Scope of DDoS Countermeasures: Taxonomy of Proposed Solutions and Design Goals for Real-World Deployment

David Champagne and Ruby B. Lee

Abstract—**Distributed Denial of Service (DDoS) attacks have been plaguing the Internet for several years. They cause economic losses due to the unavailability of services and potentially serious security problems due to incapacitation of critical infrastructures. Such severe implications lead the research community to strive to find DDoS countermeasures. In spite of all the ideas that have been developed, a practical and comprehensive defense system has yet to be deployed Internet-wide.**

Through a novel taxonomy, this paper classifies and describes DDoS countermeasures developed by industry and academia. To our knowledge, our taxonomy is the first to unify such a large body of work into a single, detailed classification. Based on the analysis of these ideas, we then introduce design goals and principles that can guide the development of a practical DDoS solution.

Index Terms—**DDoS Countermeasures, Design Goals, Distributed Denial of Service (DDoS), Taxonomy.**

I. Introduction

In a typical Distributed Denial of Service (DDoS) scenario, an attacker aims at taking down a service offered by a victim. Several computers are mobilized into an attack force. These computers (called zombies or secondary victims) have previously been compromised by the attacker. When an assault is successful, the targeted service becomes unavailable to its legitimate users.

Researchers have been exploring different possibilities of DDoS countermeasures for several years, but these may not have been accessible to real-world implementers who want to understand the scope of possible solutions. The purpose of this paper is to facilitate practical deployment of comprehensive DDoS solutions by identifying important research countermeasures and organizing them under a new taxonomy. Our goal is to provide a unified and structured view of the DDoS countermeasure design space, rather than yet another new solution to a part of the DDoS problem. We also present design goals and principles that we feel should guide the development of future real-world DDoS defense solutions.

This paper is organized as follows. In Section II, we briefly describe DDoS attacks. Section III introduces our taxonomy while Section IV and V present the design goals and principles of a practical defense mechanism. Finally, we discuss related work in Section VI.

Manuscript received August 18, 2006. This work was supported in part by DARPA and NSF Cybertrust CNS-0430487, CNS-0636808, NSF ITR CCR-0326372 and NSF CNS-0208946.

D. Champagne is with the Electrical Engineering Department, Princeton University, Princeton, NJ 08544 USA (phone: 609-258-2500; fax: 609-258-3745; e-mail: dav@princeton.edu).

R. B. Lee is with the Electrical Engineering Department, Princeton University, Princeton, NJ 08544 USA (e-mail: rblee@princeton.edu).

II. DDoS Attacks

In the initial phase of a DDoS attack, an attacker recruits computers into a zombie network (also called a botnet). One popular recruitment strategy is to install a software attack tool on a computer, which can later be remotely commanded to launch an attack on the victim.

Once recruited, zombie computers connect to a pre-determined handler machine by which communication with the attackers takes place. In a typical DDoS attack, the attacker orders every zombie in the network to send a stream of packets to a particular victim. The objective is to produce a packet flood exceeding the victim's processing capacity such that legitimate traffic cannot be serviced.

DDoS attacks either target the resources on a victim server or the bandwidth of the network through which it is accessed by clients. A comprehensive taxonomy of DDoS attacks and attack tools can be found in [1]. A common type of attack, the bandwidth depletion attack, aims at overloading links and routers. With sufficient intensity, such an attack can prevent legitimate traffic from reaching the victim server. Other attacks – called resource depletion attacks in [1] – target the resources on the server: typically, the attacker floods the server with packets which processing hogs the server's limited resources – e.g. memory, CPU time. In all cases, the DDoS attack prevents legitimate clients from accessing server resources.

III. Taxonomy

We classify DDoS countermeasures into three main categories as shown in Fig. 1: those that mitigate the damage done by a DDoS attack (section A), those that try to prevent

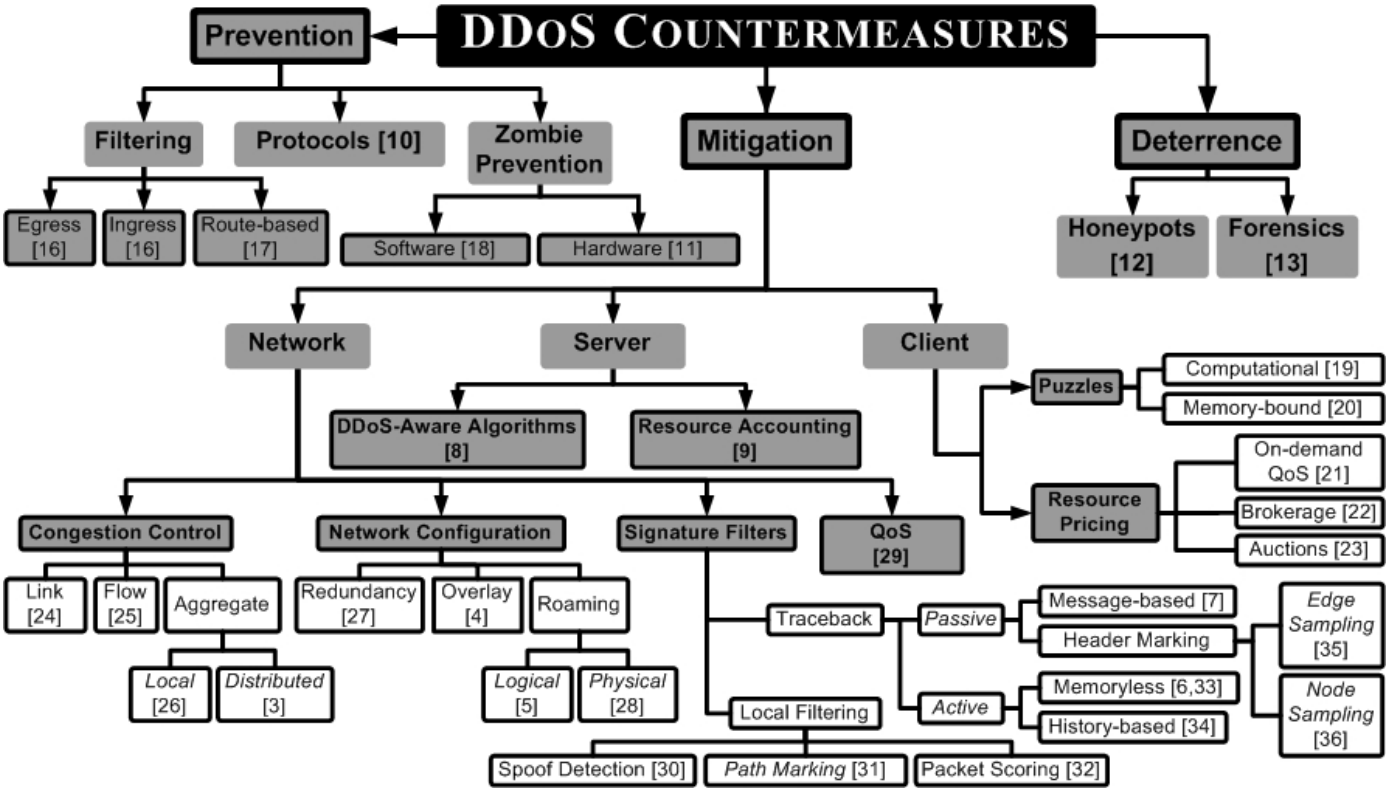

Fig. 1 - Taxonomy of DDoS Countermeasures

DDoS attacks from even occurring (section B), and those that serve to deter future DDoS attacks (section C). A fully exemplified version of the taxonomy presented below can be found in [2].

A. DDoS Mitigation

The main category of proposed DDoS countermeasures deals with mitigating the effects of an ongoing attack. These can be further classified as separate techniques for server, network or client based mitigation.

1) Network

In network-based mitigation, we consider techniques involving the intermediate network as well as some mitigation methods assisted by client or server subnets.

a) Congestion Control

Mitigating the effects of a DDoS attack does not necessarily require detection of the attack. A defense mechanism can be effective without knowing whether there is an ongoing attack or not. Applying policies that isolate certain portions of traffic from others can limit the impact of malicious behavior without the need for an attack detection mechanism. Such congestion control can be done at different levels of abstraction which we classify as link, flow or aggregate.

Link: In a link-based congestion control scheme, a router maintains a queue for each incoming link. Forwarding of packets is done by sampling the head packet from each queue on a round-robin basis.

Flow: One way to abstract traffic is to classify packets according to their network origins and destinations. The resulting classes of packets are called flows. Overloaded routers can be configured to throttle certain flows rather than a specific ingress link. This means routing policies do not affect a well-behaved flow sharing an ingress link with a misbehaved one.

Aggregate: When controlling a large number of zombie machines, attackers have the possibility to produce attacking flows that individually appear well-behaved. Such an attack is harder to detect but can have sufficient power to deplete available resources on the victim's network. Defense systems can thwart those attacks by classifying traffic with more precision than what is possible with flows. The resulting classes of packets are called aggregates.

An aggregate is defined as a group of packets sharing a common property [3]. Such properties include a packet's origin and destination as well as application or protocol type. For example, refined aggregates could consider only TCP SYN or ICMP ECHO packets. Isolating particular aggregates from the rest of the traffic allows more precise filtering that reduces the impact of DDoS attacks on unrelated traffic.

b) Network Configuration

Several schemes provide protection from DDoS attacks by modifying the physical or logical configuration of a network and its servers.

Redundancy: Basic schemes introduce redundancy in order to increase server processing capacity. The objective is to process all incoming traffic at any time so increases in a server's load do not affect any of its clients. Those techniques thus help a server withstand DDoS attacks as well as *flashcrowds*. *Flashcrowds* are sudden bursts in traffic due to a

large number of content requests from legitimate users.

Overlay: Advanced methods require either adding an extra layer of networking components atop an existing infrastructure or extending the functionality of nodes already in place.

In [4], a Secure Overlay Service (SOS) is provided by an overlay network of routers which use a hash-based algorithm to route packets to a server. An outside host wishing to communicate with that server must first contact a Secure Overlay Access Point (SOAP), a designated router that lets a packet enter the overlay only after authenticating its source.

Roaming: Many of the known attack tools carry out attacks without performing DNS lookups: they send packets directly to a victim's IP address [5]. Using this fact, a server under attack can modify its IP address. Doing so affects legitimate clients until they perform a new DNS lookup, but it protects the server from the malicious traffic still directed at the old IP address.

c) *Signature Filters*[1]

Signature-based strategies maintain the traditional best-effort routing model of the Internet in normal conditions. The defense system reacts only when an attack detection mechanism flags certain packets as malicious. Based on their reactive actions, signature-based mechanisms fall into two categories: local filtering and traceback mechanisms. Members of the latter category locate the source of the attack before taking action while local filtering methods immediately trigger countermeasures without knowledge of the attacker's location.

Local Filtering: Local filtering methods are distinguishable by the type of signature they use to classify traffic. When a node determines that packets with a certain signature are malicious, subsequent incoming packets with the same signature are discarded or rate-limited.

IP Traceback: Traceback mechanisms form a subset of DDoS countermeasures that focuses on localizing the origin of a stream of attacking packets, regardless of whether their source addresses have been spoofed. Knowledge of the attacker's location allows activation of filters closer to the source of the attack. This reduces the impact of collateral damage, where filtering nodes drop legitimate traffic identified as malicious. Such false positives are inevitable, but they are less likely to occur as filters affect a smaller part of the Internet.

Our taxonomy first classifies traceback methods according to the way they gather path data. Active traceback mechanisms recursively query upstream routers to obtain information about a certain malicious stream. With passive mechanisms, the intermediate network automatically sends path information to the victim. In both cases, the victim compiles the partial (or complete) path information to construct the sequence of routers used by a malicious packet to travel from the attacking machine to the victim.

Active traceback mechanisms can be further divided into *Memoryless* and *History-based* approaches. The former category does not require the routers in the intermediate network to store information about forwarded packets whereas the latter category requires routers to store some information about each packet they forward.

(*Active/Memoryless: Controlled Flooding*) Burch and Cheswick [6] proposed one of the first techniques tracing IP packets back to their source. In this method, the node under attack first requests generation of a brief burst of UDP packets on each link connected to the closest router. The overload due to the temporary increase in traffic causes the router to drop packets. Downstream, the victim monitors the effect of flooding each link. A decrease in attacking traffic concurrent to the flood of a certain link indicates an attacker located further upstream on that particular link. This link is thus added to the list of links forming the path to the attacker. The search for the attacker then goes on by recursively applying the method hop by hop. During that process, links for which flooding has no effect are pruned out of the exploration space.

Passive traceback methods can be divided into two subcategories: *message-based* methods on the one hand and *header marking* methods on the other hand. In *message-based* methods, nodes on the path of a packet generate extra packets that contain tracing information. With *header marking* techniques, routers include path information within the header of an IP packet.

ICMP traceback, an early method proposed in [7], uses such a *message-based* scheme. With a low probability (typically 1/20,000), routers participating in ICMP traceback send ICMP messages to the destination addresses of the packets they forward.

An ICMP message contains the IP address of the generating router. It also contains either the IP address of the router from which the generator received the packet of interest, or the IP address of the router to which the generator forwarded the packet of interest, or both. A control message thus consists of one or two edges in the graph representing the path of a packet. In order to form an attack graph, a host targeted by an attack compiles all the edges corresponding to packets it considers malicious. The resulting graph is a tree with the victim as a root and attackers as leaves.

d) *QOS*

A possible way to cope with attacks targeting depletion of network bandwidth is to introduce a service differentiation mechanism that reserves a share of the bandwidth to certain categories of traffic. Such a mechanism creates classes of packets that are each treated differently by the network (higher priority classes are forwarded first).

Ideally, only legitimate packets get preferential treatment and all attacking traffic belongs to the lowest class of service. However, in real-life situations, such precise categorization is hard to achieve.

[1] Throughout this paper, a certain packet stream's distinguishing characteristics are referred to as its *signature*.

2) Server

Several strategies investigate how to defend a server from DDoS attacks by modifying the server itself. These techniques deal primarily with software.

a) DDoS-Aware Algorithms

There exist simple ways for an operating system to mitigate the effects of a DDoS attack. For example, an OS can periodically scan the TCP connection queue and drop half-open connections. By doing so, the OS prevents a TCP SYN attack from hogging memory resources. Lazy Receiver Processing (LRP) [8] can also help an operating system in the case of a flooding attack by avoiding certain computations on packets that end up being dropped due to overload.

b) Resource Accounting

Some OS-level resource accounting schemes like Escort [9] are more elaborate than simple DDoS-aware algorithms. They enforce policies which control allocation of time-multiplexed resources such as CPU time or network bandwidth.

3) Client

Puzzles and Resource Pricing schemes are client-centric classes of countermeasures that take advantage of the limited computational and monetary resources of client hosts in order to force them to regulate their traffic. With Puzzles, every client requesting access to services must commit a certain amount of resources determined by the network or server. Resource Pricing strategies establish different market-like schemes in which resources are available for purchase by the clients.

B. DDoS PREVENTION

Reactive measures protecting critical services during an ongoing attack – e.g. the majority of measures in section A – undoubtedly are important, but ultimately, proactive measures need to be implemented to prevent the occurrence of an attack in the first place. Such preventive measures should either eliminate exploitable flaws on a network or work towards complicating the task of a potential attacker.

1) Filtering

When internetworking protocols such as TCP and IP were initially designed, functionality was of greater concern than security. As a result, malicious parties can generate invalid information or send harmful commands without being detected by those protocols. Filtering packets with such contents is a first step towards reducing an eventual enemy's perniciousness.

2) Protocols

How to design protocols that do not offer opportunities for DoS attacks is still an unsolved research problem. However, there are some desirable properties known to prevent specific types of attacks. For example, the goal of some attacks is to deplete a server's resources by establishing a large number of bogus TCP connections. That way, the TCP buffers are saturated with connection status and incoming connection requests must be ignored. A remedy to this problem involves designing stateless protocols which shift the burden of state holding from the server to the clients. SYN cookies [10] partially reach this goal by remaining stateless in the first steps of a TCP connection establishment.

3) Zombie Prevention

An important step towards solving the DDoS problem consists in preventing the attacker from constituting an army of zombie computers in the first place. To attain this goal, it is necessary to remediate the weaknesses attackers exploit to gain control of hosts connected to the public Internet. One of the most important weaknesses, buffer overflow [11], can be mitigated using either software or hardware mechanisms.

C. Deterrence

Even though some techniques allow the tracing of an attack back to the attacking hosts, a victim will rarely be able to identify and prosecute the attacker controlling those zombies. Any method altering this climate of impunity might deter some malicious parties from waging DDoS attacks.

1) Honeypots

Honeypots are computer systems placed on a network for the sole purpose of being abused by unsuspecting attackers [12]. Since a *honeypot* does not offer any useful service, nearly all activity detected on it is malicious. It is thus simple to use such a computer as an intrusion detection system.

In regular operating systems, *rootkits* can be used to cover the attacker's traces. In advanced *honeypot* systems, the OS is encapsulated in a logging framework so all attacker activity is recorded, regardless of attempts to alter the audit trail. The possibility to track the attacker's every move can be a deterrent since the attacker may not want to expose his tactics.

2) Forensics

With the use of custom-made "sniffers" and scripts, skilled programmers can manually trace the activity of malicious software back to the IRC channel used by the master attacker for controlling the bot network [13]. Such forensic activity could ultimately lead to the discovery of the attacker's identity.

IV. DESIGN GOALS

In this section, we present design goals which we believe should be targeted by developers of a comprehensive DDoS defense employing a coherent set of countermeasures. As we examine the body of work presented previously, these considerations emerge as being independent of the approach taken to solve the DDoS problem.

First of all, designers of DDoS defense systems should *aim at producing a low-cost solution*. A high dollar cost will discourage widespread adoption of a method by ISPs and potential victims of a DDoS attack. A method will also be disfavored if it slows down networks by a high computational overhead or if it requires significant memory resources at each node involved.

Ideally, defense mechanisms should *not degrade performance when there is no ongoing attack*. This is

especially important if some modifications are necessary on high-bandwidth routers of ISP networks.

Scalability is crucial for any real-world countermeasure. It should be possible to extend a defense system when additional resources need to be protected. Systems should also be able to deal with zombie armies of bigger sizes than what has been observed up to now.

Any defense mechanism should *take into account the aggregation of legitimate and malicious traffic*. As they are being routed to the server, packets from attackers and legitimate clients converge towards the same subnet and may end up sharing links. Defense mechanisms should thus be capable of either upstream filtering or precise downstream packet discrimination.

As long as software vulnerabilities and naïve users continue to exist, it will be possible for attackers to build vast zombie armies. Defense systems must thus be able to *deal with an attack consisting of a large number of individually well-behaved flows*. This is necessary to thwart clever attackers who configure their botnets to produce such attacks. In addition, defense methods should be able to *adapt to attacks with rapidly changing characteristics and intermittent attacks*. A comprehensive DDoS defense should *converge rapidly* in order to mitigate such attacks.

It should be *possible to adjust the reaction of a defense mechanism to a victim's load*. When defenses are enabled, there is usually a risk of mistaking a legitimate packet for a malicious one. In a situation where a victim is able to process all traffic despite ongoing malicious activity, it may be desirable to eliminate false positives (dropping of legitimate packets) by disabling certain countermeasures.

A practical countermeasure must have *minimal interference with* flashcrowds. Mechanisms with drastic reactions to abnormal traffic patterns may deny access to a large number of legitimate clients whose requests arrive in a given short interval of time.

Defenses must hold in the case of forged information. Mechanisms that rely on information contained in messages or packet markings should be efficient even if an attacker is trying to fool them.

Finally, a practical countermeasure *must allow incremental deployment*. It is unrealistic to assume that modifications can be "instantaneously" and uniformly carried out on every network, client or server around the globe. A solution to the DDoS problem should thus provide some protection even though it is not yet widely deployed.

V. DESIGN PRINCIPLES

In this section, we attempt to integrate important properties of practical DDoS countermeasures into design principles. We propose that these design principles can help defense solutions reach the goals stated in the previous section.

A. Distributed Defense Mechanisms

The primary benefit of a distributed countermeasure is to be easily scalable. Indeed, in a system which uses multiple entities (routers or computers) to protect resources, adding more defensive nodes typically enhances the protection.

Having a distributed DDoS defense system also makes it more difficult for a malicious party to attack the mechanism. If multiple entities are each responsible for a part of the defense, the attacker usually needs to take most of them down before considering an attack on a protected server. Moreover, distribution of the workload lowers the processing cost per entity.

When a distributed mechanism includes components further upstream, closer to traffic sources, it can take defensive actions that limit the aggregation of legitimate and malicious traffic. This is not possible for a single-component system located on the victim's network.

B. Collaborative Mechanisms

Each entity of a distributed defense mechanism can take action according to the information that is locally observable. For example, each node of a distributed filtering system filters out packet flows which appear malicious locally. Let's say that a given malicious flow is small enough to appear benign to upstream routers. As this flow progresses towards the victim, it aggregates with similar flows to form a larger flow which, to downstream routers, is obviously malevolent. If downstream routers communicate the signature of this flow to routers upstream, filtering can be implemented closer to the source of the attack. This limits the aggregation of legitimate and malicious traffic, lightening the filtering burden of downstream routers.

Collaboration thus increases the efficiency of a distributed countermeasure by providing each entity with global information that it could not gather by itself. Without collaboration, an entity's action is more limited. Moreover, getting a global view of incoming traffic might make it easier for a defense mechanism to discriminate between *flashcrowds* and DDoS attacks.

C. Minimize Information Requirements

The more information a router, a client or a server needs to handle in the context of a countermeasure, the more expensive the method is likely to be in terms of computational, memory and network bandwidth overhead. In addition, processing more information typically slows down the mechanism's convergence.

Moreover, countermeasures that rely extensively on communicated data are susceptible to attacks. Indeed, an attacker can try to fool the mechanism with bogus information. Authentication and verification of data prevent attackers from misleading the mechanism but open the door to denial of service through flooding of the verification process.

D. Independently Useful Mechanisms

Even though collaboration can enhance defenses, each node should ideally be able to function without the intervention of others. In a system where every node can independently take defensive actions, failure or overload of a given node should

not neutralize the whole system. Such a property is important in withstanding attacks on the defense mechanism itself.

Ideally, each piece of information should be meaningful in itself so it independently contributes to the system's defensive goals. It is also reasonable to assume such "information independence" reduces the impact of forged information: an attacker's fake message should not have any effect on the meaning of other messages.

VI. RELATED WORK

[14] and [15] describe taxonomies that rely on multiple types of classification to cover existing defense mechanisms. In [14], methods are classified either by activity level, by cooperation degree or by deployment location, while [15] uses countermeasure activity and location as classification systems. Classifying methods with multiple systems separates certain countermeasures whose purpose is similar and thus makes it harder for the reader to perceive the similarity. Under our integrated classification system, such a relation between countermeasures is exposed by their proximity in the taxonomy tree.

In summary, this paper introduces a unified taxonomy system that allows a clear and simplified view of the design space of countermeasure proposals. The paper also proposes novel design goals and design principles for comprehensive, scalable and practical DDoS solutions that are low-cost and can be incrementally deployed.

VII. REFERENCES

[1] S. M. Specht and R. B. Lee, "Distributed denial of service: taxonomies of attacks, tools and countermeasures," in *Proc. of the 17th ICPADS*, 2004 International Workshop on Security in Parallel and Distributed Systems, pp. 543-550, Sept. 2004.

[2] David Champagne and R. B. Lee, "Scope of DDoS countermeasures: taxonomy of proposed solutions and design goals for real-world deployment", *Princeton Univ. Tech. Report CE-L2005-007*, July 2005.

[3] R. Mahajan, et al., "Controlling high bandwidth aggregates in the network," in *Computer Communication Review*, Vol. 32 (3), pp. 62-73, 2002.

[4] D. K. Angelos, et al., "SOS: secure overlay services," in *Proc. of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 61-72, Aug. 2002.

[5] Federal Computer Incident Response Center (FedCIRC), "Defense Tactics for Distributed Denial of Service Attacks", Washington, DC, 2000.

[6] H. Burch and B. Cheswick, "Tracing anonymous packets to their approximate source," in *Proc. of the 14th Systems Administration Conference,* pp. 319-327, Dec. 2000.

[7] S. Bellovin, et al., "ICMP traceback messages", *Internet Draft (IETF)*, 2000.

[8] P. Druschel and G. Banga, "Lazy receiver processing (LRP): a network subsystem architecture for server systems," in *Proc. of the 2nd USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 261-275, Oct. 1996.

[9] O. Spatscheck and L. L. Peterson, "Defending against denial of service attacks in Scout," in *Proc. of the 3rd USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 59-72, Feb. 1999.

[10] D. J. Bernstein, "SYN Cookies", http://cr.yp.to/syncookies.html, 1996.

[11] R. B. Lee, et al., "Enlisting Hardware Architecture to Thwart Malicious Code Injection" in *Proc. of the International Conference on Security in Pervasive Computing*, pp. 237-252, Mar. 2003.

[12] L. Spitzner, *Honeypots: Tracking Hackers*, Addison-Wesley Professional, 2002.

[13] S. Gibson, "The Strange Tale of the DOS Attacks Against GRC.com", http://www.grc.com/files/grcdos.pdf, 2002.

[14] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," in *Computer Communication Review*, Vol. 34 (2), pp. 39-53, 2004.

[15] C. Douligeris and A. Mitrokotsa, "DDoS attacks and defense mechanisms: classification and state of the art," in *Computer Networks*, Vol. 44 (5), pp. 643-666, 2004.

[16] P. Ferguson and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", *Internet RFC's*, 2000.

[17] P. Kihong and L. Heejo, "On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law Internets," in *Computer Communication Review*, Vol. 31 (4), pp. 15-26, 2001.

[18] C. Cowan, et al., "StackGuard: Automatic Adaptive Detection and Prevention of Buffer-Overflow Attacks" in *Proc. of the 7th USENIX Security Conference*, pp. 63-78, Jan. 1998.

[19] A. Juels and J. Brainard, "Client puzzles: a cryptographic countermeasure against connection depletion attacks," in *Proc. of the Network and Distributed System Security Symposium*, pp. 151-165, Feb. 1999.

[20] M. Abadi, et al., "Moderately hard, memory-bound functions," in *Proc. of the Network and Distributed System Security Symposium*, pp. 107-121, Feb. 2003.

[21] M. A. Lejeune, "Awareness of Distributed Denial of Service Attacks' Dangers: Role of Internet Pricing Mechanisms," in *Netnomics*, Vol. 4 (2), pp. 145-162, 2002.

[22] D. Mankins, et al., "Mitigating distributed denial of service attacks with dynamic resource pricing," in *Proc. of the Computer Security Applications Conference*, pp. 411-421, Dec. 2001.

[23] X. Wang and M. K. Reiter, "Defending against denial-of-service attacks with puzzle auctions," in *Proc. of the IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 78-92, May 2003.

[24] Cisco Systems Inc., *Internetworking Technologies Handbook*, Pearson Education, 2003.

[25] Y. H. Hu, et al., "Packet filtering for congestion control under DoS attacks," in *Proc. of the 2nd IEEE Int. Information Assurance Workshop*, pp. 3-18, Apr. 2004.

[26] Cisco Systems Inc., http://www.cisco.com/univercd/cc/td/doc/product/software/ios111/cc111/car.htm.

[27] Akamai Technologies Inc., "On-demand DoS mitigation", http://www.akamai.com/en/html/services/site_protection.html.

[28] C. Sangpachatanaruk, et al., "Design and analysis of a replicated elusive server scheme for mitigating denial of service attacks," in *Journal of Systems and Software*, Vol. 73 (1), pp. 15-29, 2004.

[29] J. Brustoloni, "Protecting electronic commerce from distributed denial-of-service attacks," in *Proc. of the 11th international conference on World Wide Web*, pp. 553-561, May 2002.

[30] J. Cheng, et al., "Hop-count filtering: an effective defense against spoofed DDoS traffic," in *Proc. of the 10th ACM conf. on Computer and Communications Security*, pp. 30-41, Oct. 2003.

[31] A. Yaar, et al., "Pi: a path identification mechanism to defend against DDoS attacks," in *Proc. of the 2003 Symposium on Security and Privacy*, pp. 93-107, May 2003.

[32] K. Yoohwan, et al., "Packetscore: statistics-based overload control against distributed denial-of-service attacks," in *Proc. of the IEEE INFOCOM 2004*, pp. 2594-2604, Mar. 2004.

[33] S. Savage, et al., "Network support for IP traceback," in *IEEE/ACM Transactions on Networking*, Vol. 9 (3), pp. 226-237, 2001.

[34] A. C. Snoeren, et al., "Hash-based IP traceback," in *Computer Communication Review*, Vol. 31 (4), pp. 3-14, 2001.

[35] D. X. Song and A. Perrig, "Advanced and authenticated marking schemes for IP traceback," in *Proc. of the IEEE INFOCOM 2001*, pp. 878-886, Apr. 2001.

[36] T. W. Doeppner, et al., "Using router stamping to identify the source of IP packets," in *Proc. of the ACM Conference on Computer and Communications Security*, pp. 184-189, Nov. 2000.