Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Networks 44 (2004) 643–666
COMPUTER NETWORKS
www.elsevier.com/locate/comnet

DDoS attacks and defense mechanisms: classification and state-of-the-art

Christos Douligeris *, Aikaterini Mitrokotsa

Department of Informatics, University of Piraeus, 80 Karaoli and Dimitriou Str, Piraeus 18534, Greece

Received 9 October 2003; accepted 13 October 2003

Responsible Editor: I.F. Akyildiz

Abstract

Denial of Service (DoS) attacks constitute one of the major threats and among the hardest security problems in today's Internet. Of particular concern are Distributed Denial of Service (DDoS) attacks, whose impact can be proportionally severe. With little or no advance warning, a DDoS attack can easily exhaust the computing and communication resources of its victim within a short period of time. Because of the seriousness of the problem many defense mechanisms have been proposed to combat these attacks. This paper presents a structural approach to the DDoS problem by developing a classification of DDoS attacks and DDoS defense mechanisms. Furthermore, important features of each attack and defense system category are described and advantages and disadvantages of each proposed scheme are outlined. The goal of the paper is to place some order into the existing attack and defense mechanisms, so that a better understanding of DDoS attacks can be achieved and subsequently more efficient and effective algorithms, techniques and procedures to combat these attacks may be developed.

Keywords: DoS attacks; DDoS attacks; Defenses; Network security; Intrusion detection

1. Introduction

Denial of Service (DoS) attacks are undoubtedly a very serious problem in the Internet, whose impact has been well demonstrated in the computer network literature. The main aim of a DoS is the disruption of services by attempting to limit access to a machine or service instead of subverting the service itself. This kind of attack aims at rendering a network incapable of providing normal service by targeting either the network's bandwidth or its connectivity. These attacks achieve their goal by sending at a victim a stream of packets that swamps his network or processing capacity denying access to his regular clients. In the not so distant past, there have been some large-scale attacks targeting high profile Internet sites [1–3].

Distributed Denial of Service (DDoS), is a relatively simple, yet very powerful technique to attack Internet resources. DDoS attacks add the many-to-one dimension to the DoS problem making the prevention and mitigation of such attacks more difficult and the impact proportionally severe. DDoS exploits the inherent weakness of the

* Corresponding author. Tel.: +30-1-4142137.
E-mail addresses: cdoulig@unipi.gr (C. Douligeris), mitrokat@unipi.gr (A. Mitrokotsa).

1389-1286/$ - see front matter
doi:10.1016/j.comnet.2003.10.003

Internet system architecture, its open resource access model, which ironically, also happens to be its greatest advantage.

DDoS attacks are comprised of packet streams from disparate sources. These attacks engage the power of a vast number of coordinated Internet hosts to consume some critical resource at the target and deny the service to legitimate clients. The traffic is usually so aggregated that it is difficult to distinguish legitimate packets from attack packets. More importantly, the attack volume can be larger than the system can handle. Unless special care is taken, a DDoS victim can suffer from damages ranging from system shutdown and file corruption, to total or partial loss of services.

There are no apparent characteristics of DDoS streams that could be directly and wholesalely used for their detection and filtering. The attacks achieve their desired effect by the sheer volume of attack packets, and can afford to vary all packet fields to avoid characterization and tracing.

Extremely sophisticated, "user-friendly" and powerful DDoS toolkits are available to potential attackers increasing the danger of becoming a victim in a DoS or a DDoS attack. DDoS attacking programs have very simple logic structures and small memory sizes making them relatively easy to implement and hide. Attackers constantly modify their tools to bypass security systems developed by system managers and researchers, who are in a constant alert to modify their approaches to handle new attacks.

The DDoS field is evolving quickly, thus becoming increasingly hard to grasp a global view of the problem. Although there is no panacea for all flavors of DDoS, there are several countermeasures that focus on either making the attack more difficult or on making the attacker accountable.

In this paper, we try to introduce some structure to the DDoS field by presenting the state-of-the-art in the field through a classification of DDoS attacks and a classification of the defense mechanisms that can be used to combat these attacks. The classification of attacks includes both known and potential attack mechanisms. In each attack category we define special and important features and characteristics. We also classify published approaches of defense mechanisms and even though we point out vulnerabilities of certain defense systems, our purpose is not criticize the defense mechanisms but to describe the existing problems so that they might be solved.

Following this introduction, the paper is organized as follows. Section 2 investigates first the problem of DoS attacks and then presents a classification of DoS attacks. Section 3 introduces the problem of DDoS attacks, gives the basic characteristics of well-known DDoS tools and presents a taxonomy of DDoS attacks. Section 4 presents the DDoS defense problems and proposes a classification of DDoS defense mechanisms, while Section 6 concludes the paper.

2. DoS attacks

2.1. Defining DoS attacks

According to the WWW Security FAQ [4] a DoS attack can be described as an attack designed to render a computer or network incapable of providing normal services. A DoS attack is considered to take place only when access to a computer or network resource is intentionally blocked or degraded as a result of malicious action taken by another user. These attacks don't necessarily damage data directly or permanently, but they intentionally compromise the availability of the resources.

The most common DoS attacks target the computer network's bandwidth or connectivity. Bandwidth attacks flood the network with such a high volume of traffic that all available network resources are consumed and legitimate user requests cannot get through, resulting in degraded productivity. Connectivity attacks flood a computer with such a high volume of connection requests, that all available operating system resources are consumed, and the computer can no longer process legitimate user requests.

2.2. DoS attack classification

DoS attacks can be classified into five categories based on the attacked protocol level, as illustrated in Fig. 1 [5].

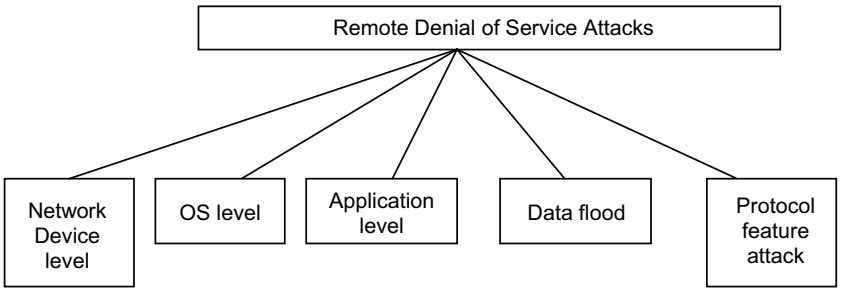

Fig. 1. Classification of Remote Denial of Service attacks.

DoS attacks in the *Network Device Level* include attacks that might be caused either by taking advantage of bugs or weaknesses in software, or by trying to exhaust the hardware resources of network devices. One example of a network device exploit is the one that is caused by a buffer overrun error in the password checking routine. Using these exploits certain Cisco 7xx routers [6] could be crashed by connecting to the routers via telnet and entering extremely long passwords.

In the *OS level* DoS attacks take advantage of the ways operating systems implement protocols. One example of this category of DoS attacks is the Ping of Death attack [7]. In this attack, ICMP echo requests having total data sizes greater than the maximum IP standard size are sent to the targeted victim. This attack often has the effect of crashing the victim's machine.

Application-based attacks try to settle a machine or a service out of order either by taking advantage of specific bugs in network applications that are running on the target host or by using such applications to drain the resources of their victim. It is also possible that the attacker may have found points of high algorithmic complexity and exploits them in order to consume all available resources on a remote host. One example of an application-based attack is the finger bomb [8]. A malicious user could cause the finger routine to be recursively executed on the hostname, potentially exhausting the resources of the host.

In *data flooding attacks*, an attacker attempts to use the bandwidth available to a network, host or device at its greatest extent, by sending massive quantities of data and so causing it to process extremely large amounts of data. An attacker could attempt to use up the available bandwidth of a network by simply bombarding the targeted victim with normal, but meaningless packets with spoofed source addresses. An example is flood pinging. Simple flooding is commonly seen in the form of DDoS attacks, which will be discussed later.

DoS *attacks based on protocol features* take advantage of certain standard protocol features. For example several attacks exploit the fact that IP source addresses can be spoofed. Several types of DoS attacks have focused on DNS, and many of these involve attacking DNS cache on name servers. An attacker who owns a name server may coerce a victim name server into caching false records by querying the victim about the attacker's own site. A vulnerable victim name server would then refer to the rogue server and cache the answer [9].

3. DDoS attacks

3.1. Defining DDoS attacks

According to the WWW Security FAQ [4] on Distributed Denial of Service (DDoS) attacks: "A DDoS attack uses many computers to launch a coordinated DoS attack against one or more targets. Using client/server technology, the perpetrator is able to multiply the effectiveness of the DoS significantly by harnessing the resources of multiple unwitting accomplice computers, which serve as attack platforms". The DDoS attack is the most

advanced form of DoS attacks. It is distinguished from other attacks by its ability to deploy its weapons in a "distributed" way over the Internet and to aggregate these forces to create lethal traffic. DDoS attacks never try to break the victim's system, thus making any traditional security defense mechanism inefficient. The main goal of a DDoS attack is to cause damage on a victim either for personal reasons, either for material gain, or for popularity.

DDoS attacks mainly take advantage of the Internet architecture and this is that makes them even more powerful. The Internet was designed with functionality, not security, in mind. Its design opens several security issues that can be exploited by attackers. More analytically

- *Internet security is highly interdependent*. No matter how secure a victim's system may be, whether or not this system will be a DDoS victim depends on the rest of the global Internet [10].
- *Internet resources are limited*. No Internet host has unlimited resources that sooner or later can be consumed by a sufficient number of users.
- *Many against a few*. If the resources of attackers are greater than the resources of the victims then the success of the attack is almost definite.
- *Intelligence and resources are not collocated*. Most of the intelligence needed for service guarantees is located in end hosts. At the same time in order to have large throughput high bandwidth pathways are designed in the intermediate network. This way, attackers can exploit the abundant resources of an unwitting network in order to flood a victim with messages.

3.2. DDoS strategy

A Distributed Denial of Service Attack is composed of four elements, as shown in Fig. 2:

- The real attacker.
- The handlers or masters, which are compromised hosts with a special program running on them, capable of controlling multiple agents.
- The attack daemon agents or zombie hosts, who are compromised hosts that are running a special program and are responsible for generating a stream of packets towards the intended victim. Those machines are commonly external

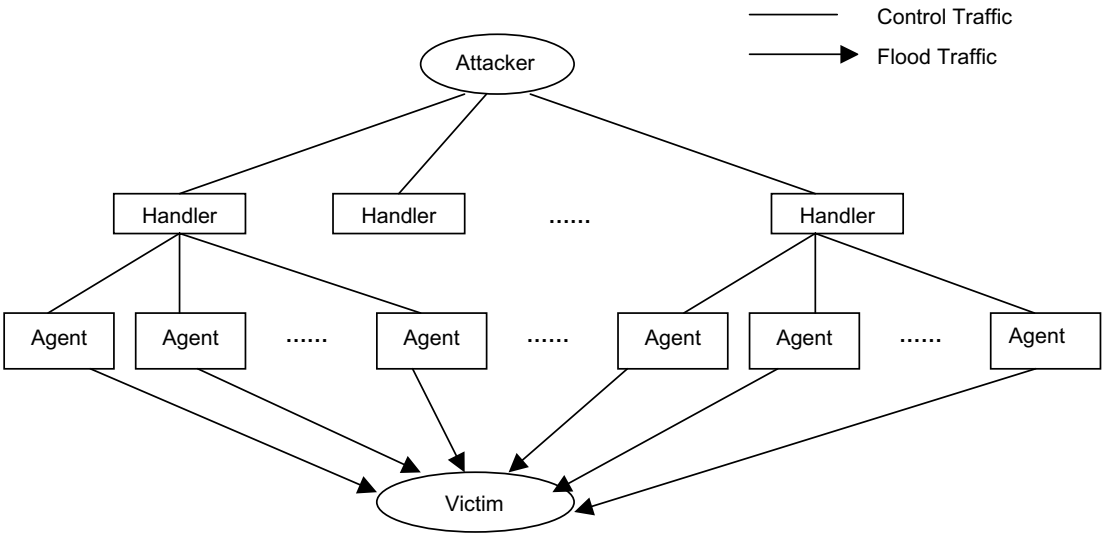

Fig. 2. Architecture of DDoS attacks.

to the victim's own network, to avoid efficient response from the victim, and external to the network of the attacker, to avoid liability if the attack is traced back.
- A victim or target host.

The following steps take place while preparing and conducting a DDoS attack:

1. *Selection of agents*. The attacker chooses the agents that will perform the attack. These machines need to have some vulnerability that the attacker can use to gain access to them. They should also have abundant resources that will enable them to generate powerful attack streams. At the beginning this process was performed manually, but it was soon automated by scanning tools.
2. *Compromise*. The attacker exploits the security holes and vulnerabilities of the agent machines and plants the attack code. Furthermore he tries to protect the code from discovery and deactivation. Self-propagating tools such as the Ramen worm [11] and Code Red [12] soon automated this phase. The owners and users of the agent systems typically have no knowledge that their system has been compromised and that they will be taking part in a DDoS attack. When participating in a DDoS attack, each agent program uses only a small amount of resources (both in memory and bandwidth), so that the users of computers experience minimal change in performance.
3. *Communication*. The attacker communicates with any number of handlers to identify which agents are up and running, when to schedule attacks, or when to upgrade agents. Depending on how the attacker configures the DDoS attack network, agents can be instructed to communicate with a single handler or multiple handlers. The communication between attacker and handler and between the handler and agents can be via TCP, UDP, or ICMP protocols.
4. *Attack*. At this step the attacker commands the onset of the attack. The victim, the duration of the attack as well as special features of the attack such as the type, length, TTL, port numbers etc, can be adjusted. The variety of the properties of attack packets can be beneficial for the attacker, in order to avoid detection.

Recently a multi-user, on-line chatting system known as Internet Relay Chat (IRC) channels started being used for communication between the attacker and the agents [13], since IRC chat networks allow their users to create public, private and secret channels. An IRC-based DDoS attack network is similar to the agent–handler DDoS attack model except that instead of using a handler program installed on a network server, an IRC server tracks the addresses of connected agents and handlers and facilitates communication between them. The discovery of the single participant leads to the discovery of the communication channel, but other participants' identities are protected. IRC offers several other advantages for delivering a DDoS attack, pointing to three major benefits: it affords a high degree of anonymity; it is difficult to detect; and it provides a strong, guaranteed delivery system. Furthermore, the attacker no longer needs to maintain a list of agents, since he can simply log on to the IRC server and see a list of all available agents [14]. The agent software installed in the IRC network usually communicates to the IRC channel and notifies the attacker when the agent is up and running. In an IRC-based DDoS attack, the agents are often referred to as "Zombie Bots" or "Bots".

3.3. DDoS tools

There are several known DDoS attack tools. The architecture of these tools is very similar and in fact some tools have been constructed through minor modifications of other tools. In this section, we present the functionality of some of these tools. For presentation purposes we divide them in agent-based and IRC-based DDoS tools.

3.3.1. Agent-based DDoS tools

Trinoo [15] is credited with being the first DDoS attack tool to be widely distributed and used. Trinoo [16] is a bandwidth depletion attack tool that can be used to launch coordinated UDP flood attacks against one or many IP addresses. The

attack uses constant-size UDP packets to target random ports on the victim machine. Early versions of trinoo appear to support IP source address spoofing. Typically, the trinoo agent gets installed on a system that suffers from remote buffer overrun exploitation. This "bug" in the software allows an attacker to remotely compile and run the agent installation within the secondary victim's system buffer. The handler uses UDP or TCP to communicate with the agents so intrusion detection systems can only find them by sniffing for UDP traffic. This channel can be encrypted and password protected as well. However currently the password is not sent in encrypted format, so it can be "sniffed" and detected. Trinoo does not spoof source addresses although it can easily be extended to include this capability. Trinoo's attack daemons implement UDP Flood attacks against the target victim.

Tribe Flood Network (TFN) [17], written in 1999, is a DDoS attack tool that provides the attacker with the ability to wage both bandwidth depletion and resource depletion attacks. It uses a command line interface to communicate between the attacker and the control master program but offers no encryption between agents and handlers or between handlers and the attacker. In addition to Trinoo's UDP flooding it also allows TCP SYN and ICMP flood as well as smurf attacks. Handlers are accessed using standard TCP connections like telnet or ssh. Other alternatives are ICMP tunneling tools like LOKI [18,19]. Communication between the handler and the daemons is accomplished with ICMP ECHO REPLY packets, which are harder to detect than UDP packets and can often pass firewall systems. TFN launches coordinated Denial of Service attacks that are especially difficult to counter as it can generate multiple types of attacks and it can generate packets with spoofed source IP addresses and also randomize the target ports. It is capable of spoofing either one or all 32 bits of the IP source address, or just the last eight bits. Some of the attacks that can be launched by TFN include: Smurf, UDP flood, TCP SYN flood, ICMP echo request flood, and ICMP directed broadcast.

TFN2K [20] is a DDoS attack tool based on the TFN architecture. The TFN2K attack tool adds encrypted messaging between all of the attack components [21]. Communication between the real attacker and control master program is encrypted using a key-based CAST-256 algorithm [22]. In addition, TFN2K conducts covert exercises to hide itself from intrusion detection systems. TFN2K attack daemons implement Smurf, SYN, UDP, and ICMP Flood attacks. Targets are attacked via UDP, TCP SYN, ICMP_ECHO flood or smurf attack, and the attack type can be varied during the attack. Commands are sent from the master to the agent via TCP, UDP, ICMP, or all three at random, making it harder to detect TFN2K by scanning the network.

The command packets may be interspersed with any number of decoy packets sent to random IP addresses to avoid detection. In networks that employ ingress filtering as described in [23], TFN2K can forge packets that appear to come from neighboring machines. All communication between handlers and agents is encrypted and base-64 encoded. There is one additional attack form called TARGA attack. TARGA works by sending malformed IP packets known to slow down or hang-up many TCP/IP network stacks. Another option is the called MIX attack, which mixes UDP, SYN and ICMP ECHO REPLY flooding [24].

Stacheldraht [25] (German term for "barbed wire") is based on early versions of TFN and attempts to eliminate some of its weak points. It combines features of Trinoo (handler/agent architecture) with those of the original TFN. It also has the ability to perform updates on the agents automatically. This means that the attacker can provide the installation file on an anonymous server and when each agent system turns on (or logs on to the Internet), the agent will automatically look for updates and install them. Stacheldraht also provides a secure telnet connection via symmetric key encryption between the attacker and the handler systems. Communication is performed through TCP and ICMP packets. Some of the attacks that can be launched by Stacheldraht include UDP flood, TCP SYN flood, ICMP echo request flood, and ICMP directed broadcast. The attack daemons for Stacheldraht implement Smurf, SYN Flood, UDP Flood, and ICMP

Flood attacks. New program versions have more features and different signatures.

The *mstream* [26] tool uses spoofed TCP packets with the ACK flag set to attack the target. Mstream is a simple point-to-point TCP ACK flooding tool that, as a side effect, can overwhelm the tables used by fast routing routines in some switches. Communication is not encrypted and is performed through TCP and UDP packets and the master connects via telnet to zombie. Access to the handler is password protected. This program has a feature not found in other DDoS tools. It informs all connected users of access, successful or not, to the handler(s) by competing parties. The target gets hit by ACK packets and sends TCP RST to non-existent IP addresses. Routers will return "ICMP unreachable" causing more bandwidth starvation. It has very limited control features and can spoof randomizes all 32 bits of the source IP address.

Shaft [27] is a derivative of the trinoo tool. It uses UDP communication between handlers and agents. The attacker communicates with the handlers via a TCP telnet connection. UDP is used for communication between handlers and agents, and messages are not encrypted Shaft provides UDP, ICMP, and TCP flooding attack options. The attacks can be run individually, or they can be combined to form one attack with UDP/TCP/ICMP flooding. Shaft randomizes the source IP address and the source port in packets. The size of packets remains fixed during the attack. A new feature is the ability to switch the handler's IP address and port in real time during the attack, making the detection tools difficult. A distinctive feature of Shaft is the ability to switch control master servers and ports in real time, hence making detection by intrusion detection tools difficult. Furthermore shaft provides statistics on the flood attack. These statistics are useful to the attacker to know when the victim system is completely down and allows the attacker to know when to stop adding zombie machines to the DDoS attack.

3.3.2. IRC-based DDoS attack tools

IRC-based DDoS attack tools were developed after the agent–handler attack tools. This has as a result many IRC-based tools to be more sophisticated as they include some important features that can be found in many agent–handler attack tools.

Trinity v3 [28] besides the up to now well-known UDP, TCP SYN, TCP ACK, TCP NUL packet floods introduces TCP fragment floods, TCP RST packet floods, TCP random flag packet floods, and TCP established floods, while randomizing all 32 bits of the source IP address [29]. It also generates TCP flood packets with random control flags set and this way, a wider set of TCP based attacks is provided by Trinity. In the same generation with Trinity is *myServer* [27], which relies on external programs to provide DoS and *Plague* [27], which provides TCP ACK and TCP SYN flooding.

Knight is an IRC-based DDoS attack tool very lightweight and powerful that was first reported in July 2001 [29]. The Knight DDoS attack tool provides SYN attacks, UDP Flood attacks, and an urgent pointer flooder [30]. It is designed to run on Windows operating systems and has features such as an automatic updater via http or ftp, a checksum generator and more. The Knight tool is typically installed by using Trojan horse program called Back Orifice [29]. Another DDoS tool that is based on Knight is *Kaiten* [27], which includes UDP, TCP flood attacks, SYN and PUSH+ACH attacks and randomizes the 32 bits of its source address.

3.4. DDoS classification

To be able to understand DDoS attacks it is necessary to have a formal classification. We propose a classification of DDoS attacks that combines efficiently the classifications proposed by Mirkovic et al. [31], Lee [32] and more recent research results. This classification is illustrated in Fig. 3 and consists of two levels. In the first level attacks are classified according to their degree of automation, exploited vulnerability, attack rate dynamics and their impact. In the second level specific characteristics of each first level category are recognized.

3.4.1. Classification by degree of automation

Based on the degree of automation of the attack DDoS attacks can be classified into *manual*, *semi-automatic* and *automatic* DDoS attacks.

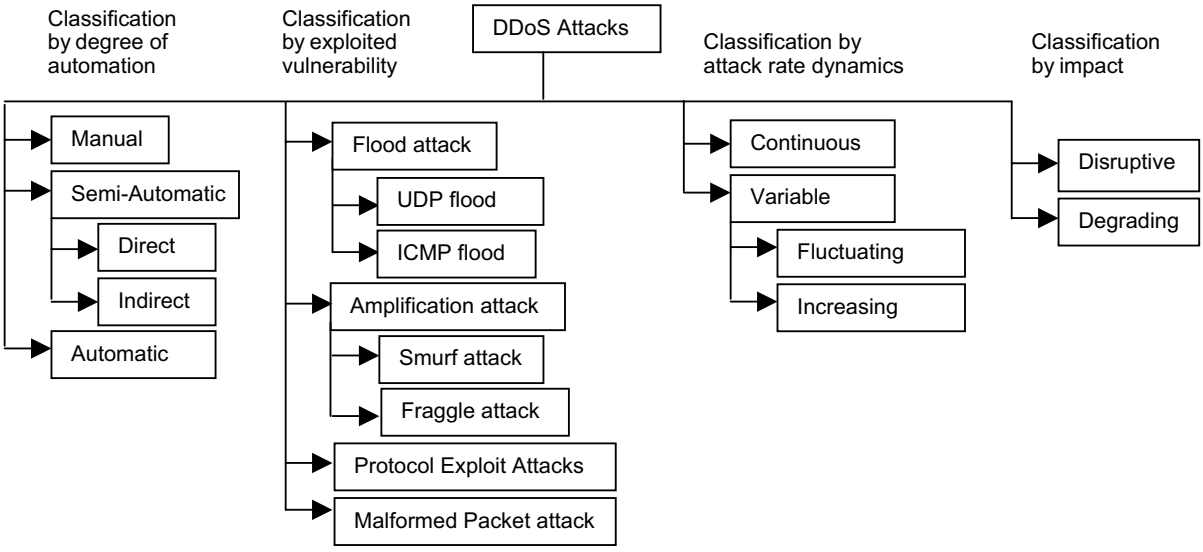

Fig. 3. Classification of DDoS attacks.

- The early DDoS attacks were manual. This means that the DDoS strategy included the scanning of remote machines for vulnerabilities, breaking into them and installing the attack code. All of these steps were later automated, by the use of semi-automatic DDoS attacks, and automatic DDoS attacks.
- In *semi-automatic attacks*, the DDoS attack belongs in the agent–handler attack model. The attacker scans and compromises the handlers and agents by using automated scripts. The attack type, the victim's address and the onset of the attack are specified by the handler machines. Semi-automatic attacks can be divided further to *attacks with direct communication* and *attacks with indirect communication*. *Attacks with direct communication* include attacks, during which the agent and handler need to know each other's identity in order to communicate. This approach includes the hard coding of the IP address of the handler machines. The main drawback of this approach is that the discovery of one compromised machine can expose the whole DDoS network. *Attacks with indirect communication* use indirection in order to achieve a greater survivability of DDoS attacks. A representative example of this kind of attacks is the IRC-based DDoS attacks that has already been discussed in the previous section.
- In *automatic DDoS attacks* the communication between attacker and agent machines is completely avoided. In most cases the attack phase is limited to a single command. All the features of the attack, for example the attack type, the duration and the victim's address, are preprogrammed in the attack code. This way, the attacker has a minimal exposure and the possibility of revealing his identity is small. The drawback of this approach is that the propagation mechanisms usually leave the backdoor to the compromised machine open, making possible future access and modification of the attack code.

3.4.2. Classification by exploited vulnerability

DDoS attacks according to the exploited vulnerability can be divided in the following categories: *flood attacks*, *amplification attacks*, *protocol exploit attacks* and *malformed packet attacks*.

- In a *flood attack*, the zombies send large volumes of IP traffic to a victim system in order to congest the victim system's bandwidth. The

impact of packet streams sent by the zombies to the victim system varies from slowing it down or crashing the system to saturation of the network bandwidth. Some of the well-known flood attacks are *UDP flood attacks* and *ICMP flood attacks*.

A *UDP Flood attack* is possible when a large number of UDP packets is sent to a victim system. This has as a result the saturation of the network and the depletion of available bandwidth for legitimate service requests to the victim system. In a DDoS UDP Flood attack, the UDP packets are sent to either random or specified ports on the victim system. Typically, UDP flood attacks are designed to attack random victim ports. A UDP Flood attack is possible when an attacker sends a UDP packet to a random port on the victim system. When the victim system receives a UDP packet, it will determine what application is waiting on the destination port. When it realizes that there is no application that is waiting on the port, it will generate an ICMP packet of "destination unreachable" [14] to the forged source address. If enough UDP packets are delivered to ports of the victim, the system will go down. By the use of a DDoS tool the source IP address of the attacking packets can be spoofed and this way the true identity of the secondary victims is prevented from exposure and the return packets from the victim system are not sent back to the zombies.

ICMP Flood attacks exploit the *Internet Control Message Protocol* (ICMP), which enables users to send an echo packet to a remote host to check whether it's alive. More specifically during a DDoS ICMP flood attack the agents send large volumes of ICMP_ECHO_REPLY packets ("ping") to the victim. These packets request reply from the victim and this has as a result the saturation of the bandwidth of the victim's network connection [15]. During an ICMP flood attack the source IP address may be spoofed.

- In *amplification attacks* the attacker or the agents exploit the broadcast IP address feature found on most routers to amplify and reflect the attack and send messages to a broadcast IP address. This instructs the routers servicing the packets within the network to send them to all the IP addresses within the broadcast address range. This way the malicious traffic that is produced reduces the victim system's bandwidth. In this type of DDoS attack, the attacker can send the broadcast message directly, or by the use of agents to send the broadcast message in order to increase the volume of attacking traffic. If the broadcast message is sent directly, the attacker can use the systems within the broadcast network as agents without needing to infiltrate them or install any agent software. Some well known amplification attacks, are *Smurf* and *Fraggle attacks*.

The intermediary nodes that are used as attack launchers in amplification attacks are called reflectors [33]. A reflector is any IP host that will return a packet if sent a packet. So, web servers, DNS servers, and routers are reflectors, since they return SYN ACKs or RSTs in response to SYN or other TCP packets.

An attacker sends packets that require responses to the reflectors. The packets are address-spoofed with source addresses set to a victim's address. The reflectors return response packets to the victim according to the types of the attack packets. The attack packets are essentially reflected in the normal packets towards the victim. The reflected packets can flood the victim's link if the number of reflectors is large enough. Note that the reflectors are readily identified as the source addresses in the flooding packets received by the victim. The operator of a reflector on the other hand, cannot easily locate the slave that is pumping the reflector, because the traffic sent to the reflector does not have the slave's source address, but rather the source address of the victim.

The attack architecture of reflector attacks is very similar to the one used for direct ones. However, there are several important differences [34].

- A reflector attack requires a set of predetermined reflectors.
- The reflectors could also be dispersed on the Internet, because the attacker does not need to install any agent software.

- The reflected packets are normal packets with legitimate source addresses and cannot be filtered based on route-based mechanisms.

  Smurf attacks send ICMP echo request traffic with a spoofed source address [35] of the target victim to a number of IP broadcast addresses. Most hosts on an IP network will accept ICMP echo requests [35] and reply to the source address, in this case, the target victim. On a broadcast network, there could potentially be hundreds of machines to reply to each ICMP packet. The use of a network in order to elicit many responses to a single packet has been labeled as "amplifier" [36]. In this type of attack the party that is hurt is not only the spoofed source address target (the victim) but also he intermediate broadcast devices (amplifiers). The Fraggle attacks are a similar attack to the Smurf except that they use UDP echo packets instead of ICMP echoes. Fraggle attacks generate even more bad traffic and can create even more damaging effects than just a Smurf attack.

- *Protocol exploit attacks* exploit a specific feature or implementation bug of some protocol installed at the victim in order to consume excess amounts of its resources. A representative example of protocol exploit attacks is *TCP SYN attacks*.

  TCP SYN attacks exploit the inherent weakness of the three-way handshake involved in the TCP connection setup. A server, upon receiving an initial SYN (synchronize/start) request from a client, sends back a SYN/ACK (synchronize/acknowledge) packet and waits for the client to send the final ACK (acknowledge). An attacker initiates an SYN flooding attack by sending a large number of SYN packets and never acknowledges any of the replies, essentially leaving the server waiting for the nonexistent ACK's [37]. Considering that the server only has a limited buffer queue for new connections, SYN Flood results in the server being unable to process other incoming connections as the queue gets overloaded [38].

  Other examples of protocol exploit attacks are PUSH + ACK attacks, CGI request attacks and the authentication server attacks.

- *Malformed packet attacks* [32] rely on incorrectly formed IP packets that are sent from agents to the victim in order to crash the victim system. The malformed packet attacks can be divided in two types of attacks: *IP address attack* and *IP packet options attack*. In an *IP address attack*, the packet contains the same source and destination IP addresses. This has as a result the confusion of the operating system of the victim system and the crash of the victim system. In an *IP packet options attack*, a malformed packet may randomize the optional fields within an IP packet and set all quality of service bits to one. This would have as a result the use of additional processing time by the victim in order to analyze the traffic. If this attack is combined with the use of multiple agents, it could lead to the crash of the victim system.

3.4.3. Classification by attack rate dynamics

Depending on the attack rate dynamics DDoS attacks can be divided in *continuous rate* and *variable rate* attacks.

- *Continuous rate attacks* comprise attacks that after the onset of the attack are executed with full force and without a break or decrement of force. The impact of such an attack is very quick.
- *Variable rate attacks* as their name indicates, "vary the attack rate" and thus they avoid detection and immediate response. Based on the rate change mechanism we differentiate between attacks with *increasing rate* and *fluctuating rate*. *Increasing rate* attacks gradually lead to the exhaustion of victim's resources, thus delaying detection of the attack. *Fluctuating rate* attacks have a wavy rate that is defined by the victim's behavior and response to the attack, at times decreasing the rate in order to avoid detection.

3.4.4. Classification by impact

Based on the impact of a DDoS attack we can divide DDoS attacks to *disruptive* and *degrading* attacks.

- *Disruptive attacks* lead to the complete denial of the victim's service to its clients.

- The goal of *degrading attacks* is to consume some portion of a victim's resources. This has as an effect the delay of the detection of the attack and at the same time an immense damage on the victim.

4. DDoS defense problems and classification

DDoS attacks are a hard problem to solve. First, there are no common characteristics of DDoS streams that can be used for their detection. Furthermore, the distributed nature of DDoS attacks makes them extremely difficult to combat or trace back. Moreover, the automated tools that make the deployment of a DDoS attack possible can be easily downloaded. Attackers may also use IP spoofing in order to hide their true identity, and this makes the traceback of DDoS attacks even more difficult. Finally, there is no sufficient security level on all machines in the Internet, while there are persistent security holes in Internet hosts.

We may classify DDoS defense mechanisms using two different criteria. The first classification categorizes the DDoS defense mechanisms according to the activity deployed. Thus we have the following four categories:

- Intrusion Prevention,
- Intrusion Detection,
- Intrusion Tolerance and Mitigation, and
- Intrusion Response.

The second classification divides the DDoS defenses according to the location deployment resulting into the following three categories of defense mechanisms:

- Victim Network,
- Intermediate Network, and
- Source Network.

Our classification of DDoS mechanisms is illustrated in Fig. 4. In the following, we discuss extensively the techniques used in each of the categories of the first classification and just refer to the DDoS defenses and the way they are categorized for the last classification.

5. Classification by activity

5.1. Intrusion prevention

The best mitigation strategy against any attack is to completely prevent the attack. In this stage we try to stop DDoS attacks from being launched in the first place. There are many DDoS defense mechanisms that try to prevent systems from attacks:

Using globally coordinated filters, attacking packets can be stopped, before they aggregate to lethal proportions. Filtering mechanisms can be divided into the following categories:

Ingress filtering is an approach to set up a router such that to disallow incoming packets with illegitimate source addresses into the network. Ingress filtering, proposed by Ferguson and Senie [39], is a restrictive mechanism to drop traffic with IP address that does not match a domain prefix connected to the ingress router. This mechanism can drastically reduce the DoS attack by IP spoofing if all domains use it. Sometimes legitimate traffic can be discarded by an ingress filtering when Mobile IP [40] is used to attach a mobile node to a foreign network.

Egress filtering [41] is an outbound filter, which ensures that only assigned or allocated IP address space leaves the network. Egress filters do not help to save resource wastage of the domain where the packet is originated but it protects other domains from possible attacks. Besides the placement issue, both ingress and egress filters have similar behavior.

Route-based distributed packet filtering has been proposed by Park and Lee [42]. This approach is capable of filtering out a large portion of spoofed IP packets and preventing attack packets from reaching their targets as well as to help in IP traceback. Route-based filters use the route information to filter out spoofed IP packets, making this their main difference from ingress filtering. If route-based filters are partially deployed, a synergistic filtering effect is possible, so that spoofed IP flows are prevented from reaching other Autonomous Systems. Furthermore, since routes on the Internet change with time [43] it is a great challenge for route-based filters to be

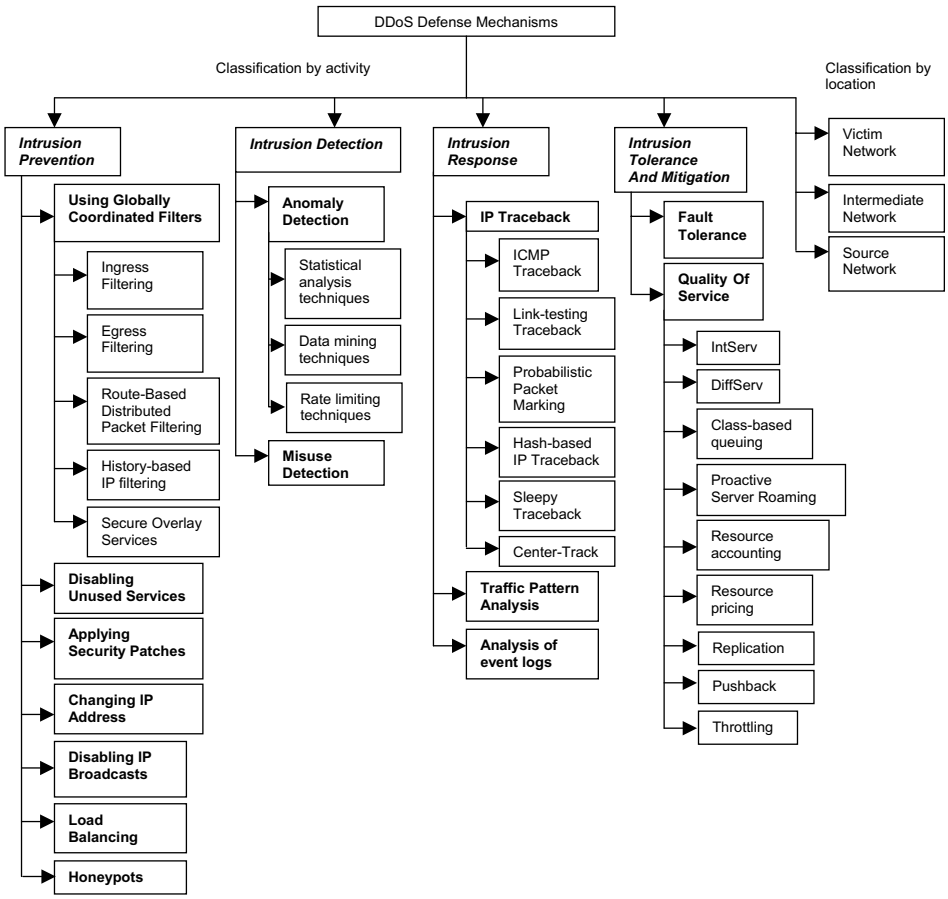

Fig. 4. Classification of DDoS defense mechanisms.

updated in real time. The main disadvantage of this approach is that it requires global knowledge of the network topology leading to scalability issues.

History-based IP filtering (HIP) is another filtering mechanism that has been proposed by Peng et al. [44] in order to prevent DDoS attacks. According to this approach the edge router admit the incoming packets according to a pre-built IP address database. The IP address database is based on the edge router's previous connection history. This scheme is robust, does not need the cooperation of the whole Internet community, is applicable to a wide variety of traffic types and requires little configuration. On the other hand, if the attackers know that the IP packet filter is based on previous connections, they could mislead the server to be included in the IP address database. This can be prevented by increasing the period over which IP addresses must appear in order to be considered frequent.

Secure Overlay Services (SOS) [45] is an architecture in which only packets coming from a small number of nodes, called servlets, are assumed to be legitimate client traffic that can reach the servlets through hash-based routing inside an overlay network. All other requests are filtered by the overlay. In order to gain access to the overlay network, a client has to authenticate itself with one of the replicated access points (SOAPs). SOS is a distributed system that offers excellent protection to the specified target at the cost of modifying client systems, thus it is not suitable for protection of public servers.

Disabling unused services [46] is another approach in order to prevent DDoS attacks. If UDP echo or character generator services are not required, disabling them will help to defend against these attacks. In general, if network services are not needed or unused, the services should be disabled to prevent attacks.

Applying security patches [46], can armor the hosts against DDoS attacks. The host computers should update themselves with the latest security patches for the bugs present and use the latest techniques available to minimize the effect of DDoS attack.

Changing IP address [46], is another simple solution to a DDoS attack in order to invalidate the victim computer's IP address by changing it with a new one. This is called moving target defense. Once the IP address change is completed all Internet routers will have been informed, and edge routers will drop the attacking packets. Although this action leaves the computer vulnerable because the attacker can launch the attack at the new IP address, this option is practical for local DDoS attacks, which are based on IP addresses. On the other hand, attackers can render this technique a futile process by adding a domain name service tracing function to the DDoS attack tools.

By *disabling IP broadcasts* [46], host computers can no longer be used as amplifiers in ICMP Flood and Smurf attacks. However, a defense against this attack will be successful only if all the neighboring networks disable IP broadcasts.

Load balancing [32] is a simple approach that enables network providers to increase the provided bandwidth on critical connections and prevent them from going down in the event of an attack. Additional failsafe protection can be the use the replication of servers in the case some go down during a DDoS attack. Furthermore, in a multiple-server architecture the balance of the load is necessary so that both the improvement of normal performance as well as the prevention or mitigation of the effect of a DDoS attack can be achieved.

Honeypots [47] can also be used in order to prevent DDoS attacks. Honeypots are systems that are set up with limited security and can be used to trick the attacker to attack the honeypot and not the actual system. Honeypots typically have value not only in protecting systems, but they can also be used in order to gain information about attackers by storing a record of their activity and learning what types of attacks and software tools the attacker is using. Current research discusses the use of honeypots that mimic all aspects of a legitimate network (such as web servers, mail servers, clients, etc.) in order to attract potential DDoS attackers. The idea is to lure the attacker into the believing that he has compromised the system (e.g. honeypot) for attack as its slave and attract him to install either handler or agent code within the honeypot. This prevents some legitimate systems from getting compromised, tracks the handler or agent behavior and allows the system to better understand how to defend against future DDoS installation attacks. However, this scheme has several drawbacks. First, the method assumes that the attack must be detectable using signature-based detection tools. If not, the packet is forwarded to the destination in operational networks. Furthermore, the attacker can easily thwart the static and passive nature of honeypot's approach since the approach is static and passive in the sense that it is not a dynamically moving scheme with complete disguise.

Prevention approaches offer increased security but can never completely remove the threat of DDoS attacks because they are always vulnerable to new attacks for which signatures and patches do not exist in the database.

5.2. Intrusion detection

Intrusion detection has been a very active research area. By performing intrusion detection, a host computer and a network can guard themselves against being a source of network attack as well as being a victim of a DDoS attack. Intrusion detection systems detect DDoS attacks either by using the database of known signatures or by recognizing anomalies in system behaviors.

Anomaly detection relies on detecting behaviors that are abnormal with respect to some normal standard. Many anomaly detection systems and approaches have been developed to detect the faint signs of DDoS attacks.

A scalable network monitoring system called NOMAD has been designed by Talpade et al. [48]. This system is able to detect network anomalies by making statistical analysis of IP packet header information. It can be used for detecting the anomalies of the local network traffic and does not support a method for creating the classifier for the high-bandwidth traffic aggregate from distributed sources.

Another detection method of DDoS attacks uses the Management Information Base (MIB) data from routers. The MIB data from a router includes parameters that indicate different packet and routing statistics. Cabrera et al. [49] has focused on identifying statistical patterns in different parameters, in order to achieve the early detection of DDoS attacks. It looks promising for possibly mapping ICMP, UDP and TCP packet statistical abnormalities to specific DDoS attacks. Although, this approach can be effective for controlled traffic loads, it needs to be further evaluated in a real network environment. This research area could provide important information and methods that can be used in the identification and filtering of DDoS attacks.

A mechanism called congestion triggered packet sampling and filtering has been proposed by Huang and Pullen [50]. According to this approach, a subset of dropped packets due to congestion is selected for statistical analysis. If an anomaly is indicated by the statistical results, a signal is sent to the router to filter the malicious packets.

Lee and Stolfo [51] use data mining techniques to discover patterns of system features that describe program and user behavior and compute a classifier that can recognize anomalies and intrusions. This approach focuses on the host-based intrusion detection. An improvement of this approach is a meta-detection model [52], which uses results from multiple models to provide more accurate detection.

Mirkovic et al. [53] proposed a system called D-WARD that does DDoS attack detection at the source based on the idea that DDoS attacks should be stopped as close to the sources as possible. D-WARD is installed at the edge routers of a network and monitors the traffic being sent to and from the hosts in its interior. If an asymmetry in the packet rates generated by an internal host is noticed, D-WARD rate limits the packet rate. The drawback of this approach is that there is a possibility of numerous false positives while detecting DDoS conditions near the source, because of the asymmetry that there might be in the packet rates for a short duration. Furthermore, some legitimate flows like real time UDP flows do exhibit asymmetry.

In [54] Gil and Poletto proposed a heuristic data-structure (MULTOPS), which postulates if the detection of IP addresses that participate in a DDoS attack is possible, then measures are taken to block only these particular addresses. Each network device maintains a multi-level tree that contains packet rate statistics for subnet prefixes at different aggregation levels. MULTOPS uses disproportional rates to or from hosts and subnets to detect attacks. 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. Among the disadvantages of this approach, is that it requires router reconfiguration and new memory management schemes. Furthermore, it cannot prevent proportional attacks nor can it detect randomized forged IP addresses originating from a single machine or DDoS attacks that uses many zombies.

Misuse detection identifies well-defined patterns of known exploits and then looks out for the occurrences of such patterns. Intrusion patterns can be any packet features, conditions, arrangements and interrelationships among events that lead to a break-in or other misuse. These patterns are defined as intrusion signatures. Several popu-

lar network monitors perform signature-based detection, such as CISCO's NetRanger [55], NID [56], SecureNet PRO [57], RealSecure [58], NFR-NID [59] and Snort [60].

5.3. Intrusion response

Once an attack is identified, the immediate response is to identify the attack source and block its traffic accordingly. The blocking part is usually performed under manual control (e.g. by contacting the administrators of upstream routers and enabling access control lists) since an automated response system might cause further service degradation in response to a false alarm. Automated intrusion response systems do exist, but they are deployed only after a period of self-learning (for the ones that employ neural computation in order to discover the DDoS traffic) or testing (for the ones that operate on static rules). Improving attack source identification, techniques can expedite the capture of attackers and deter other attack attempts. There are many approaches that target the tracing and identifying of the real attack source [61].

IP traceback traces the attacks back towards their origin, so one can find out the true identity of the attacker and achieve detection of asymmetric routes, as well as path characterization. Some factors that render IP traceback difficult is the stateless nature of Internet routing and the lack of source accountability in the TCP/IP protocol. For efficient IP traceback it is necessary to compute and construct the attack path. It is also necessary to have a low router overhead and low false positive rate. Furthermore, a large number of packets is required to reconstruct the attack path. It is also important the robustness against multiple attacks, the reduction of the privacy of IP communication, the incremental deployment and the backward compatibility. 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 Internet Service Provider (ISP) asking for the direction from which the packets are coming. Since the manual traceback is very tedious there have been various proposals in the recent past to automate this process.

ICMP traceback has been proposed by Bellovin [24]. According to this mechanism every router samples the forwarding packets with a low probability (1 out of 20,000) and sends an ICMP traceback message to the destination. If enough traceback messages are gathered at the victim, the source of traffic can be found by constructing a chain of traceback messages. A major issue of this approach is the validation of the traceback packets. Although the PKI requirement prevents attackers from generating false ICMP traceback messages, it is unlikely that every router will implement a certificate-based scheme. Furthermore, ICMP traffic generates additional traffic and an upstream router map is required to construct an attack path since the IP addresses of the routers are encoded in the ICMP traceback message. An alternative, which introduces an intention-bit in the routing and forwarding table, is called Intention-Driven ICMP Traceback [62].

In order to face DDoS attacks by reflectors, Barros [63] proposed a modification of ICMP traceback messages. In this approach, routers send ICMP messages to the source of the currently being processed packet rather than its destination. This reverse trace enables the victim to identify the attacking agent(s) from these packets.

A link-testing traceback technique has been proposed by Burch and Cheswick [64]. In this scheme the victim tests each of its incoming links as a probable input link for the DDoS traffic. It infers the attack path by flooding the links with large bursts of traffic and examines whether this induces any perturbation on that network. If so, this link is probably part of an attack path. This scheme requires considerable knowledge of the network topology and the ability to generate large amounts of traffic in any network links and cannot handle multiple attackers. 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 [60].

Probabilistic packet marking (PPM) was originally introduced by Savage et al. [65], who described efficient ways to encode partial route path information and include the traceback data in IP packets. It is an approach that can be applied during or after an attack, and it does not require any additional network traffic, router storage, or packet size increase. Even though it is not impossible to reconstruct an ordered network path using an unordered collection of router samples, it requires the victim to receive a large amount of packets. The advantage of this approach is that no extra traffic is generated, since the extra information is bound to the packets. Furthermore, there is no interaction with ISPs and this mechanism can be used to trace attacks after an attack has completed.

On the other hand, there is a backward incompatibility as IP marking on the ID field conflicts with IPsec [66] in which the Authentication Header encrypts the identification header. Moreover probabilistic ID-field marking requires modifications of Internet routing devices to generate such marks on the fly. The reconstruction of an attack path [67] by the victim demands a high computation overhead. High false positives are generated when multiple attackers initiate DDoS. This approach is not robust against a compromised router. Ioannidis and Bellovin [68] argue that even though the attack path has been identified, it is not clear what are the next tasks that must follow.

Song and Perrig [67] improved the performance of PPM and suggested the use of hash chains for authenticating routers. They use a 5-bit distance field, but they do not fragment router messages. This marking scheme is efficient and accurate in the presence of a large number of DDoS and a clever encoding scheme is used to reduce the storage space requirements. On the other hand, this mechanism assumes that the victim has a map of upstream routers to all attackers and its incremental deployment is not supported.

Dean et al. [69] introduced an interesting algebraic approach to PPM. This scheme does not require an upstream router map to construct an attack path. But like the system proposed in [65] this scheme shares similar backwards compatibility problems and is less efficient in the presence of multiple attackers.

In addition to the above packet marking algorithms, Adler [70] and Park and Lee [71] study tradeoffs for various parameters in PPM. Park and Lee propose to put the distributed filters on the routers and filter the packets according to the network topology. This scheme can stop the spoofed traffic at an early stage. However, in order to be effective, there is a need to know the topology of the Internet and the routing policy between Autonomous Systems, which is hard to achieve in the expanding Internet.

A new packet marking technique and agent design has been proposed by Tupakula and Varadharajan [72] to identify the approximate source (nearest router) of the attack with a single packet, even in case of attacks with spoofed source addresses. Their scheme is a controller–agent model invoked only during attack times which not only is able to process the victims' traffic separately without disturbing other traffic but, also to establish different attack signatures for different attacking sources. The system can prevent the attack traffic at the nearest router to the attacking system, has a fast response time, is simple in its implementation and can be incrementally deployed. Unfortunately in this approach, prevention is limited within the domain of a single ISP and the efficiency decreases as the infrastructure of the ISP increases.

Snoeren et al. [73] proposed *a hash based IP traceback* technique that uses a source path isolation engine (SPIE). The SPIE generates audit trails of traffic and can trace the origin of single IP packet delivered by a network in recent past. The SPIE uses a very efficient method to store the information that a packet traversed through a particular router. The main advantage of SPIE over ICMP traceback messages and PPM is that SPIE can traceback the attack path even for low volume packets received at the victim.

Wang et al. [74] proposed a framework for "Sleepy Traceback" (i.e. watermarking and tracing packets to the attacker's source IP address, only if the IDS subsystem has determined that there is an attack in progress). This system is quite different from the ones mentioned above, in that it

utilizes the programmability of Active Nodes, in order to provide control over the Intrusion Response process. Nodes of an Active Network communicate with each other by means of specially crafted packets, called ''capsules'', that contain code. This code will effectively introduce a new service (or alter an existing one) on the node that examines it. While an attack is in progress, the Active Nodes will exchange information and reprogram their network components, so as to eliminate the DDoS traffic as close as possible to the source. Active Networks have been used and in other approaches in order to defend networks against DDoS attacks. AEGIS [75] is another mechanism that is based on active networks. The core-enabling technology of this framework is the Active Network, which incorporates programmability into intermediate network nodes and allows end-users to customize the way network nodes handle data traffic.

CenterTrack [76] is an architecture proposed by Stone, which creates an overlay network of IP tunnels by linking all edge routers to central tracking routers, and all suspicious traffic is rerouted from edge routers to the tracking routers. When a DoS attack is detected, routers at the edge of the backbone network are instructed to reroute packets that are addressed to the attack target. The tracking routers can then identify the ingress points of the main attack traffic flows. Edge routers do not have to support input debugging. On the other hand, there is a high overhead of storage and processing because of the requirement of edge routers to log packets in order to identify the attack traffic.

Traffic Pattern Analysis [32] is another method in order to response to DDoS attacks. During a DDoS attack, traffic pattern data can be stored and then analyzed after the attack in order to find specific characteristics and features that may indicate an attack. The results from this analysis of data can be used in order to update load balancing and throttling techniques as well as in developing new filtering mechanisms in order to achieve the prevention from DDoS attacks.

Analysis of event logs [32] is another good approach that targets the response to DDoS attacks. The selection of event logs that occurred during the setup and the execution of the attack can be used, in order to discover the type of DDoS attacks that has been used and do a forensic analysis. Network equipment such as firewall, packet sniffers, server logs, and honeypots [47] can be used in the selection of event logs.

5.4. Intrusion tolerance and mitigation

Research on intrusion tolerance accepts that it is impossible to prevent or stop DDoS completely and focuses on minimizing the attack impact and on maximizing the quality of its services. Intrusion tolerance can be divided in two categories: fault tolerance and quality of service (QoS).

- *Fault tolerance* is a well-developed research area whose designs are built-in in most critical infrastructures and applied in three levels: hardware, software and system [77]. The idea of fault tolerance is that by duplicating the network's services and diversifying its access points, the network can continue offering its services when flooding traffic congests one network link.
- *Quality of service* (QoS) describes the assurance of the ability of a network to deliver predictable results for certain types of applications or traffic. Many Intrusion Tolerant QoS Techniques and Intrusion Tolerant QoS systems have been developed in order to mitigate DDoS attacks.

  Among intrusion tolerant QoS techniques Integrated (IntServ) and Differentiated Services (DiffServ) have emerged as the principal architectures [78]. IntServ uses the Resource Reservation Protocol (RSVP) to coordinate the allocation of resources allocation along the path that a specific traffic flow will pass. The link bandwidth and buffer space are assured for that specific traffic flow. DiffServ [79,80] is a per-aggregate-class based discrimination framework. Diffserv makes use of the type-of-service (TOS) byte in the IP header and allocates resource based on the TOS of each packet.

  Queuing techniques are also employed extensively to combat DDoS attacks. There are many queuing disciplines. The oldest and most widely applied queuing technique is Class-based queuing (CBQ). Class-based queuing (CBQ) or

traffic shaping sets up different traffic queues for different types of packets and for packets of different TOS. A certain amount of outbound bandwidth can then be assigned to each of the queues. Class-based queuing has shown to maintain QoS during a DDoS attack on clusters of web servers [81].

An architecture that relies on the provision of QoS mechanisms in intermediate routers is VIPnets that was proposed by Brustoloni [82]. In VIPnets legitimate traffic is assumed to be the traffic coming from networks implementing the VIPnet service. All other traffic is considered as low-priority and can be dropped in the case of an attack.

A similar approach to VIPnets was adopted by Khattab et al. [83] and they propose an approach called proactive server roaming in order to mitigate DoS attacks. According to this approach the active server proactively changes its location within a pool of servers to defend against unpredictable and undetectable attacks. Only legitimate clients can track the moving server. This roaming scheme has insignificant overhead in attack-free situations and can provide good response time in case of attacks.

Using the techniques employed in Quality of Service (QoS) regulation Garg and Reddy [84] proposed a defensive approach against DDoS attacks by regulating resource consumption that belong in the category of *resource accounting*. They suggest that resource regulation can be done at the flow level, where each flow gets a fair share of the resource much in the same way as round robin scheduling in CPUs. However, it is still possible to mount a Denial of Service attack by having a large number of hosts connecting to the server each claiming their slice of the resource, thus causing resource starvation, similar to the famous dining philosophers problem. Their basic idea for this was to ''extend resource control to the network subsystem''. They split network traffic into classes where classification is based on its likely resource consumption. Other such mechanisms for regulating traffic include firewalling, ACK pacing, etc.

In the same category of *resource accounting* belongs an approach called *creating client bottlenecks*. These kinds of remedies try to create a bottleneck process on agent computers and limit their attacking capability. RSA's Client Puzzles algorithm and Turing test need the client to do some extra computation or answer a question before setting up a connection. This causes the users of zombie systems to detect performance degradation, and could possibly stop their participation in sending DDoS attack traffic. Juels and Brainard [85] propose a pricing mechanism, where the client has to solve a cryptographic problem (puzzle) with varying complexity before the server allocates resources to the requests and starts servicing it. Client puzzles allow for the ''graceful degradation of services'' when an attack occurs a server can increase the difficulty of the puzzles that the client receives and has to solve before a server accepts a client and allocates some of it's resources. The main disadvantage to the use of client puzzles is that in order for a client to deal with puzzles, the client requires specialized software. Aura, Nikander et al. [86], suggested a slight variation to those proposed by Juels and Bernard. They propose improvements to the efficiency by reducing the length of the puzzle and its solution by reducing the number of hash functions required for verification of solutions at the expense of slightly coarser puzzle granularity.

Resource pricing is another approach that was proposed by Mankins et al., in order to mitigate DDoS attacks. Mankins et al., [87] noted that DDoS attacks work because the cost falls overwhelmingly on the server, and during an attack, the attack traffic is virtually impossible to tell apart from legitimate traffic. They propose a distributed gateway architecture and a payment protocol that imposes dynamically changing prices on both network, server, and information resources in order to push some cost of initiating service requests—in terms of monetary payments and/or computational burdens—back onto the requesting clients. By employing different price and purchase functions, the architecture can provide service quality differentiation and furthermore, select good client behavior and discriminate against adver-

sarial behavior. They identify allotting a priority mechanism to desirable clients as being key, and punish clients that cause load on the server. The drawback of this approach is that a malicious user can populate the system with fake request at a low price, thus driving up the price for legitimate users. Mankins' et al. recommendation to solve this is partitioning resources into classes and using different pricing functions for each class.

Various autonomous architectures have been proposed that demonstrate intrusion tolerance during DDoS bandwidth consumption attacks. The *XenoService* [88] is an infrastructure of a distributed network of web hosts that respond to an attack on any website by replicating the web site rapidly and widely among XenoService servers, thereby allowing the attacked site to acquire more network connectivity to absorb a packet flood. Although such infrastructure can ensure QoS during DDoS attacks, it is doubtful that a large number of ISPs will adopt such an infrastructure quickly.

The *pushback architecture* is a promising mitigation technique where the idea is to notify upstream routers to rate-limit or drop specific traffic identified as poor (aggregate). In the Aggregate-based Congestion Control (ACC) [68] an aggregate is defined as a subset of the traffic with an identifiable property [89]. According to [68], a pushback daemon determines whether there is an indication of any attacks by running a detection algorithm. The incremental deployment of this approach is possible and furthermore, there is no need for upstream routers. On the other hand, there is a great storage requirement for the pushback daemon, so that dropped packets from the rate-limiter and the output queue, can be analyzed.

DARPA has supported research on sophisticated autonomous response systems based on the Cooperative Intrusion Traceback and Response Architecture (CITRA) and the Intruder Detection and Isolation Protocol (IDIP) [90]. IDIP is a special protocol for reporting intrusions and coordinating attack trace-back and response actions among network devices. The CITRA network components can be IDSs, firewalls, routers, or any devices that adopts IDIP to cooperatively trace and block network intrusions.

Throttling [91] is a mitigation approach against DDoS attacks, which prevents servers (web servers in particular) from going down. This approach uses max–min fair server-centric router throttles and 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 is rate limited to the throttle rate. This scheme can 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. This method is still in the experimental stage, however, similar techniques to throttling are being implemented by network operators. The difficulty with implementing throttling is that it is still hard to decipher legitimate traffic from malicious traffic. In the process of throttling, legitimate traffic may sometimes be dropped or delayed and malicious traffic may be allowed to pass to the servers.

5.5. Classification by deployment location

Based on the deployment location, we divide DDoS defense mechanisms to those deployed at the victim, at the intermediate or at the source network.

- *Victim-network mechanisms*. Historically, most of the systems for combating DDoS attacks have been designed to work on the victim side, since this side suffered the greatest impact of the attack. The victim has the greatest incentive to deploy a DDoS defense system, and maybe sacrifice some of its performance and resources for increased security. Examples of these systems are EMERALD [92], resource accounting [85,93,94,84,95], and protocol security mechanisms [96,97,90,98]. All these mechanisms increase a victim's ability to recognize that it is the target of an attack, and gain more time to respond.

- *Intermediate-network mechanisms*. DDoS defense mechanisms deployed at the intermediate network are more effective than a victim network mechanisms since the attack traffic can be handled easily and traced back to the attackers. Characteristic examples of these mechanisms are WATCHERS [99], traceback [65,73,68,69,74] and pushback [100]. However these defense mechanisms present several disadvantages that prevent their wide deployment such as the increase of the intermediate network's performance and the greater difficulty to detect the attack since the intermediate network usually does not feel any effect from the attack.
- *Source network mechanisms*. DDoS defense mechanisms deployed at the source network can stop attack flows before they enter the Internet core and before they aggregate with other attack flows. Being close to the sources, they can facilitate easier traceback and investigation of the attack. Examples of these mechanisms are proposed in [54,101,53]. A source network mechanism has the same disadvantage as the intermediate network mechanism of detecting the occurrence on an attack, since it does not experience any difficulties. This disadvantage can be balanced by its ability to sacrifice some of its resources and performance for better DDoS detection. However, such a system might restrict legitimate traffic from a network in the case of unreliable attack detection.

6. Conclusions

Undoubtedly, DDoS attacks present a serious problem in the Internet and challenge its rate of growth and wide acceptance by the general public, skeptical government and businesses.

In this paper, we tried to achieve a clear view of the DDoS attack problem and the numerous defense solutions that have been proposed. Having, this clear view of the problem, our thinking is clarified and this way we can find more effective solutions to the problem of DDoS attacks.

One great advantage of the development of DDoS attack and defense classifications is that effective communication and cooperation between researchers can be achieved so that additional weaknesses of the DDoS field can be identified. These classifications need to be continuously updated and expanded as new threats and defense mechanisms are discovered. Their value in achieving further research and discussion is undoubtedly large. A next step in this path would be to create sets of data and an experimental testbed so that all these various mechanisms can be compared and evaluated.

DDoS attacks are not only a serious threat for wired networks but also for wireless infrastructures. Some progress has been made in order to defend wireless networks against DDoS attacks. Geng et al. [102] propose a conceptual model for defending against DDoS attacks on the wireless Internet, which incorporates both cooperative technological solutions and economic incentive mechanisms built on usage-based fees. Further work is though needed that combines well known security drawbacks of wireless protocols with defense techniques that are already mature in a wireless environment.

References

[1] CERT Coordination Center, Denial of Service attacks, Available from <http://www.cert.org/tech_tips/denial_of_service.html>.
[2] Computer Security Institute and Federal Bureau of Investigation, CSI/FBI Computer crime and security survey 2001, CSI, March 2001, Available from <http://www.gocsi.com>.
[3] D. Moore, G. Voelker, S. Savage, Inferring Internet Denial of Service activity, in: Proceedings of the USENIX Security Symposium, Washington, DC, USA, 2001, pp. 9–22.
[4] L.D. Stein, J.N. Stewart, The World Wide Web Security FAQ, version 3.1.2, February 4, 2002, Available from <http://www.w3.org/Security/Faq>.
[5] D. Karig, R. Lee, Remote Denial of Service Attacks and countermeasures, Department of Electrical Engineering, Princeton University, Technical Report CE-L2001-002, October 2001.
[6] CIAC, Information Bulletin, I-020: Cisco 7xx password buffer overflow, Available from <http://ciac.llnl.gov/ciac/bulletins/i-020.shtml>.
[7] Kenney, Malachi, Ping of Death, January 1997, Available from <http://www.insecure.org/sploits/ping-o-death.html>.

[8] Finger bomb recursive request, Available from <http://xforce.iss.net/static/47.php>.
[9] D. Davidowicz, Domain Name System (DNS) Security, 1999, Available from <http://compsec101.antibozo.net/papers/dnssec/dnssec.html>.
[10] CERT Coordination Center, Trends in Denial of Service attack technology, October 2001, Available from <http://www.cert.org/archive/pdf/DoS_trends.pdf>.
[11] CIAC Information Bulletin, L-040: The Ramen Worm, 2001, Available from <http://www.ciac.org/ciac/bulletins/l-040.shtml>.
[12] CERT Coordination Center, CERT Advisory CA-2001-19 'Code Red' worm exploiting buffer overflow in IIS indexing service DLL, Available from <http://www.cert.org/advisories/CA-2001-19.html>.
[13] J. Lo et al., An IRC tutorial, 1998, Available from <http://www.irchelp.org/irchelp/irctutorial.html#part1>.
[14] K.J. Houle, G.M. Weaver, Trends in Denial of Service attack technology, CERT and CERT coordination center, Carnegie Mellon University, October 2001, Available from <http://www.cert.org/archive/pdf/DoS_trends.pdf>.
[15] P.J. Criscuolo, Distributed Denial of Service Trin00, Tribe Flood Network, Tribe Flood Network 2000, and Stacheldraht CIAC-2319, Department of Energy Computer Incident Advisory (CIAC), UCRL-ID-136939, Rev. 1, Lawrence Livermore National Laboratory, February 14, 2000, Available from <http://ftp.se.kde.org/pub/security/csir/ciac/ciacdocs/ciac2319.txt>.
[16] D. Dittrich, The DoS Project's "trinoo" Distributed Denial of Service attack tool, University of Washington, October 21, 1999, Available from <http://staff.washington.edu/dittrich/misc/trinoo.analysis.txt>.
[17] D. Dittrich, The Tribe Flood Network Distributed Denial of Service attack tool, University of Washington, October 21, 1999, Available from <http://staff.washington.edu/dittrich/misc/trinoo.analysis.txt>.
[18] Phrack Magazine 7 (49), File 06 of 16 [Project LOKI], Available from <http://www.phrack.com/search.phtml?view&article=p49-6>.
[19] Phrack Magazine 7 (51) September 01, 1997, article 06 of 17 [LOKI2 (the implementation)], Available from <http://www.phrack.com/search.phtml?view&article=p51-6>.
[20] J. Barlow, W. Thrower, TFN2K—an analysis, 2000, Available from <http://security.royans.net/info/posts/bugtraq_ddos2.shtml>.
[21] CERT Coordination Center, Center Advisory CA-1999-17 Denial of Service tools, Available from <http://www.cert.org/advisories/CA-1999-17.html>.
[22] C. Adams, J. Gilchrist, The CAST-256 encryption algorithm, RFC 2612, June 1999, Available from <http://www.cis.ohio-state.edu/htbin/rfc/rfc2612.html>.
[23] P. Ferguson, D. Senie, Network ingress filtering: defeating Denial of Service attacks which employ IP source address spoofing, RFC 2827, May 2000.
[24] S. Bellovin, The ICMP traceback message, Network Working Group, Internet Draft, March 2000, Available from <http://www.research.att.com/~smb/papers/draft-bellovin-itrace-00.txt>.
[25] D. Dittrich, The 'Stacheldraht' Distributed Denial of Service attack tool, University of Washington, December 1999, Available from <http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt>.
[26] D. Dittrich, G. Weaver, S. Dietrich, N. Long, The 'mstream' Distributed Denial of Service attack tool, May 2000, Available from <http://staff.washington.edu/dittrich/misc.mstream.analysis.txt>.
[27] S. Dietrich, N. Long, D. Dittrich, Analyzing Distributed Denial of Service tools: the Shaft Case, in: Proceedings of the 14th Systems Administration Conference (LISA 2000), New Orleans, LA, USA, December 3–8, 2000, pp. 329–339.
[28] B. Hancock, Trinity v3, a DDoS tool, hits the streets, Computers Security 19 (7) (2000) 574.
[29] CERT Coordination Center, Carnegie Mellon Software Engineering Institute, CERT Advisory CA-2001-20 Continuing threats to home users, 23, 2001, Available from <http://www.cert.org/advisories/CA-2001-20.html>.
[30] Bysin, Knight.c sourcecode, PacketStormSecurity.nl July 11, 2001, Available from <http://packetstormsecurity.nl/distributed/knight.c>.
[31] J. Mirkovic, J. Martin, P. Reiher, A taxonomy of DDoS attacks and DDoS defense mechanisms, UCLA CSD Technical Report no. 020018.
[32] R.B. Lee, Taxonomies of Distributed Denial of Service networks, attacks, tools and countermeasures, Princeton University, Available from <http://www.ee.princeton.edu/~rblee>.
[33] V. Paxson, An analysis of using reflectors for Distributed Denial of Service attacks, ACM Computer Communication Review 31 (3) (2001) 38–47.
[34] R.K.C. Chang, Defending against flooding-based, Distributed Denial of Service attacks: a tutorial, IEEE Communications Magazine 40 (10) (2002) 42–51.
[35] Daemon9, Route, Infinity, IP-spoofing demystified: trust-relationship exploitation, Phrack Magazine, Guild Productions, kid, June 1996.
[36] C.A. Huegen, The latest in Denial of Service attacks: 'Smurfing' description and information to minimize effects, 2000, Available from <http://www.pentics.net/denial-of-service/white-papers/smurf.cgi>.
[37] S. Bellovin, Security problems in the TCP/IP protocol suite, Computer Communications Review 19 (2) (1989) 32–48.
[38] Cisco Systems, Inc., Defining strategies to protect against TCP SYN Denial of Service attacks, 1999, Available from <http://www.cisco.com/warp/public/707/4.html>.
[39] P. Ferguson, D. Senie, Network ingress filtering: defeating Denial of Service attacks which employ IP source address spoofing, in: RFC 2827, 2001.
[40] C. Perkins, IP mobility support for IPv4, IETF RFC 3344, 2002.

[41] Global Incident analysis Center—Special Notice—Egress filtering, Available from <http://www.sans.org/y2k/egress.htm>.
[42] K. Park, H. Lee, On the effectiveness of route-based packet filtering for Distributed DoS attack prevention in power-law Internets, in: Proceedings of the ACM SIGCOMM'01 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, ACM Press, New York, 2001, pp. 15–26.
[43] V. Paxson, End-to-end Internet packet dynamics, IEEE/ACM Transactions on Networking 7 (3) (1999) 277–292.
[44] T. Peng, C. Leckie, K. Ramamohanarao, Protection from Distributed Denial of Service attack using history-based IP filtering, in: Proceedings of IEEE International Conference on Communications (ICC 2003), Anchorage, AL, USA, 2003.
[45] A. Keromytis, V. Misra, D. Rubenstein, SoS: secure overlay services, in: Proceedings of the ACM SIGCOMM'02 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, ACM Press, New York, 2002, pp. 61–72.
[46] X. Geng, A.B. Whinston, Defeating Distributed Denial of Service attacks, IEEE IT Professional 2 (4) (2000) 36–42.
[47] N. Weiler, Honeypots for Distributed Denial of Service, in: Proceedings of the Eleventh IEEE International Workshops Enabling Technologies: Infrastructure for Collaborative Enterprises 2002, Pitsburgh, PA, USA, June 2002, pp. 109–114.
[48] R.R. Talpade, G. Kim, S. Khurana, NOMAD: Traffic-based network monitoring framework for anomaly detection, in: Proceedings of the Fourth IEEE Symposium on Computers and Communications, 1998.
[49] J.B.D. Cabrera, L. Lewis, X. Qin, W. Lee, R. K. Prasanth, B. Ravichandran, R.K. Mehra, Proactive detection of Distributed Denial of Service Attacks using MIB traffic variables—a feasibility study, in: Proceedings of the 7th IFIP/IEEE International Symposium on Integrated Network Management, Seattle, WA, May 14–18, 2001.
[50] Y. Huang, J.M. Pullen, Countering Denial of Service attacks using congestion triggered packet sampling and filtering, in: Proceedings of the 10th International Conference on Computer Communiations and Networks, 2001.
[51] W. Lee, S.J. Stolfo, Data mining approaches for intrusion detection, in: Proceedings of the 7th USENIX Security Symposium, San Antonio, TX, January 1998, pp. 79–93.
[52] W. Lee, S.J. Stolfo, K.W. Mok, A data mining framework for building intrusion detection models, in: Proceedings of the1999 IEEE Symposium on Security and Privacy, Oakland, CA, May 9–12, 1999, pp. 120–132.
[53] J. Mirkovic, G. Prier, P. Reiher, Attacking DDoS at the source, in: Proceedings of ICNP 2002, Paris, France, 2002, pp. 312–321.
[54] T.M. Gil, M. Poleto, MULTOPS: a data-structure for bandwidth attack detection, in: Proceedings of 10th Usenix Security Symposium, Washington, DC, August 13–17, 2001, pp. 23–38.
[55] Cisco, NetRanger Overview, Available from <http://www.cisco.com/univercd/cc/td/doc/product/iaabu/csids/csids1/csidsug/overview.htm>.
[56] Computer Incident Advisory Capability, Network intrusion detector overview, Available from <http://ciac.llnl.gov/cstc/nid/intro.html>.
[57] Mimestar.com, SecureNet PRO Feature List, Available from <http://www.mimestar.com/products>.
[58] Internet Security systems, Intrusion Detection Security Products, Available from <http://www.iss.net/securing_e-business/security_products/intrusion_detection/index.php>.
[59] NFR Security, NFR Network Intrusion Detection, Available from <http://www.nfr.com>.
[60] The Open Source Network Intrusion Detection System: Snort, Available from <http://www.snort.org>.
[61] P. Zaroo, A survey of DDoS attacks and some DDoS defense mechanisms, Advanced Information Assurance (CS 626).
[62] A. Mankin, D. Massey, C.L. Wu, S.F. Wu, L. Zhang, On design and evaluation of "Intention-Driven ICMP Traceback", in: Proceedings of the 10th International Conference on Computer Communications and Networks (IC3N'2001), Arizona, 2001.
[63] C. Barros, A proposal for ICMP traceback messages, Internet Draft, Available from <http://www.research.att.com/lists/ietf-itrace/2000/09/msg00044.html>, 18, 2000.
[64] H. Burch, H. Cheswick, Tracing anonymous packets to their approximate source, in: Proceedings of USENIX LISA (New Orleans) Conference, 2000, pp. 319–327.
[65] S. Savage, D. Wetherall, A. Karlin, T. Anderson, Network support for IP traceback, IEEE/ACM Transaction on Networking 9 (3) (2001) 226–237.
[66] S. Kent, R. Atkinson, Security architecture for the Internet Protocol, IETF RFC2401, 1998.
[67] D.X. Song, A. Perrig, Advanced and authenticated Marking Schemes for IP Traceback, in: Proceedings of IEEE INFOCOMM, Anchorage, AK, USA, 2001, pp. 878–886.
[68] J. Ioannidis, S.M. Bellovin, Implementing pushback: router-based defense against DDoS Attacks, in: Proceedings of Network and Distributed System Security Symposium, NDSS'02, San Diego, CA, 2002, pp. 6–8.
[69] D. Dean, M. Franklin, A. Stubblefield, An algebraic approach to IP traceback, ACM Transactions on Information and System Security 5 (2) (2002) 119–137.
[70] M. Adler, Tradeoffs in probabilistic packet marking for IP traceback, in: Proceedings of the 34th ACM Symposium Theory of Computing (STOC), Montreal, Quebec, Canada, May 19–21, 2002, pp. 407–418.
[71] K. Park, H. Lee, On the effectiveness of probabilistic packet marking for IP traceback under Denial of Service attack, in: Proceedings of IEEE INFOCOMM, Anchorage, AK, USA, 2001, pp. 338–347.
[72] U.K. Tupakula, V. Varadharajan, A practical method to counteract Denial of Service Attacks, in: Proceedings of the 26th Australian Computer Conference in Pesearch and

Practice in Information Technology, ACM International Conference Proceeding Series, 2003, pp. 204–275.
[73] A.C. Snoeren, C. Partridge, L.A. Sanchez, C.E. Jones, F. Tchakountio, S.T. Kent, W.T. Strayer, Hash-based IP traceback, in: Proceedings of the ACM SIGCOMM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, ACM Press, New York, 2001, pp. 3–14.
[74] X. Wang, D.S. Reeves, S.F. Wu, J. Yuill, Sleepy watermark tracing: an active network-based intrusion response framework, in: Proceedings of the 16th International Conference of Information Security (IFIP/SEC'01), Paris, France.
[75] E.Y. Chen, AEGIS: an Active-network-powered defense mechanism against DDoS attacks, in: Proceedings of the Third International Working Conference on Active Networks (IWAN 2001), Lecture Notes in Computer Science, vol. 2207, Springer, Berlin, 2001, pp. 1–15.
[76] R. Stone, CenterTrack: An IP Overlay Network for Tracking DoS Floods, in: Proceedings of the 9th USENIX Security Symposium, Denver, CO, August 14–17, 2000, pp. 199–212.
[77] National Institute of Standards and Technology, A Conceptual framework for system fault tolerance, 1995, Available from <http://hissa.nist.gov/chissa/SEI_Framework/framework_1.html>.
[78] W. Zhao, D. Olshefski, H. Schulzrinne, Internet Quality of Service: an overview, Columbia Technical Report CUCS-003-00, 2000.
[79] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, An architecture for differentiated services, in: IETF, RFC 2475, 1998.
[80] M.B. Geoffrey, G. Xie, A feeedback mechanism for mitigating Denial of Service attacks against differentiated services clients, in: Proceedings of the 10th International Conference on Telecommunications systems, Monterey, CA, October 2002, pp. 204–213.
[81] F. Kargl, J. Maier, M. Weber, Protecting web servers from Distributed Denial of Service attacks, in: Proceedings of the Tenth International Conference on World Wide Web, Hong Kong, May 1–5, 2001, pp. 514–524.
[82] J. Brustoloni, Protecting electronic commerce from Distributed Denial of Service attacks, in: Proceedings of the 11th International World Wide Web Conference, ACM, Honolulu, HI, 2002, pp. 553–561.
[83] S.M. Khattab, C. Sangpachatanaruk, R. Melhem, D. Mosse, T. Znati, Proactive server roaming for mitigating Denial of Service attacks, in: Proceedings of the 1st International Conference on International Technology: Research and Education (ITRE 03), Newark, NJ, August 2003, pp. 500–504.
[84] A. Garg, A.L.N. Reddy, Mitigating Denial of service Attacks using QoS regulation, in: Proceedings of the Tenth IEEE International Workshop on Quality of Service, 2002, pp. 45–53.
[85] A. Juels, J. Brainard, Client puzzles: a cryptographic countermeasure against connection depletion attacks, in: Proceedings of NDSS '99 (Networks and Distributed Security Systems), San Diego, CA, USA, February 1999, Internet Society, pp. 151–165.
[86] T. Aura, P. Nikander, J. Leiwo, DoS-resistant authentication with client puzzles, in: Proceedings of the 8th International Workshop on Security Protocols, Springer, New York, 2000, pp. 170–177.
[87] S.M. Mankins, C. Sangpachatanaruk, T. Znati, R. Melhem, D. Moss, Proactive server roaming for mitigating Denial of Service attacks, in: Proceedings of 1st International Conference on Information Technology Research and Education (ITRE), Newark, NJ, USA, August 10–13, 2003.
[88] J. Yan, S. Early, R. Anderson, The XenoService—a distributed defeat for Distributed Denial of Service, in: Proceedings of ISW 2000, in: Proceedings of ISW 2000, IEEE Computer Society, Boston USA, 2000.
[89] R. Mahajan, S. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, S. Shenker, Aggregate-based congestion control, ICSI Center for Internet Research (ICIR) AT&T Labs—Research.
[90] D. Sterne, K. Djahandari, B. Wilson, B. Babson, D. Schnackenberg, H. Holliday, T. Reid, Autonomic response to Distributed Denial of Service attacks, in: Proceedings of Recent Advances in Intrusion Detection, 4th International Symposium, RAID 2001 Davis, CA, USA, October 10–12, 2001, pp. 134–149.
[91] D.K. Yau, J.C.S. Lui, F. Liang, Defending Against Distributed Denial of Service attacks with max–min fair server-centric router throttles, in: Proceedings of the Tenth IEEE International Workshop on Quality of Service (IWQoS), Miami Beach, FL, 2002, pp. 35–44.
[92] P.A. Porras, P.G. Neumann, EMERALD: event monitoring enabling responses to anomalous live disturbances, in: Proceedings of the Nineteenth National Computer Security Conference, Baltimore, MD, October 22–25, 1997, pp. 353–365.
[93] Y.L. Zheng, J. Leiwo, A method to implement a Denial of Service protection base, in: Information Security and Privacy, Lecture Notes in Computer Science, vol. 1270, Springer, Berlin, 1997, pp. 90–101.
[94] O. Spatscheck, L. Peterson, Defending against Denial of Service requests in Scout, in: Proceedings of the 3rd USENIX/ACM Symposium on Operating System Design and Implementation, New Orleans, LA, 1999, pp. 59–72.
[95] F. Lau, S.H. Rubin, M.H. Smith, Lj. Trajkovic, Distributed Denial of Service attacks, in: Proceedings of 2000 IEEE International Conference on Systems, Man, and Cybernetics, Nashville, Nashville, TN, 2000, pp. 2275–2280.
[96] J. Leiwo, P. Nikander, T. Aura, Towards network denial of service resistant protocols, in: Proceedings of the 15th International Information Security Conference (IFIP/SEC 2000), Beijing, China, Kluwer, Dordrecht, 2000.
[97] C. Meadows, A formal framework and evaluation method for network Denial of Service, in: Proceedings of the 12th IEEE Computer Security Foundations Workshop, IEEE

Computer Society Press, Silver Spring, MD, 1999, pp. 4–13.

[98] C. Schuba, I. Krsul, M. Kuhn, G. Spafford, A. Sundaram, D. Zamboni, Analysis of a Denial of Service attack on TCP, in: Proceedings of IEEE Security and Privacy Symposium, Oakland, CA, USA, May 4–7, 1997, IEEE Computer Society, Silver Spring, MD, 1997, pp. 208–223.

[99] K.A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, R.A. Olsson, Detecting disruptive routers: a distributed network monitoring approach, in: Proceedings of the 1998 IEEE Symposium on Security and Privacy, Oakland, CA, IEEE Press, New York, 1998, pp. 115–124.

[100] S. Floyd, S. Bellovin, J. Ioannidis, K. Kompella, R. Mahajan, V. Paxson, Pushback messages for controlling aggregates in the network, Internet Draft, Work in progress, 2001.

[101] Mananet, Reverse Firewall, Available from <http://www.cs3-inc.com/ps_rfw.html>.

[102] X. Geng, Y. Huang, A.B. Whinston, Defending wireless infrastructure against the challenge of DDoS attacks, Mobile Networks and Applications 7 (3) (2002) 213–223.

Christos Douligeris received the Diploma in Electrical Engineering from the National Technical University of Athens in 1984 and the M.S., M.Phil. and Ph.D. degrees from Columbia University in 1985, 1987, 1990, respectively. He has held positions with the Department of Electrical and Computer Engineering at the University of Miami, where he reached the rank of associate professor and was the associate director for engineering of the Ocean Pollution Research Center. He is currently teaching at the Department of Informatics of the University of Piraeus, Greece. He has served in technical program committees of several conferences. His main technical interests lie in the areas of performance evaluation of high speed networks, neurocomputing in networking, resource allocation in wireless networks and information management, risk assessment and evaluation for emergency response operations. He was the guest editor of a special issue of the IEEE Communications Magazine on "Security for Telecommunication Networks" and he is preparing a book on "Network Security" to be published by IEEE Press/Wiley. He is an editor of the IEEE Communications Letters, a technical editor of IEEE Network, a technical editor of Computer Networks (Elsevier), and a technical editor of the IEEE Communications Magazine Interactive.

Aikaterini Mitrokotsa received the Bachelor of Science in Informatics from the University of Piraeus in 2001. She is currently a doctoral student at the Department of Informatics of the University of Piraeus. Her research interest lie in the areas of network security, denial of service attacks and performance evaluation of computer networks.