookie. A Message Authentication Code (MAC) is calculated over the entire cookie.

Once the cookie is generated and included in the $INITACK$ chunk, the server should erase from its memory - what the protocol calls its "Transmission Control Block (TCB)" - all the information included into the cookie, it *erases* the cookie. Then, when the $COOKIE-ECHO$ message is sent back to the server, it is authenticated, checked if within its validity period, not previously seen, and a new connection is established (with the state kept/restablished actually on the side of the server).

For more details on the SCTP, please see the Internet Engineering Task Force (IETF) Request for Comments (RFC): `http://www.ietf.org/rfc/rfc2960.txt`

Cookies have been used (in a protocol compliant manner) for TCP, see for example `http://cr.yp.to/syncookies.html`. Then, a more recent effort seek to address those limitations, notably an experimental RFC: `https://tools.ietf.org/html/rfc6013`.

The article by Metzger, Simpson, and Vixie, on "Improving TCP security with robust cookies" (`http://static.usenix.org/publications/login/2009-12/openpdfs/metzger.pdf`) can be interesting to probe further - it is better to revisit it at a later point, e.g., when we cover the DNS security.

[12] `http://datatag.web.cern.ch/datatag/WP3/sctp/primer.htm`

Question to ponder, as we continue with the upcoming lectures: Where would you implement DDoS defenses, on the server itself, on another machine (which one?), or both?

3.2 Protecting against SYN-Flooding: Proxies (Slide 15)

In order to protect a server from being overwhelmed by TCP requests, network administrators can use strong proxies that filter attack traffic from legitimate traffic. In the case of SYN-Flooding traffic, the proxy will forward to the client only the TCP-Connections that have been established (handshake has been completed).

3.3 Protecting against SYN-Flooding: Cryptographic Puzzles (Slide 16)

Another approach against SYN-Flooding attacks is the use of cryptographic puzzles.[13] The idea behind this countermeasure is that a client requesting to establish a TCP connection with a server will have to consume computational resources in order to solve a cryptographic puzzle. There are two reasons based on which the use of cryptographic puzzles can work as a solution against SYN-Flooding (or any kind of DDoS attacks). When a client sends the solution of the cryptographic puzzle, the server can be sure that the IP address of the client is not spoofed. In addition, these puzzles require the clients to consume computational resources in order to solve the puzzle. This makes it harder for the attacker (e.g., a BotNet, see below) to overwhelm a server with TCP connections because it would have to consume a significant amount of resources.

4 Smurf Attacks (Slides 17-18)

During a Smurf attack[14], the attacker sends tailored ICMP messages to the broadcast addresses of a network. The source IP of the packet is the IP of the desired victim that they want to flood. As a result, all the hosts will receive this message and will issue and ICMP reply to the victim and consequently flood it. This version of DDoS attack makes use of a sophisticated technique in order to amplify its effectiveness. Networks that allow this kind of attacks are known as “amplifier networks”. To prevent this type of attacks, filtering of outgoing packets that contain a source address from a different network[15] can be used.

5 Distributed Denial of Service Attacks Overview and BotNet Recruitment (Slides 25-26)

Unlike DoS attacks, the DDoS attacks rely on the collective resources of large groups of computers - these came to be known as *BotNets*. In many cases malicious code (Viruses, Trojans, back-doors) are used to “recruit” vulnerable machines to BotNets.

[13] `http://www.ece.vt.edu/parkjm/Research/techReport_pTCP.pdf`

[14] `http://www.cert.org/advisories/CA-1998-01.html`

[15] `http://www.cert.org/advisories/CA-1998-01.html`

5.1 DDoS Attacks (Slides 19-23)

A brief description of DDoS incidents.

5.2 Basic BotNet Architectures (Slides 24-29)

The main components in a BotNet architecture are:

- **Bot-Master**: The agent that controls the BotNet. Needs to be online only briefly.
- **Handlers**: The control over the BotNet is done through Handlers. The Bot-Master communicates with the Handlers in order to command and control the BotNet.
- **Stepping Stones**: In order to prevent the tracking of the Bot-Master (once a DDoS is launched), an additional level of hierarchy is included. Stepping Stones serve as proxies between the Bot Masters and the Handlers. Command traffic takes place between the Bot-Master and the Handlers (through the Stepping Stones) whereas attack traffic is sent from the Bots to the targeted server.
- **Zombie machines**: The machines that have been infected with some sort of malicious code and have been "recruited" to the BotNet.

5.3 IRC controlled BotNets (Slides 30-31)

The architecture presented is known as *Direct Communication*. This kind of communication scheme can be problematic (for attacker) as it can allow tracing the communication between the Bot-Master and the Handlers. For this reason, more efficient and obscure architectures are used by attackers in an effort to hide their traces. These schemes fall into the category of *Indirect Communication* and usually rely on IRC (Internet Relay Chat, IETF RFC 1459) communication channels.

Each IRC server is responsible for a subset of the malicious network. Slide 24 illustrates the idea of indirect communication with the use of IRC servers. The main advantage of this mode of control is that, ownership relations between owners and zombie machines where hidden within the thousands of communication channels that existed on the IRC servers.

5.4 BotNet Recruitment (Slides 32-34)

In this part of the lecture we discuss on different ways of recruiting vulnerable machines to a BotNet. Recruiting a vulnerable machine requires two steps:

- **Detection**: The process of identifying a vulnerable machine.
- **Recruitment**: Once the vulnerable machine is detected, the Bot-Master can use various techniques that take advantage of the vulnerabilities identified.

The Recruitment phase includes a variety of methods and tools. Different techniques that can be used to detect a vulnerable machine. The most common ones are:

- **Random Scanning**: The attacker randomly selects an IP address from the address space. If this IP is in use, then the corresponding machine is examined for vulnerabilities. In case vulnerabilities are discovered, the attacker tries to exploit them and get control of the machine. If successful, a piece of malicious code is installed at the targeted machine. This piece of code will allow the attacker to take control of the machine on-demand. It is noteworthy that random scanning needs not necessarily to be performed by the attacker. Any zombie computer already part of the DDoS network can perform it. This is feasible due to the automated techniques for vulnerability scanning that are broadly available. In this case, a list with the addresses of vulnerable machines will be sent to the attacker so that the manual process of exploitation takes place later.

- **Hit-List Scanning**: This technique includes a list with the addresses of potentially vulnerable machines. Once a machine from the list is found to be vulnerable, the attacker or a machine within the network that the attacker controls installs the piece of malicious code that in fact recruits the machine into the network. In addition to that, the hit list is split into half. For the first half of the list, the machine that recruited the new member of the network is in charge. The second half is given to the newly recruited machine that will work on it in the same manner.

- **Topological Scanning**: This scanning technique is equally effective compared to Hit List scanning in terms of speed of propagation. The main idea behind it is that as long as a machine is recruited into the DDoS network, its hard disk is scanned for addresses (IP addresses or domain names) of other machines. The result of this process defines what the next targets will be. The success of this approach is based on the usually correct assumption that since information about other machines exist on the recruited machines; some kind of trust relations might have been established between them. These trust relations might turn out to make it easier to break into the new targets.

- **Local-subnet Scanning**: In case a machine is compromised, it can act as a Trojan-Horse and perform vulnerability scanning within its the internal network it belongs. An interesting fact in regards to this approach is that it allows an attacker to recruit hosts that have private IP addresses and might be behind a firewall or a NAT. As a result, this technique permits the increase of the pool of potential victims as it can targets machines within a sub-net.

Again the scanning traffic can be either in the form of Direct Scanning (in this case Handlers scan for vulnerable machines) or Indirect Scanning (IRC communication channels are used).

6 Distributed TCP Flooding (Slide 35)

The SYN flooding attack described in the previous slides can be used in a distributed manner. A large number of bot agents commanded by some Bot-Master will try to establish half-open TCP connections in an effort to exhaust the resources of the targeted server. Of course, it is clear that the Distributed version of TCP flooding is much more effective.

7 P2P Attacks (Slide 36-37)

The attack described in these slides targets file-sharing communities.[16] In such attacks, the attacker does not need to use any BotNet to attack a server. Her main goal is to convince the P2P community that the desired content is available at a specific address. When this happens, and based on the popularity of the content, the peers might aggressively try to fetch the content from the server and as a result flood it.

8 Operation Payback (Slide 38-40)

A new type of DDoS attacks emerged during the incidents of Wikileaks on 2010. The hacktivist group known as *Anonymous* launched a series of DDoS attacks against companies like VISA, MasterCard and PayPal among others. The interesting point about these DDoS attacks was that it included no BotNet. Supporters of Wikileaks joined voluntarily the efforts of this group. With the use of simple tools [17] and social networking sites like Tweeter, they managed to overwhelm the servers of the targeted companies and disrupt their services.

9 Research on DDoS Countermeasures (Slide 41-44)

One of the main approach when it comes to detecting and discarding malicious DDoS traffic is the use of network based mechanisms.Examples of such mechanisms are the various "IP Trace back" mechanisms. In this category of mechanisms, network routers apply their individual marking on packets as they forward them to the network. Different approaches exist in regards to packet marking mechanisms.

Deterministic packet marking adds to all packets with the same marking given the fact that they follow the same path. This marking is the concatenation of all the individual markings of the routers that have forwarded the packet. Probabilistic packet marking, on the other hand, introduces probabilities concerning the markings that the packets will receive. A network router marks packets based on a predefined probability. No matter their nature, the common idea behind both techniques is that the victim, based on the markings of the packets, will be able to create filters capable of blocking malicious traffic. By maintaining a pool of markings that have been black-listed due to increased malicious traffic, filters can efficiently discard malicious traffic.

[16] http://delivery.acm.org/10.1145/1150000/1146894/a47-naoumov.pdf?ip=130.229.175.66&acc=ACTIVE\%20SERVICE&CFID=68695286&CFTOKEN=84237470&__acm__=1330688466_1ff4fee4e0984a9b6d8f6be0892f9099

[17] http://en.wikipedia.org/wiki/Low_Orbit_Ion_Cannon