A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms

Jelena Mirkovic, Janice Martin and Peter Reiher
Computer Science Department
University of California, Los Angeles
Technical report #020018

Abstract

This paper proposes a taxonomy of distributed denial-of-service attacks and a taxonomy of the defense mechanisms that strive to counter these attacks. The attack taxonomy is illustrated using both known and potential attack mechanisms. Along with this classification we discuss important features of each attack category that in turn define the challenges involved in combating these threats. The defense system taxonomy is illustrated using only the currently known approaches. The goal of the paper is to impose some order into the multitude of existing attack and defense mechanisms that would lead to a better understanding of challenges in the distributed denial-of-service field.

1. Introduction

Distributed denial-of-service attacks (DDoS) pose an immense threat to the Internet, and consequently many defense mechanisms have been proposed to combat them. Attackers constantly modify their tools to bypass these security systems, and researchers in turn modify their approaches to handle new attacks. The DDoS field is evolving quickly, and it is becoming increasingly hard to grasp a global view of the problem. This paper strives to introduce some structure to the DDoS field by developing a taxonomy of DDoS attacks and DDoS defense systems. The goal of the paper is to highlight the important features of both attack and security mechanisms and stimulate discussions that might lead to a better understanding of the DDoS problem.

The proposed taxonomies are complete in the following sense: the attack taxonomy covers known attacks and also those that have not currently appeared but are potential threats that would affect current defense mechanisms; the defense systems taxonomy covers not only published approaches but also some commercial approaches that are sufficiently documented to be analyzed. Along with classification, we emphasize important features of each attack or defense system category, and provide representative examples of existing mechanisms. This paper does not propose or advocate any specific DDoS defense mechanism. Even though some sections might point out vulnerabilities of certain classes of defense systems, our purpose is not to criticize but to draw attention to these problems so that they might be solved.

Following this introduction, the paper is organized as follows. Section 2 investigates the problem of DDoS attacks, and Section 3 proposes their taxonomy; Section 4 proposes a taxonomy of DDoS defense systems. Section 5 provides an overview of related work and Section 6 concludes the paper.

2. DDoS Attack Overview

A denial-of-service attack is characterized by an explicit attempt by attackers to prevent legitimate users of a service from using that service [1]. A distributed denial-of-service attack deploys multiple machines to attain this goal. The service is denied by sending a stream of packets to a victim that either consumes some key resource, thus rendering it unavailable to legitimate clients, or provides the attacker with unlimited access to the victim machine so he can inflict arbitrary damage. This section will answer the following questions:

1. What makes DDoS attacks possible?
2. How do these attacks occur?
3. Why do they occur?

2.1. Internet Architecture

The Internet was designed with functionality, not security, in mind, and it was indeed very successful in reaching this goal. It offers its participants fast, easy and cheap communication mechanisms, enforced with various higher-level protocols that ensure reliable or timely delivery of messages or a certain level of quality of service. Internet design follows the end-to-end paradigm: communicating end hosts deploy complex functionalities to achieve desired service guarantees, while the intermediate network provides the bare-minimum, best-effort service. The Internet is managed in a distributed manner; therefore no common policy can be enforced among its participants. Such design opens several security issues that provide opportunities for distributed denial-of-service attacks:

1. *Internet security is highly interdependent*. DDoS attacks are commonly launched from systems that are subverted through security-related compromises. Regardless of how well secured the victim system may be, its susceptibility to DDoS attacks depends on the state of security in the rest of the global Internet [2].
2. *Internet resources are limited*. Each Internet host has limited resources that can be consumed by a sufficient number of users.
3. *Power of many is greater than power of few*. Coordinated and simultaneous malicious actions by some participants can always be detrimental to others, if the resources of the attackers are greater than the resources of the victims.
4. *Intelligence and resources are not collocated*. An end-to-end communication paradigm led to locating most of the intelligence needed for service guarantees with end hosts. At the same time, a desire for large throughput led to the design of high bandwidth pathways in the intermediate network. Thus, malicious clients can misuse the abundant resources of unwitting network for delivery of numerous messages to a victim.

2.2. DDoS Attack Strategy

In order to perform a distributed denial-of-service attack, the attacker needs to *recruit* the multiple agent (slave) machines. This process is usually performed automatically through scanning of remote machines, looking for security holes that would enable subversion. Vulnerable machines are then *exploited* by using the discovered vulnerability to gain access to the machine, and they are *infected* with the attack code. The exploit/infection phase is also automated, and the infected machines can be used for further recruitment of new agents (see discussion of propagation techniques in Section 3.1 and in [6]). Agent machines perform the attack against the victim. Attackers usually hide the identity of the agent machines during the attack through spoofing of the source address field in packets. The agent machines can thus be reused for future attacks.

2.3. DDoS Goals

The goal of a DDoS attack is to inflict damage on the victim, either for personal reasons (a significant number of DDoS attacks are against home computers, presumably for purposes of revenge), for material gain (damaging competitor's resources) or for popularity (successful attacks on popular Web servers gain the respect of the hacker community).

3. Taxonomy of DDoS Attacks

In order to devise a taxonomy of distributed denial-of-service attacks we observe the *means* used to prepare and perform the attack, the *characteristics* of the attack itself and the *effect* it has on the victim. Various classification criteria are indicated in bold type. Figure 1 summarizes the taxonomy.

3.1. Classification by Degree of Automation

During the attack preparation, the attacker needs to locate prospective agent machines and infect them with the attack code. Based on the **degree of automation** of the attack, we differentiate between *manual*, *semi-automatic* and a*utomatic* DDoS attacks.

Manual Attacks

Only the early DDoS attacks belonged to the manual category. The attacker scanned remote machines for vulnerabilities, broke into them and installed the attack code, and then commanded the onset of the attack. All of these actions were soon automated, leading to development of semi-automatic DDoS attacks, the category where most contemporary attacks belong.

Semi-Automatic Attacks

In semi-automatic attacks, the DDoS network consists of handler (master) and agent (slave, daemon) machines. The attacker deploys automated scripts for scanning and compromise of those machines and installation of the attack code. He then uses handler machines to specify the attack type and the victim's address and to command the onset of the attack to agents, who send packets to the victim.

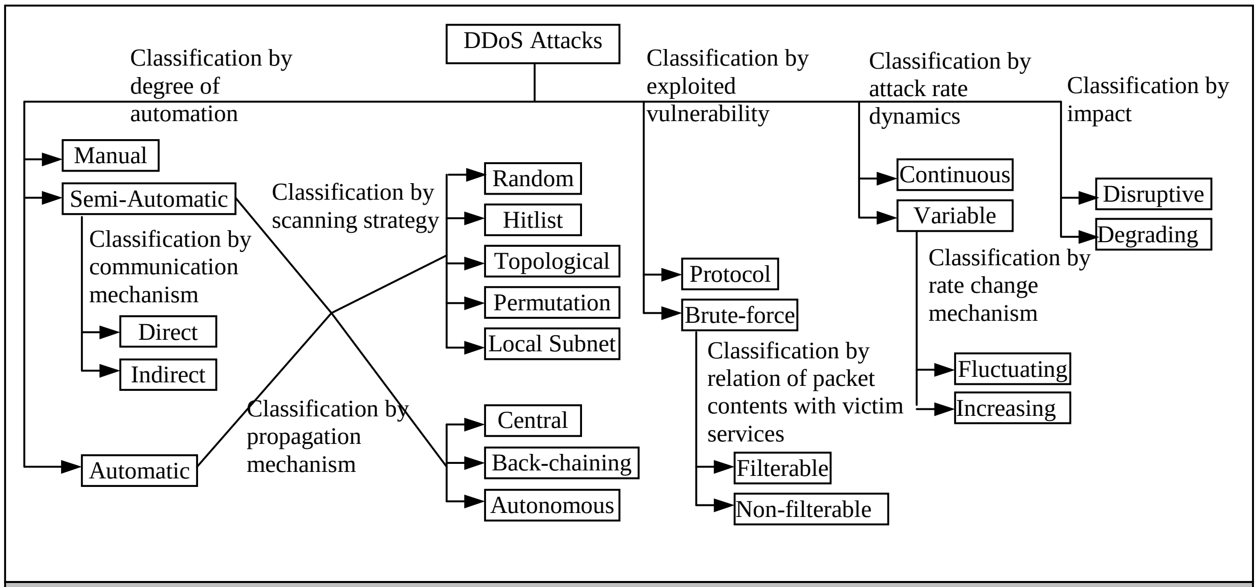
DDoS Attacks
Classification by degree of automation
Manual
Semi-Automatic
Classification by communication mechanism
Direct
Indirect
Automatic
Classification by scanning strategy
Random
Hitlist
Topological
Permutation
Local Subnet
Classification by propagation mechanism
Central
Back-chaining
Autonomous
Classification by exploited vulnerability
Protocol
Brute-force
Classification by relation of packet contents with victim services
Filterable
Non-filterable
Classification by attack rate dynamics
Continuous
Variable
Classification by rate change mechanism
Fluctuating
Increasing
Classification by impact
Disruptive
Degrading

Figure 1: Taxonomy of distributed denial-of-service attacks

Based on the **communication mechanism** deployed between agent and handler machines we divide semi-automatic attacks into *attacks with direct communication* and *attacks with indirect communication*.

Attacks with direct communication

During attacks with direct communication, the agent and handler machines need to know each other's identity in order to communicate. This is achieved by hard-coding the IP address of the handler machines in the attack code that is later installed on the agent. Each agent then reports its readiness to the handlers, who store its IP address in a file for later communication. The obvious drawback of this approach is that discovery of one compromised machine can expose the whole DDoS network. Also, since agents and handlers listen to network connections, they are identifiable by network scanners.

Attacks with indirect communication

Attacks with indirect communication deploy a level of indirection to increase the survivability of a DDoS network. Recent attacks provide the example of using IRC channels [2] for agent/handler communication. The use of IRC services replaces the function of a handler, since the IRC channel offers sufficient anonymity to the attacker. Since DDoS agents establish outbound connections to a standard service port used by a legitimate network service, agent communications to the control point may not be easily differentiated from legitimate network traffic. The agents do not incorporate a listening port that is easily detectable with network scanners. An attacker controls the agents using IRC communications channels. Thus, discovery of a single agent may lead no further than the identification of one or more IRC servers and channel names used by the DDoS network. From there, identification of the DDoS network depends on the ability to track agents currently connected to the IRC server. Although the IRC service is the only current example of indirect communication, there is nothing to prevent attackers from subverting other legitimate services for similar purposes.

Automatic Attacks

Automatic DDoS attacks additionally automate the attack phase, thus avoiding the need for communication between attacker and agent machines. The time of the onset of the attack, attack type, duration and victim's address is preprogrammed in the attack code. It is obvious that such deployment mechanisms offer minimal exposure to the attacker, since he is only involved in issuing a single command – the start of the attack script. The hardcoded attack specification suggests a single-purpose use of the DDoS network. However, the propagation mechanisms usually leave the backdoor to the compromised machine open, enabling easy future access and modification of the attack code.

Both semi-automatic and automatic attacks recruit the agent machines by deploying automatic scanning and propagation techniques. Based on the **scanning strategy**, we differentiate between attacks that deploy *random scanning, hitlist scanning*, *topological scanning*, *permutation scanning* and *local subnet scanning*. We give a brief description of these scanning techniques here and refer the reader to [6] for a detailed description and performance comparison. Attackers usually combine the scanning and exploitation phases, thus gaining a larger agent population, and our description of scanning techniques relates to this model.

Attacks with Random Scanning

During random scanning each compromised host probes random addresses in the IP address space, using a different seed. This potentially creates a high traffic volume since many machines probe the same addresses. Code Red (CRv2) performed random scanning [7].

Attacks with Hitlist Scanning

A machine performing hitlist scanning probes all addresses from an externally supplied list. When it detects the vulnerable machine, it sends one half of the initial hitlist to the recipient and keeps the other half. This technique allows for great propagation speed (due to exponential spread) and no collisions during the scanning phase. An attack deploying hitlist scanning could obtain the list from netscan.org of domains that still support directed IP broadcast and can thus be used for a Smurf attack.

Attacks with Topological Scanning

Topological scanning uses the information on the compromised host to select new targets. All E-mail worms use topological scanning, exploiting the information from address books for their spread.

Attacks with Permutation Scanning

During permutation scanning, all compromised machines share a common pseudo-random permutation of the IP address space; each IP address is mapped to an index in this permutation. A machine begins scanning by using the index computed from its IP address as a starting point. Whenever it sees an already infected machine, it chooses a new random start point. This has the effect of providing a semi-coordinated, comprehensive scan while maintaining the benefits of random probing. This technique is described in [6] as not yet deployed.

Attacks with Local Subnet Scanning

Local subnet scanning can be added to any of the previously described techniques to preferentially scan for targets that reside on the same subnet as the compromised host. Using this technique, a single copy of the scanning program can compromise many vulnerable machines behind a firewall. Code Red II [8] and Nimda Worm [9] used local subnet scanning.

Based on the attack code **propagation mechanism**, we differentiate between attacks that deploy *central source propagation, back-chaining propagation* and *autonomous propagation* [2].

Attacks with Central Source Propagation

During central source propagation, the attack code resides on a central server or set of servers. After compromise of the agent machine, the code is downloaded from the central source through a file transfer mechanism. The 1i0n [4] worm operated in this manner.

Attacks with Back-chaining Propagation

During back-chaining propagation, the attack code is downloaded from the machine that was used to exploit the system. The infected machine then becomes the source for the next propagation step. Back-chaining propagation is more survivable than central-source propagation since it avoids a single point of failure. The Ramen worm [5] and Morris Worm [19] used back-chaining propagation.

Attacks with Autonomous Propagation

Autonomous propagation avoids the file retrieval step by injecting attack instructions directly into the target host during the exploitation phase. Code Red [3], Warhol Worm [6] and numerous E-mail worms use autonomous propagation.

3.2. Classification by Exploited Vulnerability

Distributed denial-of-service attacks exploit different strategies to deny the service of the victim to its clients. Based on the **vulnerability that is**

targeted during an attack, we differentiate between *protocol attacks* and *brute-force attacks*.

Protocol Attacks

Protocol attacks exploit a specific feature or implementation bug of some protocol installed at the victim in order to consume excess amounts of its resources. Examples include the TCP SYN attack, the CGI request attack and the authentication server attack.

In the TCP SYN attack, the exploited feature is the allocation of substantial space in a connection queue immediately upon receipt of a TCP SYN request. The attacker initiates multiple connections that are never completed, thus filling up the connection queue indefinitely. In the CGI request attack, the attacker consumes the CPU time of the victim by issuing multiple CGI requests. In the authentication server attack, the attacker exploits the fact that the signature verification process consumes significantly more resources than bogus signature generation. He sends numerous bogus authentication requests to the server, tying up its resources.

Brute-force Attacks

Brute-force attacks are performed by initiating a vast amount of seemingly legitimate transactions. Since an upstream network can usually deliver higher traffic volume than the victim network can handle, this exhausts the victim's resources.

We further divide brute-force attacks based on the **relation of packet contents with victim services** into *filterable* and *non-filterable* attacks.

Filterable Attacks

Filterable attacks use bogus packets or packets for non-critical services of the victim's operation, and thus can be filtered by a firewall. Examples of such attacks are a UDP flood attack or an ICMP request flood attack on a Web server.

Non-filterable Attacks

Non-filterable attacks use packets that request legitimate services from the victim. Thus, filtering all packets that match the attack signature would lead to an immediate denial of the specified service to both attackers and the legitimate clients. Examples are a HTTP request flood targeting a Web server or a DNS request flood targeting a name server.

The line between protocol and brute force attacks is thin. Protocol attacks also overwhelm a victim's resources with excess traffic, and badly designed protocol features at remote hosts are frequently used to perform "reflector" brute-force attacks, such as the DNS request attack [10] or the Smurf attack [11].

The difference is that a victim can mitigate the effect of protocol attacks by modifying the deployed protocols at its site, while it is helpless against brute-force attacks due to their misuse of legitimate services (non-filterable attacks) or due to its own limited resources (a victim can do nothing about an attack that swamps its network bandwidth).

Countering protocol attacks by modifying the deployed protocol pushes the corresponding attack mechanism into the brute-force category. For example, if the victim deploys TCP SYN cookies [55] to combat TCP SYN attacks, it will still be vulnerable to TCP SYN attacks that generate more requests than its network can accommodate. However, the brute-force attacks need to generate a much higher volume of attack packets than protocol attacks, to inflict damage at the victim. So by modifying the deployed protocols the victim pushes the vulnerability limit higher. Evidently, classification of the specific attack needs to take into account both the attack mechanisms used and the victim's configuration.

It is interesting to note that the variability of attack packet contents is determined by the exploited vulnerability. Packets comprising protocol and non-filterable brute force attacks must specify some valid header fields and possibly some valid contents. For example TCP SYN attack packets cannot vary the protocol or flag field, and HTTP flood packets must belong to an established TCP connection and therefore cannot spoof source addresses, unless they hijack connections from legitimate clients.

3.3. Classification by Attack Rate Dynamics

Depending on **the attack rate dynamics** we differentiate between *continuous rate* and *variable rate* attacks.

Continuous Rate Attacks

The majority of known attacks deploy a continuous rate mechanism. After the onset is commanded, agent machines generate the attack packets with full force. This sudden packet flood disrupts the victim's services quickly, and thus leads to attack detection.

Variable Rate Attacks

Variable rate attacks are more cautious in their engagement, and they vary the attack rate to avoid detection and response.

Based on the **rate change mechanism** we differentiate between attacks with *increasing rate* and *fluctuating rate*.

Increasing Rate Attacks

Attacks that have a gradually increasing rate lead to a slow exhaustion of victim's resources. A state change of the victim could be so gradual that its services degrade slowly over a long time period, thus delaying detection of the attack.

Fluctuating Rate Attacks

Attacks that have a fluctuating rate adjust the attack rate based on the victim's behavior, occasionally relieving the effect to avoid detection. At the extreme end, there is the example of *pulsing attacks*. During pulsing attacks, agent hosts periodically abort the attack and resume it at a later time. If this behavior is simultaneous for all agents, the victim experiences periodic service disruptions. If, however, agents are divided into groups who coordinate so that one group is always active, then the victim experiences continuous denial of service.

3.4. Classification by Impact

Depending on the **impact** of a DDoS attack on the victim we differentiate between *disruptive* and *degrading* attacks.

Disruptive Attacks

The goal of disruptive attacks is to completely deny the victim's service to its clients. All currently known attacks belong to this category.

Degrading Attacks

The goal of degrading attacks would be to consume some (presumably constant) portion of a victim's resources. Since these attacks do not lead to total service disruption, they could remain undetected for a significant time period. On the other hand, damage inflicted on the victim could be immense. For example, an attack that effectively ties up 30% of the victim's resources would lead to denial of service to some percentage of customers during high load periods, and possibly slower average service. Some customers, dissatisfied with the quality, would consequently change their service provider and victim would thus lose income. Alternately, the false load could result in a victim spending money to upgrade its servers and networks.

4. Taxonomy of DDoS Defense Mechanisms

The seriousness of the DDoS problem and the increased frequency of DDoS attacks have led to the advent of numerous DDoS defense mechanisms. Some of these mechanisms address a specific kind of DDoS attack such as attacks on Web servers or authentication servers. Other approaches attempt to solve the entire generic DDoS problem. Most of the proposed approaches require certain features to achieve their peak performance, and will perform quite differently if deployed in an environment where these requirements are not met.

As is frequently pointed out, there is no "silver bullet" against DDoS attacks. Therefore we need to understand not only each existing DDoS defense approach, but also how those approaches might be combined together to effectively and completely solve the problem. The proposed taxonomy, shown in Figure 2, should help us reach this goal. Various classification criteria are indicated in bold type.

4.1. Classification by Activity Level

Based on the **activity level** of DDoS defense mechanisms, we differentiate between *preventive* and *reactive* mechanisms.

Preventive Mechanisms

The goal of preventive mechanisms is either to eliminate the possibility of DDoS attacks altogether or to enable potential victims to endure the attack without denying services to legitimate clients. According to these **goals** we further divide preventive mechanisms into *attack prevention* and *denial-of-service prevention* mechanisms.

Attack Prevention Mechanisms

Attack prevention mechanisms modify the system configuration to eliminate the possibility of a DDoS attack. Based on the **target** they

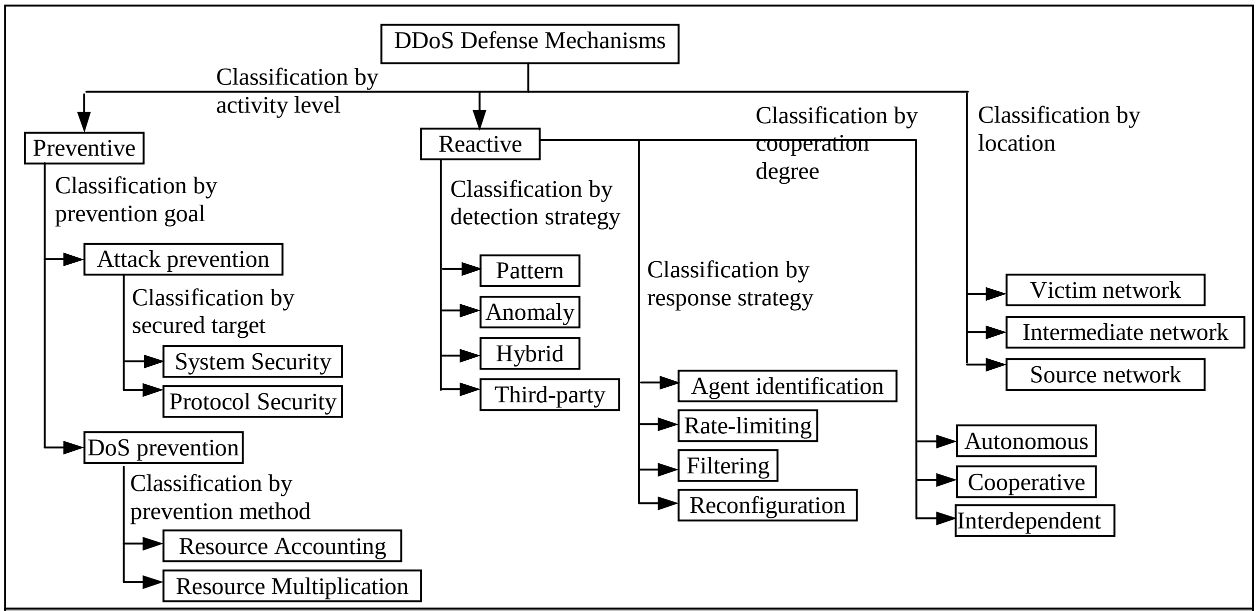

Figure 2: Taxonomy of distributed denial-of-service defense mechanisms

secure, we further divide them into *system security* and *protocol security* mechanisms.

System Security Mechanisms

System security mechanisms increase the overall security of the system, guarding against illegitimate accesses to the machine, removing application bugs and updating protocol installations to prevent intrusions and misuse of the system. DDoS attacks owe their power to large numbers of subverted machines that cooperatively generate the attack streams. If these machines were secured, the attackers would lose their army and the DDoS threat would then disappear. On the other hand, systems vulnerable to intrusions can themselves become victims of DDoS attacks in which the attacker, having gained unlimited access to the machine, deletes or alters its contents. Potential victims of DDoS attacks can be easily overwhelmed if they deploy vulnerable protocols. Examples of system security mechanisms include monitored access to the machine [20], applications that download and install security patches, firewall systems [43], virus scanners [44], intrusion detection systems [17], access lists for critical resources [46], capability-based systems [53] and client-legitimacy-based systems [54]. The history of computer security suggests that this approach can never be 100% effective, but doing a good job here will certainly decrease the frequency and strength of DDoS attacks.

Protocol Security Mechanisms

Protocol security mechanisms address the problem of bad protocol design. Many protocols contain operations that are cheap for the client but expensive for the server. Such protocols can be misused to exhaust the resources of a server by initiating large numbers of simultaneous transactions. Classic misuse examples are the TCP SYN attack, the authentication server attack, and the fragmented packet attack, in which the attacker bombards the victim with malformed packet fragments forcing it to waste its resources on reassembling attempts. Examples of protocol security mechanisms include guidelines for a safe protocol design in which resources are committed to the client only after sufficient authentication is done ([28], [12]), or the client has paid a sufficient price [31], deployment of powerful proxy server that completes TCP connections [29], etc.

Deploying comprehensive protocol and system security mechanisms can make the victim completely resilient to protocol attacks. Also, these approaches are inherently compatible with and complementary to all other approaches.

Denial-of-Service Prevention Mechanisms

Denial-of-service prevention mechanisms enable the victim to endure attack attempts without denying service to legitimate clients. This is done either by enforcing policies for resource consumption or by ensuring that abundant resources exist so that legitimate clients will not be affected by the attack. Consequently, based on the **prevention method**, we differentiate between *resource accounting* and *resource multiplication* mechanisms.

Resource Accounting Mechanisms

Resource accounting mechanisms police the access of each user to resources based on the privileges of the user and his behavior. Such mechanisms guarantee fair service to legitimate well-behaving users. In order to avoid user identity theft, they are usually coupled with legitimacy-based access mechanisms that verify the user's identity. Approaches proposed in ([27], [30], [32], [33], [34]) illustrate resource accounting mechanisms.

Resource Multiplication Mechanisms

Resource multiplication mechanisms provide an abundance of resources to counter DDoS threats. The straightforward example is a system that deploys a pool of servers with a load balancer and installs high bandwidth links between itself and upstream routers. This approach essentially raises the bar on how many machines must participate in an attack to be effective. While not providing perfect protection, for those who can afford the costs, this approach has often proven sufficient. For example, Microsoft has used it to weather large DDoS attacks.

Reactive Mechanisms

Reactive mechanisms strive to alleviate the impact of an attack on the victim. In order to attain this goal they need to *detect* the attack and *respond* to it.

The goal of attack detection is to detect every attempted DDoS attack as early as possible and to have a low degree of false positives. Upon attack detection, steps can be taken to characterize the packets belonging to the attack stream and provide this characterization to the response mechanism.

We classify reactive mechanisms based on the **attack detection strategy** into mechanisms that deploy *pattern detection*, *anomaly detection*, *hybrid detection*, and *third-party detection*.

Mechanisms with Pattern Attack Detection

Mechanisms that deploy pattern detection store the signatures of known attacks in a database. Each communication is monitored and compared with database entries to discover occurrences of DDoS attacks. Occasionally, the database is updated with new attack signatures. The obvious drawback of this detection mechanism is that it can only detect known attacks, and it is usually helpless against new attacks or even slight variations of old attacks that cannot be matched to the stored signature. On the other hand, known attacks are easily and reliably detected, and no false positives are encountered.

Mechanisms with Anomaly Attack Detection

Mechanisms that deploy anomaly detection have a model of normal system behavior, such as a model of normal traffic dynamics or expected system performance. The current state of the system is periodically compared with the models to detect anomalies. Approaches presented in ([35], [36], [38], [39], [45], [48], [50], [51], [52]) provide examples of mechanisms that use anomaly detection.

The advantage of anomaly detection over pattern detection is that unknown attacks can be discovered. However, anomaly-based detection has to address two issues:

1. *Threshold setting.* Anomalies are detected when the current system state differs from the model by a certain threshold. The setting of a low threshold leads to many false positives, while a high threshold reduces the sensitivity of the detection mechanism.
2. *Model update.* Systems and communication patterns evolve with time, and models need to be updated to reflect this change. Anomaly-based systems usually perform automatic model update using statistics gathered at a time when no attack was detected. This approach makes the detection mechanism vulnerable to increasing rate attacks that can mistrain models and delay or even avoid attack detection.

Mechanisms with Hybrid Attack Detection

Mechanisms that deploy hybrid detection combine the pattern-based and anomaly-based detection, using data about attacks discovered through an anomaly detection mechanism to devise new attack signatures and update the database. Many intrusion detection systems use hybrid detection.

If these systems are fully automated, properly extracting a signature from a detected attack can be challenging. The system must be careful not to permit attackers to fool it into detecting normal behavior as an attack signature, or the system itself becomes a denial-of-service tool.

Mechanisms with Third-Party Attack Detection

Mechanisms that deploy third-party detection do not handle the detection process themselves, but rely on an external message that signals the occurrence of the attack and provides attack characterization. Examples of mechanisms that use third-party detection are easily found among traceback mechanisms ([21], [23], [24], [26], [41]).

The goal of the attack response is to relieve the impact of the attack on the victim, while imposing minimal collateral damage to legitimate clients of the victim. We classify reactive mechanisms based on the **response strategy** into mechanisms that deploy *agent identification*, *rate-limiting*, *filtering* and *reconfiguration* approaches.

Agent Identification Mechanisms

Agent identification mechanisms provide the victim with information about the identity of the machines that are performing the attack. This information can then be combined with other response approaches to alleviate the impact of the attack. Agent identification examples include numerous traceback techniques ([21], [23], [24], [26], [41]) and approaches that eliminate spoofing ([25], [40]), thus enabling use of the source address field for agent identification.

Rate-Limiting Mechanisms

Rate-limiting mechanisms impose a rate limit on a stream that has been characterized as malicious by the detection mechanism. Examples of rate-limiting mechanisms are found in ([36], [39], [45], [50]). Rate limiting is a lenient response technique that is usually deployed when the detection mechanism has a high level of false positives or cannot precisely characterize the attack stream. The disadvantage is that they allow some attack traffic through, so extremely high scale attacks might still be effective even if all traffic streams are rate-limited.

Filtering Mechanisms

Filtering mechanisms use the characterization provided by a detection mechanism to filter out the attack stream completely. Examples include dynamically deployed firewalls [22], and also a commercial system TrafficMaster [48]. Unless detection strategy is very reliable, filtering mechanisms run the risk of accidentally denying service to legitimate traffic. Worse, clever attackers might leverage them as denial-of-service tools.

Reconfiguration Mechanisms

Reconfiguration mechanisms change the topology of the victim or the intermediate network to either add more resources to the victim or to isolate the attack machines. Examples include reconfigurable overlay networks ([37], [38]), resource replication services [35], attack isolation strategies ([49], [51]), etc.

Reactive DDoS defense mechanisms can perform detection and response either alone or in cooperation with other entities in the Internet. Based on the **cooperation degree** we differentiate between *autonomous*, *cooperative* and *interdependent* mechanisms.

Autonomous Mechanisms

Autonomous mechanisms perform independent attack detection and response. They are usually deployed at a single point in the Internet and act locally. Firewalls and intrusion detection systems provide an easy example of autonomous mechanisms.

Cooperative Mechanisms

Cooperative mechanisms are capable of autonomous detection and response, but can achieve significantly better performance through cooperation with other entities. Mechanisms deploying pushback [36] provide examples of cooperative mechanisms. They detect the occurrence of a DDoS attack by observing congestion in a router's buffer, characterize the traffic that creates the congestion, and act locally to impose a rate limit on that traffic. However, they achieve significantly better performance if the rate limit requests can be propagated to upstream routers who otherwise may be unaware of the attack.

Interdependent Mechanisms

Interdependent mechanisms cannot operate autonomously; they rely on other entities either for attack detection or for efficient response. Traceback mechanisms ([21], [23], [24], [26], [41]) provide examples of interdependent mechanisms. A traceback mechanism deployed on a single router would provide almost no benefit.

4.2. Classification by Deployment Location

With regard to a **deployment location**, we differentiate between DDoS mechanisms deployed at the victim, intermediate, or source network.

Victim-Network Mechanisms

DDoS defense mechanisms deployed at the victim network protect this network from DDoS attacks and respond to detected attacks by alleviating the impact on the victim. Historically, most defense systems were located at the victim since it suffered the greatest impact of the attack and was therefore the most motivated to sacrifice some resources for increased security. Resource accounting ([27], [30], [32], [33], [34]) and protocol security mechanisms ([28], [12], [31], [29]) provide examples of these systems.

Intermediate-Network Mechanisms

DDoS defense mechanisms deployed at the intermediate network provide infrastructural service to a large number of Internet hosts. Victims of DDoS attacks can contact the infrastructure and request the service, possibly providing adequate compensation. Pushback [36] and traceback ([21], [23], [24], [26], [41]) techniques are examples of intermediate-network mechanisms.

Source-Network Mechanisms

The goal of DDoS defense mechanisms deployed at the source network is to prevent customers using this network from generating DDoS attacks. Such mechanisms are necessary and desirable, but motivation for their deployment is low since it is unclear who would pay the expenses associated with this service. Mechanisms proposed in ([39], [45]) provide examples of source-network mechanisms.

5. Related Work

Although distributed denial-of-service attacks have been recognized as a serious problem, we are not aware of any other attempt to introduce formal classification into the DDoS attack mechanisms. The reason might lay in the use of fairly simple attack tools that have dominated most DDoS incidents. Those tools performed full-force flooding attacks using several types of packets. As defense mechanisms are deployed to counter these simple attacks, we expect to be faced with more complex strategies.

In [15] authors present classification of denial-of-service attacks according to the type of the target (e.g. firewall, Web server, router), a resource that the attack consumes (network bandwidth, TCP/IP stack) and the exploited vulnerability (bug or overload). This classification focuses more on the actual attack phase, while we are interested in looking at the complete attack mechanism in order to highlight features that are specific to distributed attacks.

In [14] and [16] Howard proposes a taxonomy of computer and network attacks. This taxonomy focuses on computer attacks in general and does not sufficiently highlight features particular to distributed denial-of-service attacks.

CERT is currently undertaking the initiative to devise a comprehensive taxonomy of computer incidents as part of the design of common incident data format and exchange procedures, but unfortunately their results are not yet available.

We are not aware of any attempt to formally classify DDoS defense systems, although similar works exist in field of intrusion detection systems ([16], [18]) and offer informative reading for researchers in the DDoS defense field.

6. Conclusion

Distributed denial of service attacks are a complex and serious problem, and consequently, numerous approaches have been proposed to counter them. The multitude of current attack and defense mechanisms obscures the global view of the DDoS problem. This paper is a first attempt to cut through the obscurity and achieve a clear view of the problem and its solutions. The taxonomies described here are intended to help the community think about the threats we face and the measures we can use to counter those threats.

One positive benefit we foresee from development of DDoS taxonomies is to foster easier cooperation among researchers on DDoS defense mechanisms. Attackers cooperate to exchange attack code and information about vulnerable machines, and to organize their agents into coordinated networks to achieve immense power and survivability. The Internet community must be equally cooperative among itself to counter this threat. Good taxonomies for DDoS attack and defense mechanisms will facilitate communications and offer the community a common language to discuss their solutions. They will also clarify how different mechanisms are likely to work in concert, and identify areas of remaining weakness that require additional mechanisms. Similarly, the research community needs to develop common metrics and benchmarks to evaluate the efficacy of DDoS defense mechanisms, and these taxonomies can be helpful in shaping these tasks, as well.

We do not claim that these taxonomies are complete and all-encompassing. We must not be deceived by the simplicity of the current attacks; for the attackers this simplicity arises more from convenience than necessity. As defense mechanisms are deployed to counter simple attacks, we are likely to see more complex attack scenarios. Many more attack possibilities exist and must be addressed before we can completely handle the DDoS threat, and some of them are likely to be outside the current boundaries of the taxonomies presented here. Thus, these taxonomies are likely to require expansion and refinement as new threats and defense mechanisms are discovered.

The DDoS attack taxonomy and DDoS defense taxonomy outlined in this paper are useful to the extent that they clarify our thinking and guide us to more effective solutions to the problem of distributed denial-of-service. The ultimate value of the work described here will thus be in the degree of discussion and future research that it provokes.

References

[1] CERT Coordination Center, "Denial of Service Attacks," http://www.cert.org/tech_tips/denial_of_service.html

[2] CERT Coordination Center, "Trends in Denial of Service Attack Technology," October 2001, http://www.cert.org/archive/pdf/DoS_trends.pdf

[3] CERT Coordination Center, "Code Red," http://www.cert.org/incident_notes/IN-2001-08.html

[4] CERT Coordination Center, "erkms and li0n worms," http://www.cert.org/incident_notes/IN-2001-03.html

[5] CERT Coordination Center, "Ramen worm," http://www.cert.org/incident_notes/IN-2001-01.html

[6] N. Weaver, "Warhol Worm," http://www.cs.berkeley.edu/~nweaver/warhol.html

[7] D. Moore, "The spread of the code red worm (crv2)," http://www.caida.org/analysis/security/code-red/coderedv2_analysis.xml.

[8] CERT Coordination Center, "Code Red II," http://www.cert.org/incident_notes/IN-2001-09.html

[9] CERT Coordination Center, "Nimda worm," http://www.cert.org/advisories/CA-2001-26.html

[10] CERT Coordination Center, "DoS using nameservers," http://www.cert.org/incident_notes/IN-2000-04.html

[11] CERT Coordination Center, "Smurf attack," http://www.cert.org/advisories/CA-1998-01.html

[12] C. Meadows, "A formal framework and evaluation method for network denial of service," In *Proceedings of the 12th IEEE Computer Security Foundations Workshop*, June 1999.

[13] Cisco, "Strategies to protect against Distributed Denial of Service Attacks," http://www.cisco.com/warp/public/707/newsflash.html

[14] J. D. Howard, "An analysis of security incidents on the Internet," PhD thesis, Carnegie Mellon University, August 1998.

[15] F. Kargl, J. Maier and M. Weber, "Protecting web servers from distributed denial of service attacks," In *Proceedings of 10th International World Wide Web Conference*, May 2001

[16] J. D. Howard and T. A. Longstaff, "A common language for computer security incidents," Sandia Report: SAND98-8667, Sandia National Laboratories, http://www.cert.org/research/taxonomy_988667.pdf

[17] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Technical Report 99-15, Department of Computer Engineering, Chalmers University, March 2000.

[18] H. Debar, M. Dacier, and A. Wespi, "Towards a taxonomy of intrusion-detection systems," *Computer Networks*, 31(8):805-822, April 1999.

[19] K. Hafner and J. Markoff, *Cyberpunk: Outlaws and hackers on the computer frontier,* Simon & Schuster, 1991.

[20] Tripwire, "Tripwire for servers," http://www.tripwire.com/products/servers/

[21] D. X. Song and A. Perrig, "Advanced and authenticated marking schemes for IP Traceback," IEEE Infocom 2001.

[22] T. Darmohray and R. Oliver, "Hot spares for DDoS attacks," http://www.usenix.org/publications/login/2000-7/apropos.html.

[23] D. Dean, M. Franklin and A. Stubblefield, "An algebraic approach to IP Traceback," In *Proceedings of the 2001 Network and Distributed System Security Symposium*, February 2001.

[24] S. M. Bellovin, "ICMP traceback messages," Internet draft, http://search.ietf.org/internet-drafts/draft-ietf-itrace-01.txt, Oct. 2001.

[25] P. Ferguson and D. Senie, "Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing," RFC 2267, January 1998

[26] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP Traceback," In *Proceedings of 2000 ACM SIGCOMM Conference*, Aug. 2000.

[27] A. Juels and J. Brainard, "Client puzzles: A cryptographic countermeasure against connection depletion attacks," In

Proceedings of the 1999 Networks and distributed system security symposium (NDSS'99), Mar 1999.

[28] J. Leiwo, P. Nikander, and T. Aura, "Towards network denial of service resistant protocols," In *Proceedings of the 15th International Information Security Conference (IFIP/SEC 2000)*, August 2000.

[29] C. Schuba, I. Krsul, M. Kuhn, G. Spafford, A. Sundaram, and D. Zamboni, "Analysis of a denial of service attack on TCP," In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, May 1997.

[30] Y. L. Zheng and J. Leiwo, "A method to implement a denial of service protection base," In *Information Security and Privacy*, volume 1270 of LNCS, pages 90--101, 1997.

[31] T. Aura, P. Nikander, and J. Leiwo, "DOS-resistant authentication with client puzzles," In *Proceedings of the 8th International Workshop on Security Protocols*

[32] O. Spatscheck and L. Peterson, "Defending against denial-of service requests in Scout," In *Proceedings of the 1999 USENIX/ACM Symposium on Operating System Design and Implementatio*n, February 1999.

[33] A. Garg and A. L. Narasimha Reddy, "Mitigating denial of service attacks using QoS regulation," Texas A & M University Tech report, TAMU-ECE-2001-06

[34] F. Lau, S. H. Rubin, M. H. Smith, and Lj. Trajkovic, "Distributed denial of service attacks," In *Proceedings of 2000 IEEE International Conference on Systems, Man, and Cybernetics*, October 2000.

[35] J. Yan, S. Early, R. Anderson, "The XenoService – A distributed defeat for distributed denial of service", In *Proceedings of ISW 2000*, October 2000.

[36] S.Floyd, S. Bellovin, J. Ioannidis, K. Kompella, R. Mahajan and V. Paxson, “Pushback Messages for Controlling aggregates in the Network," Internet draft, Work in progress, http://search.ietf.org/internet-drafts/draft-floyd-pushback-messages-00.txt, July 2001.

[37] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, R. Morris, "Resilient Overlay Networks," In *Proceedings of 18th ACM SOSP*, October 2001.

[38] Information Sciences Institute, "Dynabone," http://www.isi.edu/dynabone/

[39] T. M. Gil and M. Poleto, "MULTOPS: a data-structure for bandwidth attack detection," In *Proceedings of 10th Usenix Security Symposium*, August 2001.

[40] J. Li, J. Mirkovic, M. Wang, P. Reiher and L. Zhang, "SAVE: Source address validity enforcement protocol," In *Proceedings of INFOCOM 2002*, June 2002. To appear.

[41] 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 ACM SIGCOMM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, August 2001

[42] R. Stone. "CenterTrack: An IP Overlay Network for Tracking DoS Floods," In *Proccedings of 9th USENIX Security Symposium*, August 2000.

[43] McAfee, "Personal Firewall," http://www.mcafee.com/myapps/firewall/ov_firewall.asp

[44] McAfee,"VirusScan Online," http://www.mcafee.com/myapps/vso/default.asp

[45] Mananet, "Reverse Firewall," http://www.cs3-inc.com/ps_rfw.html

[46] Cisco, "Strategies to protect against distributed denial of service attacks," http://www.cisco.com/warp/public/707/newsflash.html

[47] D. Moore, H. Xiao, "Cisco quality of service and DDoS," http://www.mitre.org/support/papers/tech_papers_01/moore_cisco/index.shtml

[48] Mazu Networks, "Dynamically Provisioned Monitoring," http://www.mazunetworks.com/white_papers/provmon-toc.html

[49] Asta Networks, "Vantage System Overview," http://www.astanetworks.com/products/vantage/

[50] Arbor Networks, "PeakFlow DoS for Hosting Providers Datasheet," http://www.arbornetworks.com/up_media/up_files/PFDoS_ServProv_1.6.pdf

[51] BBN Technologies, "Applications that participate in their own defense," http://www.bbn.com/infosec/apod.html

[52] BBN Technologies, "Intrusion tolerance by unpredictability and adaptation," http://www.bbn.com/infosec/itua.html

[53] J. Shapiro and N. Hardy, "EROS: A principle-driven operating system from the ground up," *IEEE Software,* pp. 26-33, January/February 2002.

[54] E.O'Brien,"NetBouncer : A practical client-legitimacy-based DDoS defense via ingress filtering," http://www.nai.com/research/nailabs/development-solutions/netbouncer.asp

[55] CERT Coordination Center, "TCP SYN flooding and IP spoofing attacks," http://www.cert.org/advisories/CA-1996-21.html