Denial of Service Attacks

Qijun Gu, PhD.
Assistant Professor
Department of Computer Science
Texas State University – San Marcos
San Marcos, TX, 78666

Peng Liu, PhD.
Associate Professor
School of Information Sciences and Technology
Pennsylvania State University
University Park, PA, 16802

Denial of Service Attacks

Outline

ABSTRACT

Denial of service (DoS) attacks have become a major threat to current computer networks. To have a better understanding on DoS attacks, this article provides an overview on existing DoS attacks and major defense technologies in the Internet and wireless networks. In particular, we describe network based and host based DoS attack techniques to illustrate attack principles. DoS attacks are classified according to their major attack characteristics. Current counterattack technologies are also reviewed, including major defense products in deployment and representative defense approaches in research. Finally, DoS attacks and defenses in 802.11 based wireless networks are explored at physical, MAC and network layers.

Key Words: Denial of Service, Distributed Denial of Service, Internet Security, Wireless Security, Scanner, Spoofing, Pushback, Traceback, Jamming, 802.11, Secure Routing, Secure Forwarding

I. INTRODUCTION

Denial of service (DoS) attacks have become a major threat to current computer networks. Early DoS attacks were technical games played among underground attackers. For example, an attacker might want to get control of an IRC channel via performing DoS attacks against the channel owner. Attackers could get recognition in the underground community via taking down popular web sites. Because easy-to-use DoS tools, such as Trinoo (Dittrich 1999), can be easily downloaded from the Internet, normal computer users can become DoS attackers as well. They sometime coordinately expressed their views via launching DoS attacks against organizations whose policies they disagreed with. DoS attacks also appeared in illegal actions. Companies might use DoS attacks to knock off their competitors in the market. Extortion via DoS attacks were on rise in the past years (Pappalardo *et al.* 2005). Attackers threatened online businesses with DoS attacks and requested payments for protection.

Known DoS attacks in the Internet generally conquer the target by exhausting its resources, that can be anything related to network computing and service performance, such as link bandwidth, TCP connection buffers, application/service buffer, CPU cycles, etc. Individual attackers can also exploit vulnerability, break into target servers, and then bring down services. Because it is difficult for attackers to overload the target's resource from a single computer, many recent DoS attacks were launched via a large number of distributed attacking hosts in the Internet. These attacks are called distributed denial of service (DDoS) attacks. In a DDoS attack, because the aggregation of the attacking traffic can be tremendous compared to the victim's resource, the attack can force the victim to significantly downgrade its service performance or even stop delivering any service. Compared with conventional DoS attacks that could be addressed by better securing service systems or prohibiting unauthorized remote or local access, DDoS attacks are more complex and harder to prevent. Since many unwitting hosts are involved in DDoS attacks, it is challenging to distinguish the attacking hosts and take reaction against them. In recent years, DDoS attacks have increased in frequency, sophistication and severity due to the fact that computer vulnerabilities are increasing fast (CERT 2006, Houle *et al.* 2001), which enable attackers to break into and install various attacking tools in many computers.

Wireless networks also suffer from DoS attacks because mobile nodes (such as laptops, cell phones, etc.) share the same physical media for transmitting and receiving signals; and mobile computing resources (such as bandwidth, CPU and power) are usually more constrained than those available to wired nodes. In a wireless network, a single attacker can easily forge, modify or inject packets to disrupt connections between legitimate mobile nodes and cause DoS effects. In this article, we will provide an overview on existing DoS attacks and major defense technologies. The article is organized as follows. In Section II, major DoS attack techniques in the Internet are overviewed. We also discuss the reasons why a DoS attack can succeed and why defense is difficult. In Section III, a taxonomy of DDoS attacks is discussed according to several major attack characteristics. In Section IV, recent DDoS defense technologies are overviewed according to their deployment locations. In Section V, DoS attacks and defenses in wireless networks are discussed according to different network layers. Finally, we conclude this article in Section VI.

II. OVERVIEW OF DOS ATTACKS IN THE INTERNET

In this section, we overview the common DDoS attack techniques and discuss why attacks succeed fundamentally.

II.A. Attack Techniques

Many attack techniques can be used for DoS purpose as long as they can disable service, or downgrade service performance by exhausting resources for providing services. Although it is impossible to enumerate all existing attack techniques, we describe several representative network based and host based attacks in this section to illustrate attack principles. Readers can also find complementary information on DoS attacks in Handley *et al.* 2006 and Mirkovic *et al.* 2005.

Network Based Attacks

TCP SYN Flooding. DoS attacks often exploit stateful network protocols (Jian 2000, Shannon *et al.* 2002), because these protocols consume resources to maintain states. TCP SYN flooding is one of such attacks and had a wide impact on many systems. When a client attempts to establish a TCP connection to a server, the client first sends a SYN message to the server. The server then acknowledges by sending a SYN-ACK message to the client. The client completes the establishment by responding with an ACK message. The connection between the client and the server is then opened, and the service-specific data can be exchanged between them. The abuse arises at the half-open state when the server is waiting for the client's ACK message after sending the SYN-ACK message to the client (CERT 1996). The server needs to allocate memory for storing the information of the half-open connection. The memory will not be released until either the server receives the final ACK message or the half-open connection expires. Attacking hosts can easily create half-open connections via spoofing source IPs in SYN messages or ignoring SYN-ACKs. The consequence is that the final ACK message will never be sent to the victim. Because the victim normally only allocates a limited size of space in its process table, too many half-open connections will soon fill the space. Even though the half-open connections will eventually expire due to the timeout, zombies can aggressively send spoofed TCP SYN packets requesting connections at a much higher rate than the expiration rate. Finally, the victim will be unable to accept any new incoming connection and thus cannot provide services.

ICMP Smurf Flooding. ICMP is often used to determine if a computer in the Internet is responding. To achieve this task, an ICMP echo request packet is sent to a computer. If the computer receives the request packet, it will return an ICMP echo reply packet. In a smurf attack, attacking hosts forge ICMP echo requests having the victim's address as the source address and the broadcast address of these remote networks as the destination address (CERT 1998). As depicted in Figure 1, if the firewall or router of the remote network does not filter the special crafted packets, they will be delivered (broadcast) to all computers on that network. These computers will then send ICMP echo reply packets back to the source (i.e., the victim) carried in the request packets. The victim's network is thus congested.

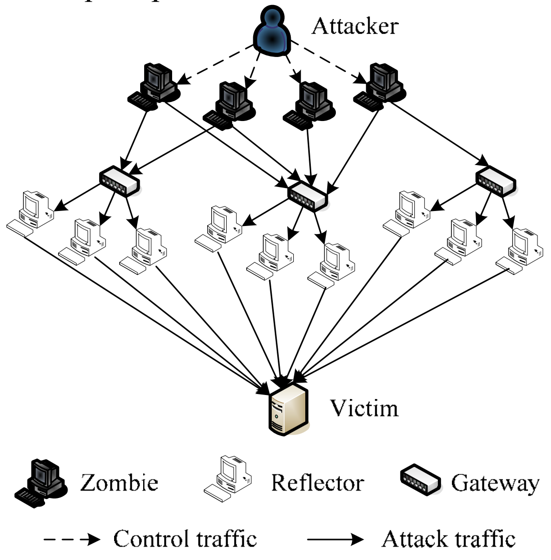

Figure 1 ICMP Smurf Attack

UDP Flooding. By patching or redesigning the implementation of TCP and ICMP protocols, current networks and systems have incorporated new security features to prevent TCP and ICMP attacks. Nevertheless, attackers may simply send a large amount of UDP packets towards a victim. Since an intermediate network can deliver higher traffic volume than the victim network can handle, the flooding traffic can exhaust the victim's connection resources. Pure flooding can be done with any type of packets. Attackers can also choose to flood service requests so that the victim cannot handle all requests with its constrained resources (i.e., service memory or CPU cycles). Note that UDP flooding is similar to flash crowds that occur when a large number of users try to access the same server simultaneously. However, the intent and the triggering mechanisms for DDoS attacks and flash crowds are different.

Intermittent Flooding. Attackers can further tune their flooding actions to reduce the average flooding rate to a very low level while achieving equivalent attack impacts on legitimate TCP connections. In shrew attacks (Kuzmanovic *et al.* 2003), attacking hosts can flood packets in a burst to congest and disrupt existing TCP connections. Since all disrupted TCP connections will wait a specific period (called retransmission-time-out (RTO)) to retransmit lost packets, attacking hosts can flood packets at the next RTO to disrupt retransmission. Thereby, attacking hosts can synchronize their flooding at the following RTOs and disable legitimate TCP connections as depicted in Figure 2. Such collaboration among attacking hosts not only reduces overall flooding traffic, but also helps avoid detection. Similar attack techniques targeting services with congestion control mechanisms for Quality of Service (QoS) have been discovered by Guirguis *et al.* (2005). When a QoS enabled server receives a burst of service requests, it will temporarily throttle incoming requests for a period until previous requests have been processed. Thus, attackers can flood requests at a pace to keep the server throttling the incoming requests and achieve the DoS effect. Guirguis's study showed that a burst of 800 requests can bring down a web server for 200 seconds, and thereby the average flooding rate could be as low as 4 requests per second.

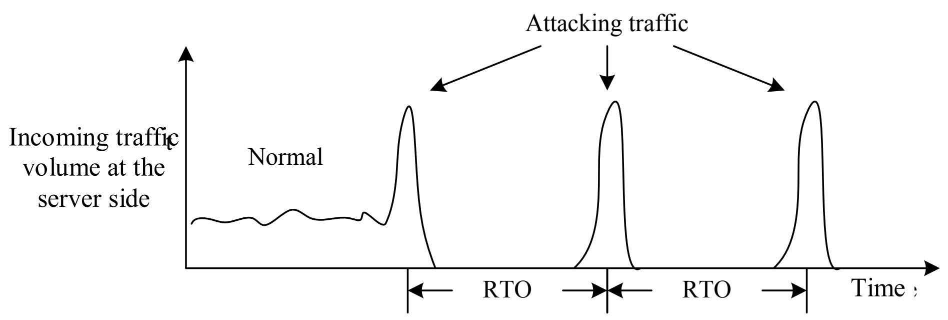

Figure 2 Low-rate Intermittent Flooding

Host Based Attacks

Besides misusing network protocols, attackers can also launch DoS attacks via exploiting vulnerabilities in target's applications and systems. Different from network based attacks, this type of attacks are application specific, i.e., exploiting particular algorithms (Crosby *et al.* 2003), memory structure (Cowan *et al.* 2003), authentication protocols (Dean *et al.* 2001, Zhang *et al.* 2005), implementation (CERT 1997), etc. Attacks can be launched either from a single host as a conventional intrusion or from a number of hosts as a network based DDoS attack. The traffic of host based attacks may not be as high as network based attacks, because application flaws and deficiencies can easily crash applications or consume a tremendous amount of computer resources. Several example attacks are described as follows.

Dean *et al.* (2001) identified that attackers could easily arrange an attack such that E-commerce web sites remain available, but clients are unable to complete any purchase. Such an attack is based on going after the secure server that processes credit card payments. In such E-commerce applications, the SSL/TLS protocol is used to make secure connections between clients and servers. The protocol allows a client to request the server to perform an RSA decryption. RSA decryption is an expensive operation. For instance, a large secure web site can process a few thousand RSA decryptions per second. If an SSL handshake request takes 200 bytes and a server can process 5000 decryptions per second, 1MB/s of requests is sufficient to paralyze an E-commerce site, which is a hard-to-notice small amount of traffic. Attackers can also send large modulo values via client certificates to increase the RSA computation per authentication. Consequently, mutual authentication cannot be done quickly and service performance is downgraded.

Researchers also found that attackers could exploit algorithmic deficiencies in many applications' data structures to launch low-bandwidth DoS attacks (Crosby *et al.* 2003). Because many frequently used data structures have "average-case" expected running time that is far more efficient than the worst case, attackers can carefully choose inputs to produce the worst scenario in data structures. Crosby *et al.* demonstrated this kind of DoS attacks against the hash table implementations. Normally, inserting n inputs into a hash table requires O(n) computation on average (Figure 3(a)). However, inputs could collide if they have the same hash value. Then, some applications use open addressing to solve the collision via following a deterministic strategy to probe for empty hash table entries. In the worst case where all n inputs collide, O(n^2) computation will be required (Figure 3(b)). Crosby *et al.* found that attackers can easily figure out such collision inputs in some hash algorithms, and demonstrated that attackers could bring down two versions of Perl, the Squid web proxy, and the Bro intrusion detection system via inputting strings that collide to crash the critical hash tables in these applications.

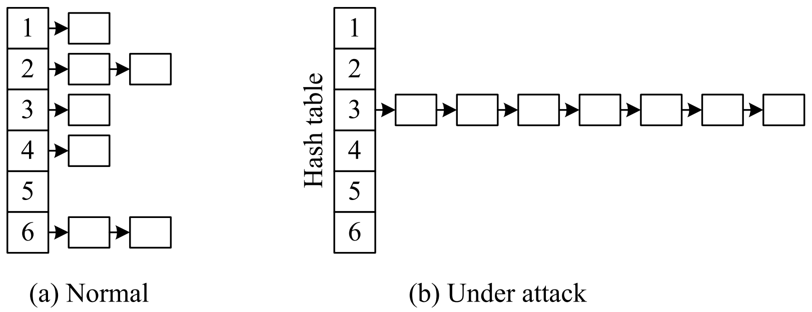

Figure 3 An Example of Attack against Application Deficiency

II.B. Attack Network

Many recent DoS attacks (also called DDoS attacks) were launched from distributed attacking hosts. A DDoS attack is launched in two phases. First, an attacker builds an attack network which is distributed and consists of thousands of compromised computers (called zombies, bots, or attacking hosts). Then, the attacking hosts flood a tremendous volume of traffic towards victims either under the command of the attacker or automatically.

To build an attack network, the attacker looks for computers that are poorly secured, such as those not having been properly patched. In general, a vulnerable host can be compromised via two types of approaches. One is to entice users to run malicious programs, such as a virus, a spyware, or a Trojan horse carried in malicious emails, files, or web pages. The other approach is via automated malicious programs, such as worms that can automatically scan vulnerable remote computers. The vulnerability in these computers is then exploited to allow the attacker to break into and install DoS attacking programs that further scan other hosts, install backdoors and flood packets. The attacker is thus called the master of these compromised computers (zombies). Some DoS programs have the ability to register the compromised computer as a member in the attack network controlled by the attacker. In addition, the newly compromised computers will automatically repeat the scanning and exploiting process to look for other vulnerable computers. Because of the self-propagation, a large attack network can quickly be built to include hundreds or thousands of computers.

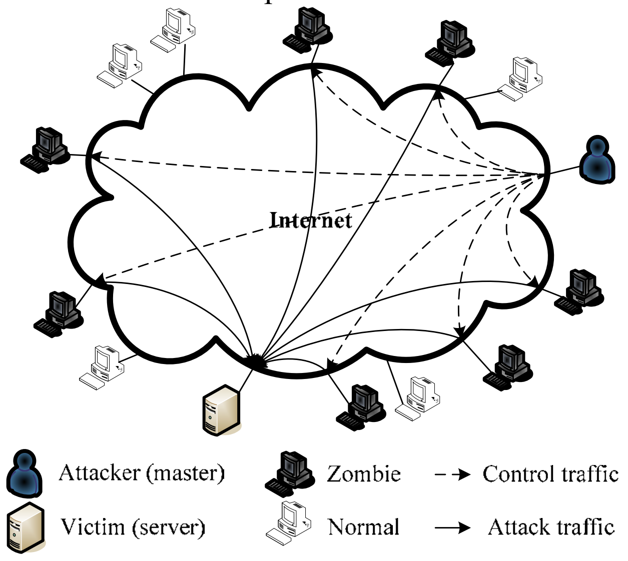

Figure 4 Attack Network (BotNet)

BotNet (Honeynet 2005) is an example of attack networks (depicted in Figure 4) in which an attacker controls a large number of zombies. In the attack network, zombies are called bots. They can be used to scan and subvert other vulnerable computers to new bots. They can communicate with the attacker and synchronize their actions via covert channels. For instance, Internet Relay Chat (IRC) is a legitimate communication service, and the attacker sends commands via an IRC channel to control the zombies. Nevertheless, the attacker usually disguises his control packets as legitimate chat traffic. The discovery of a zombie may not reveal the identification of other zombies or the attacker. To further avoid discovery, the attacker can frequently switch IRC channels. In addition, because IRC service is maintained in a distributed manner, an IRC server can host a particular IRC channel anywhere in the world. Investigation in the IRC channel thus faces many practical issues.

MyDoom virus provides another example of building such a DoS attack network (CERT 2004). Different from BotNets, the attack network was not solely built through technological vulnerabilities. One variant of MyDoom could coax computer users into executing a malicious program that was either sent as an email attachment or as a file downloaded from a P2P network. The attacking program in compromised computers was designed to automatically and simultaneously flood towards www.sco.com on February 1, 2004 and www.microsoft.com on February 3, 2004.

II.C. Why a DoS/DDoS Attack May Succeed

The design of the Internet is one of the fundamental reasons for successful DoS attacks. The Internet is designed to run end-to-end applications. Routers are expected to provide the best-effort packet forwarding, while the sender and the receiver are responsible for achieving desired service guarantees such as quality of service and security. Accordingly, different amounts of resources are allocated to different roles. Routers are designed to handle large throughput that leads to the design of high bandwidth pathways in the intermediate network. On the contrary, end hosts may be only assigned as much bandwidth as they need for their own applications. Consequently, each end host has less bandwidth than routers. Attackers can misuse the abundant resources in routers for delivery of numerous packets to a target.

The control and management of the Internet is distributed. Each component network is run according to local policies designed by its owners. No deployment of security mechanisms or security policy can be globally enforced. Because DoS attacks are commonly launched from systems that are subverted through security-related compromises, the susceptibility of the victim to DoS attacks depends on the state of security in the rest of the global Internet, regardless of how well the victim may be secured. Furthermore, it is often impossible to investigate cross-network traffic behaviors in such a distributed management. If one party in a two-way communication (sender or receiver) misbehaves, it can cause arbitrary damage to its peer. No third party will step in to stop it.

The design of the Internet also hinders the advancement of DDoS defenses. Many solutions have been proposed to require routers and source networks, together with victims, to participate in constructing a distributed and coordinated defense system. However, the Internet is administered in a distributed manner. Besides the difference of individual security policies, many entities (source networks and routers) in the Internet may not directly suffer from DDoS attacks. Hence, they may not have enough incentive to provide their own resources in the cooperative defense system.

Finally, the scale of the Internet requires more effort to get a defense system benchmark and test bed to evaluate and compare proposed defense technologies. Defense systems should be evaluated with large-scale experiments, data should be collected from extensive simulations, and cooperation is indispensable among different platforms and networks. Recently, researchers, industries and the US government are trying to develop a large-scale cyber security test bed (DETER 2005) and design benchmarking suites and measurement methodology (EMIST 2005) for security systems evaluation.

III. TAXONOMY OF DOS/DDOS ATTACKS IN THE INTERNET

In this section, DoS attacks are classified based on the material presented by Mirkovic *et al.* (2004). Interested readers can obtain further information on DDoS attacks from Mirkovic's book (Mirkovic *et al.* 2005). The classification is according to the major characteristics of DDoS attacks: 1) how attackers (or zombies) scan vulnerable computers, 2) how attack packets are spoofed, 3) what attack targets are, and 4) what attack impacts are.

III.A. Scanning

In the past, an attacker manually scans remote computers for vulnerability and installs attack programs. Now, the scanning has been automated by worms (Staniford *et al.* 2002) to help attackers to quickly probe potentially vulnerable computers. Attackers can also control the scanning behavior to find vulnerable computers in a slow and stealthy manner. Worm scanning may cause secondary impacts on ARP (CISCO 2001) and multicast (Hamadeh *et al.* 2005) due to high router CPU utilization and memory demand.

Random Scanning

In a random scanning, each compromised computer probes random addresses in either global or local IP address space. Because scanning attempts are not synchronized among attacking hosts, there are often duplicate probes to the same addresses. Thereby, a high volume of scanning traffic may present, especially when most of vulnerable computers in the Internet are infected.

Hitlist Scanning

A compromised computer performs hitlist scanning according to an externally supplied list. When it detects a vulnerable computer, it sends a portion of the hitlist to the recipient. A complete hitlist allows an attacker to infect the entire susceptible population within 30 seconds (Staniford *et al.* 2002). Nevertheless, the hitlist needs to be assembled in advance via other reconnaissance and scanning approaches. In addition, the list might be long. Thus, the transmission of the hitlist between two hosts might create a high volume of traffic.

Signpost Scanning

Signpost scanning takes advantage of habitual communication patterns of the compromised host to select new targets. E-mail worms use the information from the address books of compromised machines for their propagation. A Web-based worm could be propagated by infecting every vulnerable client that clicks on the server's Web page. Signpost scanning does not generate a high traffic load. The drawback is that the propagation speed depends on their user behavior but is not controllable by the attacker.

Permutation Scanning

This type of scanning requires all compromised computers to share a common pseudo-random permutation of the IP address space, and each IP address is mapped to an index in this permutation. Permutation scanning is preceded by a small-hitlist scanning. A computer infected in the hitlist then scans through the permutation, starting with its IP address. A compromised computer by permutation scanning starts from a random point in the permutation. Whenever a scanning host sees an already-infected machine, it chooses a new random starting point. The scanning host will become dormant to avoid collision after encountering some threshold number of infected hosts. The analysis (Staniford *et al.* 2002) shows that the propagation speed could be on the order of 15 minutes if a 10,000 entry hitlist is used and a worm generates 100 scans per second. As infection progresses, a large percentage of infected machines become dormant, which hinders detection due to the low duplicate scans.

III.B. Spoofing

Spoofing techniques define how the attacker chooses the spoofed source address in its attack packets. Spoofing attackers can choose one of the three approaches as depicted in Figure 5.

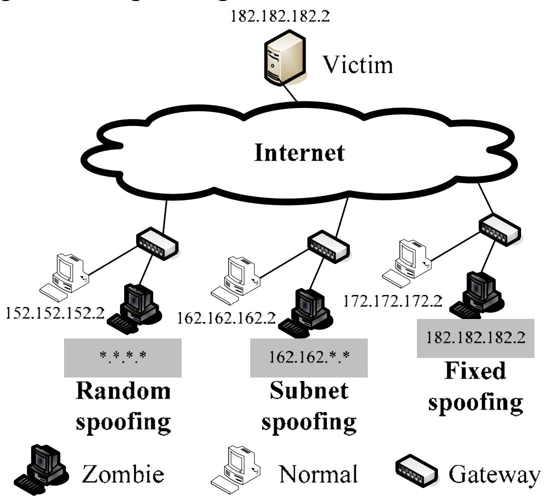

Figure 5 Source Address Spoofing

Random Spoofing

Attackers can spoof random source addresses in attack packets since this can simply be achieved by generating random 32-bit numbers and stamping packets with them. Attackers may also choose more sophisticated spoofing techniques, such as subnet spoofing, to defeat some anti-spoofing firewalls and routers using ingress filtering (Ferguson *et al.* 1998) and route-based filtering (Li *et al.* 2002).

Subnet Spoofing

In subnet spoofing, the attacker spoofs a random address within the address space of the sub-network. For example, a computer in a C-class network could spoof any address with the same prefix. It is impossible to detect this type of spoofing anywhere in the Internet. Subnet spoofing is useful for the attackers that compromise machines on networks running ingress filtering. A possible defense against this type of spoofing is to bind the IP address, the MAC address and the network port of each computer in the sub network.

Fixed Spoofing

Different from the other two spoofing techniques, the spoofed address is the address of the target. For example, an attacker performing a smurf attack spoofs the victim's address so that ICMP ECHO packets will be reflected to the victim.

III.C. Target

Although most DoS attacks work via exhausting resources, the actual target to deny services varies. The target could be the server application, the network access, or the network infrastructure.

Server Application

An application attack targets a given application on the victim (normally a server), thus disabling legitimate clients to use the service and possibly tying up resources of the host machine. Nevertheless, if the victim can well separate the resources for different applications, other applications and services in the victim should still be accessible to users. For example, a signature attack on an authentication server can send a large volume of authentication requests to exhaust the victim's resources of the signature verification and thus bring down the service. Nevertheless, if the victim can separate resources for applications, the victim can still have resources for other applications that do not require authenticated access. Consequently, the victim may need to enforce defense in each application, instead of only in its network access or its operating system.

Network Access
This type of attack disables access to the victim computer or network by crashing it or overloading its communication mechanism. An example of this attack is the UDP flooding attack. All attack packets carry the destination address of the target host without concerning the content of the target service. The high packet volume consumes network resources of the victim. At the same time, the attack traffic in fact facilitates detection, but the host cannot defend against these attacks alone. It must request help from some upstream routers and firewalls.

Infrastructure
Infrastructure attacks target some critical services that are crucial for global Internet operation. Examples include the attacks on domain name servers, core routers, certificate servers, etc. The key feature of these attacks is not the attack mechanism, but the attack target and the attack impact.

III.D. Impact

Depending on the impact of a DDoS attack on the victim, the attacks are classified as disruptive and degrading attacks.

Disruptive
The objective of disruptive attacks is to completely stall or crash the victim's service. For instance, the victim network is completely congested under attack, or the victim server crashes or halts under attack. Consequently, clients cannot access the service.

Degrading
The objective of degrading attacks is to consume some portion of a victim's resources to seriously downgrade the service performance. For example, an attack sends a high volume of authentication requests that can significantly consume computing resource in the target server. This attack can slow down the service response to legitimate customers. If customers are dissatisfied with the service quality, they may change their service provider.

IV. DDOS DEFENSES IN THE INTERNET

In this section, we overview products that have been deployed with an emphasis on their functionalities and principles. Then, we will focus our discussion on recent defense technologies proposed by researchers, and categorize them according to where they could be deployed.

IV.A. Defense Technologies in Deployment

In response to DDoS attacks, a variety of commercial products have been developed and deployed by networking and security manufactures, mainly including intrusion detection systems (IDSs), firewalls and security enhanced routers. These devices are normally deployed between the Internet and servers so that they can monitor incoming and outgoing traffic and take appropriate actions to protect servers. Fundamental technologies inside these devices include traffic analysis, access control, packet filtering, address blocking, redundancy etc.

IDSs typically log incoming traffic and make statistics from traffic traces. For example, CISCO IOS NetFlow (CISCO 2006a) can account network traffic and usage and provide valuable information about network users and applications, peak usage times, and traffic routing. Traffic traces and statistics can be compared to baseline traffic profiles to identify potential DoS attacks. In the past, most DDoS attacks caught attention due to abnormal conditions of the victim network, such as high traffic volume targeting at a certain port, slowing down of target servers, or high dropping rates of service requests. In addition, well-known DoS attack signatures (e.g., TCP SYN flooding) can also be captured to raise alerts.

Firewalls are widely deployed in the defense against DoS attacks. With correct configurations, firewalls are used to inspect ingress and egress packets and filter unwanted packets. Firewalls allow or deny certain packets according to protocols, ports, IP addresses, payloads, connection states, etc. The information is usually defined in access control lists and filtering rules in firewalls. Some firewalls and IDSs (CISCO 2006b and Bro 2006) can conduct stateful inspection so that only legitimate packets for ongoing connections can be passed into networks, and only legitimate TCP connections can be established and maintained. Firewalls can also create profiles (idle duration, data rate, etc.) for connections and conduct real time analysis to detect and forbid malicious attempts (Thomas *et al.* 2003). Several products (MAZU 2006, CISCO 2006c) integrate intrusion detection and firewall functions, and give more DDoS-specific visibility of the network for administration. Security measures in routers can push the defense frontline further away from the target, so that internal networks will not be directly impacted by DDoS flooding traffic. Similar to firewalls, many routers have access control lists, and can filter or rate-limit traffic. Routers can quickly filter packets having bogus and unwanted IP addresses, while firewalls can more closely examine packet payloads.

Service providers can also increase redundancy of network and service infrastructure (Handley *et al.* 2006). Service content can be backupped in redundant servers. When a server fails, redundant servers can take it over. Failure due to DoS attacks, in this case, is the same as a regular failure. Service access points can be distributed across network as well. Since DoS attacks may only target at a single network link, redundant network accesses can alternatively provide services. Nevertheless, redundancy solution may not be effective as expected against DoS attacks. First, redundancy asks for additional computing resources to handle incoming traffic. It could be costly for service providers to maintain enough computing resources. Second, attackers can easily find a large number of attacking agents in the Internet (refer to Section II.A) to overwhelm the capability of redundant equipments.

Although a few practical solutions and products have been deployed, many problems still exist. First, it is hard to distinguish flash crowds from flooding traffic. For example, firewalls may not prevent attacks against port 80 (web service) of servers, because many packets are just web surfing traffic to the web sites hosted by the target servers. Second, when flooding traffic is mitigated through filtering and rate-limiting mechanisms, some portion of the legitimate traffic may also be discarded. Access control lists may be setup on wrong information as well, because flooding packets may spoof addresses. Third, firewalls and routers can be easily overwhelmed under DDoS attack. They may be slowed down or congested. If they cannot forward incoming packets to servers, attackers also achieve DoS effects. Fourth, existing products act on their own without any control over other routers. Even if a border router or firewall has identified the upstream routers where flooding packets are coming, it is not easy to ask the owners of upstream routers (e.g., telecom companies or internet service providers) to throttle some specific traffic flows. Observing these problems, researchers have proposed quite a few DDoS defense technologies recently. In the following, we present them according to the location where they could be deployed.

IV.B. Attacker Side Defenses

Since source address spoofing plays an important role in DDoS attacks, restricting the source address of the traffic to known prefixes is a straightforward idea. Ingress filtering is a representative technique (Ferguson *et al.* 1998) based on this idea. As depicted in Figure 5, an attacker resides within a C-class network 152.152.152.0/24. The gateway of the network, which provides connectivity to the attacker's network, only allows traffic originating from source addresses with the 152.152.152.0/24 prefix, while prohibiting an attacker from using "invalid" source addresses which reside outside of this prefix range (e.g., 192.100.201.20). This technique can well prevent random address spoofing and fixed address spoofing. The idea of filtering packets according to their source addresses is also applicable in Internet routers (Li *et al.* 2002, Park *et al.* 2001). SAVE (Li *et al.* 2002) enforces routers to build and maintain an incoming table for verifying whether each packet arrives at the expected interface. Obviously, ingress filtering does not preclude an attacker using a forged source address of another host within the permitted prefix filter range (i.e., subnet address spoofing). Nevertheless, when an attack of subnet address spoofing occurs, a network administrator can be sure that the attack is actually originating from within the known prefixes that are being advertised.

D-WARD (Mirkovic *et al.* 2002) is another technique on the attacker side that prevents the machines from participating in DDoS attacks. D-WARD monitors two-way traffic between the internal addresses and the rest of the Internet. Statistics of active traffic are kept in a connection hash table and compared to predefined models of normal traffic, and non-complying flows are rate-limited. The imposed rate limit is dynamically adjusted as flow behavior changes, in order to facilitate fast recovery of misclassified legitimate flows and severely limit ill-behaved aggressive flows that are likely to be a part of an attack. Due to the limited size of the hash table, D-WARD can possibly discard packets of legitimate traffic during a DDoS attack where many malicious flows are present. D-WARD also needs to be able to monitor traffic in both incoming and outgoing directions, which may be not realistic when a network has several border routers. D-WARD, similar to other attacker side defense systems, can only limit attack traffic from the networks where it is deployed. Attackers can still perform successful DDoS attacks from networks that are not equipped with D-WARD system. Because attacking sources are randomly distributed, it is thus necessary to deploy attack side defense systems in as many networks as possible so that DDoS attacks can be fully controlled at source.

IV.C. Victim Side Defenses

Many defense systems are proposed at the victim side (Gil *et al.* 2001, Wang *et al.* 2002) since victims suffer the greatest impact of DoS attacks and are therefore most motivated to enforce defense. Jin *et al.* (2003) proposed hop-count filtering based on the observation that most randomly spoofed packets do not carry hop-count values that are consistent with the spoofed addresses. Thereby, time-to-live (TTL) value in packets can be used to decide if a packet is spoofed. A router decreases the TTL value of an in-transit IP packet by one before forwarding it to the next-hop. The final TTL value when a packet reaches its destination is the initial TTL subtracted by the number of intermediate hops (i.e., hop-count). However, the final TTL value that the destination can see does not directly reveal the hop-count. Fortunately, most modern OSs use only a few selected initial TTL values, 30, 32, 60, 64, 128, and 255. This set of initial TTL values covers most of the popular OSs, such as Microsoft Windows, Linux, BSD, and many commercial Unix systems. In addition, few Internet hosts are apart by more than 30 hops

(Cheswick *et al.* 2000). Hence, one can determine the initial TTL value of a packet by selecting the smallest initial value in the set that is larger than its final TTL. For example, if the final TTL value is 112, the initial TTL value is 128 (i.e., the smaller one of the two possible initial values, 128 and 255). A hop-count filtering system first builds an IP-to-hop-count mapping table, and clusters address prefixes based on hop-count. With the table, the system works in two running states: alert and action. Under normal condition, the system stays in alert state, watching for abnormal TTL behaviors without discarding any packet. Even if a legitimate packet is incorrectly identified as a spoofed one, it will not be dropped. Therefore, there is no collateral damage in the alert state. Upon detection of an attack, the system switches to action state, in which the system discards those IP packets with mismatching hop-counts. The inspection algorithm extracts the source IP address and the final TTL value from each IP packet. The algorithm infers the initial TTL value and subtracts the final TTL value to obtain the hop-count. The source IP address serves as the index into the table to retrieve the correct hop-count for this IP address. If the computed hop-count does not match the stored hop-count, the packet is classified as spoofed. Hop-count filtering cannot recognize forged packets whose source IP addresses have the same hop-count value as that of a zombie, a diverse hop-count distribution is critical to effective filtering. Jin *et al.* (2003) found that the Gaussian distribution is a good first-order approximation. The largest percentage of IP addresses that have a common hop-count value is 10%. The analysis using network measurement data shows that hop-count-filtering can recognize nearly 90% of spoofed IP packets. The stability in hop-counts between a server and its clients is also crucial for hop-count filtering to work correctly and effectively. Frequent changes in the hop-count between the server and each of its clients can lead to excessive mapping updates. Paxson (1997) found that the Internet paths are dominated by a few prevalent routes, and about two thirds of them have routes persisting for either days or weeks. Jin *et al.* (2003) also made measurements and found that 95% of paths had fewer than five observable daily changes. Hence, the hop-count filtering approach needs to update the IP-to-hop-count mapping table daily. Hop-count filtering is designed to filter spoofed packets. It does not prevent an attacker from flooding packets with true sources and correct TTL values. Hence, it cannot protect victims from recent DDoS attacks using bots that do not necessarily spoof source addresses.

IV.D. Defenses in Transit Networks

DDoS defense mechanisms deployed at the intermediate network provide infrastructural defense to a large number of Internet hosts. They usually ask routers to collaboratively monitor and exchange certain information to defend against DDoS attacks. Distributed congestion puzzle (Wang *et al.* 2004), pushback (Mahajan *et al.* 2002, Ioannidis *et al.* 2002) and traceback (Yaar *et al.* 2005, Savage *et al.* 2000, Snoeren *et al.* 2001) and capability filtering (Anderson *et al* 2003, Yaar *et al.* 2004, Yang *et al.* 2005) techniques are representative intermediate network mechanisms.

Defenses Using Puzzle

Client puzzles have been proposed to defend against DoS attacks in TCP (Juels *et al.* 1999), SYN cookie (Bernstein 1996), authentication protocols (Aura *et al.* 2000), TLS (Dean *et al.* 2001), graphic Turing test (Kandula *et al.* 2005), application traffic control (Walfish *et al*. 2006), etc. In brief, when a client requests a service, the server first asks the client to solve a problem before accepting the service request. Then, the client needs to send an answer back to the server. Usually, solving the problem takes a certain amount of computing resource, while resource demand for verifying an answer is negligible. The server will discard service requests if either the client does not reply or the answer is incorrect. Most of these protocols only involve clients and servers in the puzzle solving mechanisms. In contrast, Wang *et al.* (2004) proposed a distributed puzzle mechanism (DPM) to defend against bandwidth-exhaustion DDoS attacks in which routers take the main role in solving puzzles and verifying puzzle solutions. In DPM, a typical puzzle is composed of a moderately-hard function (such as hash function MD5); solving the puzzle requires a brute-force search in the solution space. The hash takes three parameters: a server nonce Ns created by the congested router, a client nonce Nc created by the client, and a solution X computed by the client. The solution of the puzzle satisfies that the first *d* bits of *h*(Ns,Nc,X) are zeros. *d* is called the puzzle difficulty. DPM requests routers on the puzzle distribution paths to generate their own path nonces and attach them to the congestion notification during the puzzle distribution phase. By using path nonces, DPM gives different responders different puzzles (nonce sequences), thus preventing the adversary from replaying the solutions via different paths. For example, in Figure 6, R2 and R3 treat NsN1 as the server nonce and append their own nonces to the nonce sequence; R1 treats N2Na and N3Nb as the client nonces from R2 and R3. Once a link adjacent to a router is congested, the router requires the clients to solve the puzzle and thereby imposes a computational burden on clients who transmit via this router. The computation demand is tied to the bandwidth consumed by a puzzle-based rate-limiter implemented in the router.

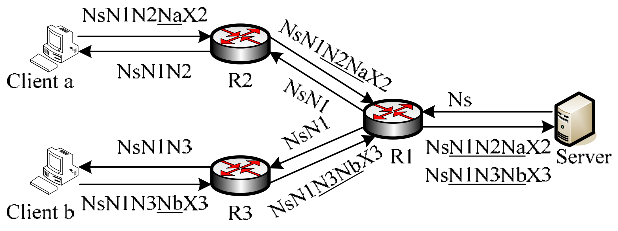

Figure 6 Distributed Congestion Puzzle

DPM requires a participating router to keep puzzles for a period to verify solutions and prevent reusing puzzles. Hence, it might be demanding for a high throughput router to be equipped with a large memory for keeping nonces and a fast CPU for verifying solutions. Probabilistic verification is applied to reduce the computational load of verification. Based on the arriving packet rate, a DPM router dynamically adjusts the sampling rate to selectively verify solution packets and prevents the router's CPU from being overloaded. In addition, a DPM router uses bloom filters to reduce the memory demand from 80MBytes to 1MBytes. Although DPM distributes computation loads to the clients via intermediate routers, it still causes unfairness among traffic because powerful clients need less time to solve a puzzle and make a DPM router pass more traffic. DPM routers may themselves suffer from DoS attacks, because they need to maintain puzzles and status of connections. Attackers can deplete the routers' resources via flooding bogus puzzles. DPM, similar to other puzzle solving defense approaches, is not effective in defending against connectionless flooding that does not need interaction between clients and servers.

Pubshback

Mahajan *et al.* (2002) proposed aggregate-based congestion control (ACC) method that manages packet flows at a finer granularity. An aggregate is defined as a collection of packets that share some properties. ACC provides a mechanism for detecting and controlling aggregates at a router using attack signatures and a pushback mechanism to propagate aggregate control requests (and attack signatures) to upstream routers. ACC mechanisms are triggered when a router experiences sustained congestion. The router first attempts to identify the aggregates responsible for the congestion. Then, the router asks its adjacent upstream routers to rate-limit the aggregates. Since the neighbors sending more traffic within the aggregate are more likely to be carrying attack traffic, this request is sent only to the neighbors that send a significant fraction of the aggregate traffic. The receiving routers can recursively propagate pushback further upstream. As illustrated in Figure 7, assuming L0 is highly congested due to a high-bandwidth aggregate, and R0 identifies the responsible aggregate L2 and L3. R0 can then pushback to R2 and R3 and subsequently to R4 and R7. Therefore, traffic from L1, L5 and L6 are protected.

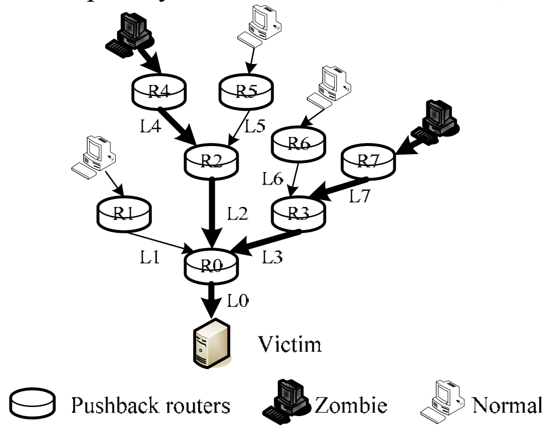

Figure 7 Aggregate-based Congestion Control

In pushback, the ACC mechanism divides the rate limit among the contributing neighbors. In general, the contributing neighbors do not contribute the same amount. It is challenging to determine which contributing neighbor and how much should be rate-limited. The heuristic of ACC is that the link which carries more traffic in the aggregate is more likely to be sending attack traffic. Hence, more traffic should be dropped from it. After computing the limit for each contributing neighbor, a pushback request message is sent to them. The recipients begin rate-limiting the aggregate with the specified limit. Therefore, pushback not only saves upstream bandwidth through early dropping of packets that would be dropped downstream at the congested router, but also helps to focus rate-limiting on the attack traffic within the aggregate. Note that the enforcement of pushback will change the traffic distribution in upstream routers. Thereby, a dynamic adjustment of rate-limiting should be integrated into ACC. However, pushback routers need to be continuously deployed in the Internet. Rate limit requests cannot be pushed back to legacy routers that do not understand the pushback approach. Pushback cannot protect legitimate traffic sharing the same path of attacking traffic either, because ACC does not distinguish traffic in the same path.

IP Traceback

Packets forwarded by routers can carry information to help victims reconstruct the attack path. Routers can add extra information as a header or an extra payload to packets so that the border router of the victim can identify the routes close to the attacking sources and ask these routers to filter flooding packets (Argyraki *et al.* 2005). Routers can also embed information into IP headers (Dean *et al.* 2002, Savage *et al.* 2000, Snoeren *et al.* 2001, Song *et al.* 2001, Yaar *et al.* 2003 2005, Aljifri *et al.* 2003) to help victims trace back the true attacking sources. FIT (Yaar *et al.* 2005) has recently been proposed as one of the probabilistic packet marking traceback schemes and consists of two major parts: a packet marking scheme to be deployed at routers, and path reconstruction algorithms used by end hosts receiving packet markings. As other probabilistic marking schemes, FIT requires routers to mark the 16 bit IP Identification (IP ID) field of the IPv4 header of a small percentage of the packets that they forward. An FIT router marks a forwarded packet with a certain probability, *q* (0.04 for the best marking over 20 hops), which is a global constant among all FIT enabled routers. As depicted in Figure 8, FIT packet markings contain three elements: a fragment of the hash of the marking router's IP address (hash_frag), the number of the hash fragment marked in the packet (frag#), and a distance field (*b*). Each FIT router pre-calculates a hash of its IP address and splits the hash into *n* fragment (e.g., *n*=4 in the Figure). When marking a packet, a router randomly selects a fragment of its IP address hash to mark into the hash_frag field, and its corresponding fragment number into the frag# field. Then, the router also sets the 5 least significant bits of the packet's TTL to a global constant *c*, and stores the 6th bit of the TTL in the distance field *b*. Yaar *et al.* (2005) shows that *c*=22 is the best parameter for traceback. This last step allows the next FIT-enabled router, or the packet receiver, to determine the distance from the router's mark.

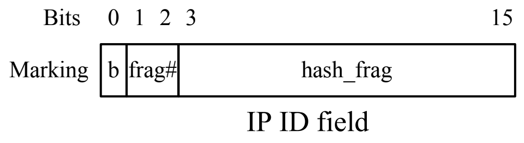

Figure 8 FIT Packet Marking

In FIT, an end host first computes hashes of all IP addresses, which can be done in hours. Then, the end host obtains distances and IP hashes of upstream routers from marked packets. Therefore, the end host can set up a table (map) of IP, hash and distance of upstream routers. The map is normally constructed prior to the attack and updated according to normal incoming traffic. When an attack occurs, the end host obtains the hash segment and the distance from a marked packet to determine from which router the packet comes. FIT uses node marking instead of the commonly used edge marking (Dean *et al.* 2002, Savage *et al.* 2000, Snoeren *et al.* 2001). Hence, FIT avoids incrementally reconstructing the attack paths. Node marking also enables the end host to reconstruct the attack paths from much fewer packets (hundreds in contrast to thousands in edge marking). Furthermore, an FIT router does not require the next router to be an FIT router. Therefore, FIT is an incrementally deployable scheme. However, FIT does not stop an ongoing attack. When true attacking sources are identified, attacking traffic should be filtered. At the victim side, it is impossible to filter attacking packets, because they may not carry true sources. Hence, victims will need to ask routers closing to the true attacking sources to filter the attacking traffic or stop attacking hosts. Unfortunately, such counterattack would need a complex and coordinated defense system deployed throughout the Internet.

Capability Filtering

The design principles of the Internet overlooked the receivers' capability. In many cases, routers are designed to forward packets from senders, regardless of whether or not the packets are desired by receivers. Although several aforementioned approaches (Mahajan *et al.* 2002, Argyraki *et al.* 2005, Li *et al.* 2002) have been proposed to filter unwanted packets at upstream routers before they can consume the resources of destinations, these approaches act without knowing the allowance of destinations indeed. As a consequence, these approaches may block legitimate traffic, because routers cannot discriminate attacking traffic from other traffic, especially when attackers can compose packets with contents of their choosing. To solve this problem, researchers proposed the approach of putting a token of capability into each data packet to prove that packets were requested by receivers (Anderson *et al* 2003, Yaar *et al.* 2004, Yang *et al.* 2005). This category of filtering approaches takes two steps. First, a sender requests permission to send; after verifying the sender is valid, the receiver provides a token of capability. Then, routers along the path from the sender to the receiver can verify packets carrying the token and forward the packets with valid tokens.

In the first step (capability handshake), capabilities are obtained using request packets that do not have capabilities. A request is sent as a part of the TCP SYN packet. If the request is authorized, a token of capability is piggybacked in the TCP SYN/ACK packet. Thereby, routers are also notified of the capability. Once the sender receives the capability, the communication is bootstrapped. To identify paths, each router at the ingress of a trust boundary tags requests with a unique value (flow nonce) derived from its incoming interface. Routers not at trust boundaries do not tag requests, but fair-queue requests using their tags as unique path identities. Thereby, an attacker that composes arbitrary requests can, at most, cause contention in one queue that is corresponding to the edge router of the attacker. At the same time, request packets only compromise a small fraction of traffic. Hence, it is difficult for attackers to manipulate the capability handshake step.
Capability is bound to a specific network path, including source and destination addresses, in a specific duration. Each router that forwards a request packet generates its own pre-capability and attaches it to the packet. The pre-capability consists of a timestamp and a cryptographic hash of the timestamp, the source and destination IP addresses, and a secret known only to the router. Each router can verify for itself that a purported pre-capability attached to a packet is valid by re-computing the hash. The pre-capability is hard to forge without knowing the router secret. The destination receives an ordered list of pre-capabilities that corresponds to a specific network path with fixed source and destination IP endpoints. The destination adopts fine-grained capabilities that grant the right to send up to N bytes along a path within the next T seconds. The destination converts the pre-capabilities it receives from routers to full capabilities by hashing them with N and T. If the destination wishes to authorize the request, it returns the ordered list of full capability, together with N and T, to the sender. Routers along the path verify their portion of the capabilities by re-computing the hashes and the validity of time. Routers cache their capabilities and the flow nonce for verification later. For longer flows, senders should renew these capabilities before they reach their limits. Finally, all packets carry a capability header that is implemented as a shim layer above IP so that there are no separate capability packets. Regular packets have two formats: packets that carry both a flow nonce and a list of valid capabilities and packets that carry only a flow nonce. A regular packet with a list of capabilities may be used to request a new set of capabilities. A few problems might exist due to the dynamic nature of the Internet. First, routes could change. Packets in this case are handled by demoting them to low priority as legacy traffic. These packets are only likely to reach the destination when there is little congestion. Second, routes could be asymmetric, and thus capability handshake cannot be achieved and communication cannot be established. Third, the approach is not applicable when partially deployed in the Internet, because a router without implementing capability filtering still forwards as many packets as possible.

IV.E. Defenses Using Overlay Networks

Aforementioned defense technologies require homogeneous system design. However, no single defense mechanism can guarantee wide deployment, as deployment depends on market conditions and social aspects. Therefore, Mirkovic *et al.* (2003) proposed Defensive Cooperative Overlay Mesh (DefCOM) as a new deployment paradigm, which uses a peer-to-peer network to integrate heterogeneous systems. Nodes in DefCOM are classified as three types: alert generator nodes that detect attacks and deliver attack signatures to the rest of the peer network; core nodes that rate-limit high-volume transit traffic matching signatures; and classifiers that perform selective rate-limiting, differentiate between legitimate flows and attack flows, and cooperate with other defense nodes to ensure preferential service for legitimate traffic. As demonstrated in

Figure 9, alert generator node R0 detects a DDoS attack on the victim, and informs all other defense nodes through the DefCOM network. The notified nodes then identify upstream-downstream relationship between peers and cooperate to trace out the topology of the victim-rooted traffic tree. Knowing the tree, the classifiers can differentiate legitimate traffic from attack streams and devise the appropriate rate-limits to restrain the attack traffic. DefCOM allows heterogeneous systems to cooperate to achieve a better overall defense, instead of every defense system operating in isolation. Each system would autonomously perform the defense functions that it is best at and compensate for its weaker traits through cooperation with other systems. Division of work would enable nodes to become more specialized, leading to better overall performance. The wide deployment necessary to handle diffuse attacks would be achieved by incorporating existing defense nodes in the framework. As the attacks evolve, new systems could join and either replace or enhance the functionality of the old ones. Nevertheless, the design of interface in the overlay network for different defense systems to cooperate is still open.

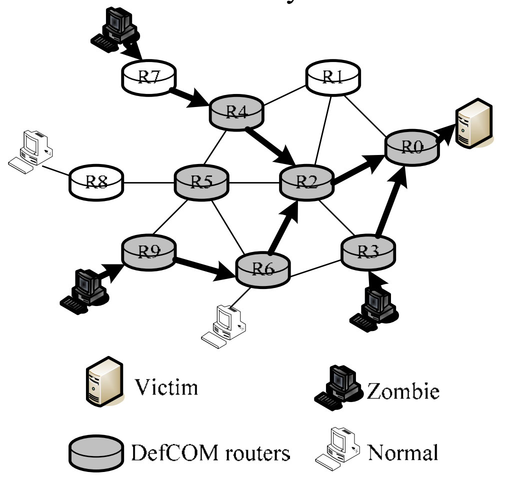

Figure 9 Architecture of DefCOM

Overlay network also provides an infrastructure to deploy distributed DDoS defense system. Secure overlay services (SOS) (Keromytis *et al.* 2004) prevents DoS attacks on critical servers via an overlay network. In general, a defense system, such as a firewall, shall be able to filter malicious traffic. To avoid the effects of a DoS attack against the firewall connectivity, expensive processing tasks of a firewall, such as access control and cryptographic protocol handling, are distributed to a large number of nodes in an overlay network. Thus, SOS becomes a large distributed firewall that distinguishes “good” (authorized) traffic from “bad” (unauthorized) traffic. Service requests from authenticated clients are routed to the protected servers via the overlay network, while non-authenticated requests are filtered. Mayday (Andersen 2003) generalizes the idea of SOS and uses a distributed set of overlay nodes that are trusted (or semi-trusted) to distinguish legitimate traffic from attack traffic. To protect a server from DDoS traffic, Mayday prevents general Internet hosts from communicating directly with the server by imposing a router-based, network-layer filter ring around the server. Clients communicate with overlay nodes that verify whether a client is permitted to use the service. These overlay nodes then use an easily implemented lightweight authenticator to get through the filter ring. Within this framework, SOS represents a particular choice of authenticator and overlay routing, using distributed hash table lookups for routing among overlay nodes and using the source address of the overlay node as the authenticator. Since these overlay based defense techniques work at the application layer, their deployment requires client systems to modify the access model of applications so that client systems are aware of the overlay and use it to access the target.

V. DOS ATTACKS AND DEFENSES IN WIRELESS NETWORKS

Wireless networks are taking a more important role today. As these networks gain popularity, providing security and trustworthiness will become an issue of critical importance. We discuss DoS attacks and defense in wireless networks according to the three lower network layers in this section. Attacks at and above the transport layer in wireless network are similar to attacks in the Internet. Note that our focus is on IEEE 802.11 based wireless networks, which is a family of physical and MAC protocols used in many hotspot, ad hoc and sensor networks.

V.A. Physical Layer Attacks and Defenses

The shared nature of the wireless medium allows attackers to easily observe communications between wireless devices and launch simple DoS attacks against wireless networks by jamming or interfering communication. Such attacks in the physical layer cannot be addressed through conventional security mechanisms. An attacker can simply disregard the medium access protocol and continually transmit in a wireless channel. By doing so, the attacker either prevents users from being able to commit legitimate MAC operations or introduces packet collisions that force repeated backoffs. Xu *et al.* (2005) studied four jamming attack models that can be used by an adversary to disable the operation of a wireless network and observed that signal strength and carrier sensing time are unable to conclusively detect the presence of a jammer. It is also found that packet delivery ratio may help differentiate between congestion and jamming scenarios but cannot help defenders to conclude whether poor link utility is due to jamming or the mobility of nodes. To address the jamming attacks, Xu *et al.* (2005) proposed two enhanced detection protocols. One scheme employs signal strength measurements as a reactive consistency check for poor packet delivery ratios, and the other employs location information to serve as the consistency check.

V.B. MAC Layer Attacks and Defenses

In the MAC layer, the defects of MAC protocol messages and procedures of a MAC protocol can be exploited by attackers. Bellardo *et al.* (2003) discussed vulnerabilities on authentication and carrier sense and showed that attackers can provide bogus information or misuse the carrier sense mechanism to deceive normal nodes. For example, attackers can forge deauthentication or disassociation packets to break the connections between nodes and access points or send Ready-To-Send and Clear-To-Send packets with a forged duration to suppress the transmission of nearby nodes. Attackers can disobey the backoff procedure so that they always get the first chance to send RTS right after the end of last transmission. Gu *et al.* (2004) analyzed how the attackers can exhaust bandwidth by using large junk packets without breaking the MAC protocol. The attackers can use certain packet generation and transmission behavior to obtain more bandwidth than other normal nodes. Wullems *et al.* (2004) identified that attackers can exploit the Clear Channel Assessment function of the 802.11 protocol to suppress other nodes with the illusion of a busy channel. Bellardo *et al.* (2003) suggested that extending explicit authentication to 802.11 control packets would be a solution to prevent attackers from easily forging the protocol control packets.

V.C. Networking Layer Attacks and Defenses

DoS attacks in the network layer mainly focus on exploiting routing and forwarding protocols in wireless networks. In particular, ad hoc and sensor networks are susceptible to these attacks. It is also noticed that network layer DoS attacks in wireless networks are very different from the attacks in the Internet. Because routers in wireless networks are computers that could be compromised as end hosts, network layer DoS attacks in wireless networks can be launched by any computer in the network. In addition, DoS defense techniques in the Internet that demand the cooperation of routers are no longer valid.

Routing Attacks and Defenses

Researchers have shown that attackers can manipulate ad hoc network routing protocols (such as DSR (Johnson *et al.* 2002) and AODV (Perkins *et al.* 2002)) to break valid routes and connections (Hu *et al.* 2002, 2003). For example, if attackers change the destination IP address in the route request message, the victim cannot establish a route to its destination and thus cannot access services. Hence, the security of routing protocols is the solution for defending routing-based DoS attacks. In order to prevent attackers from exploiting the security flaws in routing protocols, several secure routing protocols have been proposed to protect the routing messages and thus prevent DoS attacks. For example, Hu *et al.* (2002) proposed to use TESLA (Perrig *et al.* 2002), which is a symmetric broadcast authentication protocol, in routing discovery to secure routing protocols. When a source sends a route request, it authenticates the request with its own TESLA to prevent attackers from forging or modifying the request.

Forwarding Attacks and Defenses

Similar to routing attacks, attackers can also exploit forwarding behavior. Typical attacking approaches include injecting junk packets, dropping packets, and disorder packets in legitimate packets. Attackers can use spoofed packets to disguise their attacking behavior, or find partners to deceive defenders. The objective is to exhaust bandwidth (Gu *et al.* 2005) or disrupt connection (Aad *et al.* 2004) so that service cannot be delivered.

To prevent attackers from spoofing and flooding packets in wireless networks, hop-by-hop source authentication is needed so that every node participates in the protection of the network. Ye *et al.* (2004) proposed a statistic filtering scheme that allows en route nodes to filter out false data packets with some probability. This approach will not filter packets that do not carry keys that the en route nodes have, but will discard them at the destination. Zhu *et al.* (2004) proposed an interleaved hop-by-hop authentication scheme that guarantees that false data packets will be detected and dropped within a certain number of hops, although the scheme does not tolerate the change of routers. Gu *et al.* (2005) proposed another hop-by-hop source authentication approach with a higher overhead to ensure that a packet can be verified when a route is changed due to unreliability in wireless networks. In this approach, the routing node at which a new route diverges from the old route takes the responsibility of authenticating the packets. The routing nodes in the new route can then verify the packets based on the new authentication information.

VI. CONCLUSION

In this article, we overviewed existing DoS attacks and defense technologies in the Internet and wireless networks. DoS attackers exploit flaws in protocols and systems to deny access of target services. Attackers also control a large number of compromised hosts to launch DDoS attacks. Simply securing servers are no longer enough to make service available under attack, since DoS attack techniques are more complicated and many unwitting hosts are involved in DoS attacks. By reviewing several existing DoS attack techniques and classifying them, this chapter highlighted challenges of DoS defense from characteristics of DoS attacks. For defenders, it is difficult to decide whether a packet is spoofed, to prevent a host from being compromised and controlled, to ask upstream routers to filter unwanted traffic, and to keep defenders themselves from DoS attacks.

This article reviews current DoS defense solutions in deployment and in research. They are trying to address one or several problems in DoS defense, for instance, how to distinguish

legitimate traffic from flooding traffic, how to identify or trace true attacking hosts, how to filter or control flooding traffic as close as possible to attacking sources, and how to allow routers to collaborate in defense. These solutions can handle some but not all DoS attacks due to their design principles, deployment issues, etc. New DoS attack techniques may also invalidate these solutions. This chapter acknowledged these innovative ideas and provided a fundamental understanding for developing new solutions.
Different from the Internet, wireless networks have their own unique DoS attacks due to the fact that wireless is an open communication approach and mobile devices can function as routers in wireless networks. DoS attacks in wireless networks extend to the scope not viable in the Internet. The existing defense approaches illustrate that security countermeasures should be studied and incorporated into wireless protocols at lower layers, and mobile hosts should actively and collaboratively participate in protecting their wireless networks.

<h2> Acknowledgement

Peng Liu was supported in part by NSF and DHS through NRT-0335241.

VIII. Glossary

Access control A range of security methods to permit or deny the use of an object by a subject. An object refers to a resource in a system. A subject refers to an entity that performan actions. Access control methods decide whether a subject can access an object based on the subject's identity and a predefined access control matrix.
Ad hoc network A type of network that does not have an infrastructure. Normally, a host in the network should forward packets as a router even though the packets are neither originated nor destined to the host. The host can also ask other hosts to forward its packet via routing protocols.
CTS In an 802.11 network, a receiving computer sends a clear-to-send (CTS) packet to a sending computer to indicate the shared wireless channel is cleared for transmission and inform all other computers in the receiver's transmission range to suspend transmission to avoid collision.
Bot A type of malicious program that functions as a backdoor in the host and accepts commands from its master. It is more organized than other malicious programs in that many bots often form a network via certain communication mechanisms. Such a network is called BotNet.
DoS A type of attack that targets disabling service, denying service access, or downgrading service performance. DoS attack techniques vary as long as the attack objective can be achieved. Many recent DoS attacks try to exhaust resources for providing services.
MD5 A cryptographic hash function designed by Ronald Rivest in 1991. The algorithm processes a variable-length message and generates a fixed-length hash output of 128 bits.
Overlay Network A network that runs above the network layer and consists of end hosts forwarding packets as routers. The underlying communication between end hosts is unicast, even when an end host communicates with several other end hosts. The network is normally application-oriented.
RSA A public key encryption/decryption algorithm invented by Ron Rivest, Adi Shamir and Leonard Adleman in 1977. RSA are the initials of their last names.
RTS. In an 802.11 network, a sending computer sends a ready-to-send (RTS) packet to inform a receiving computer that a data packet is ready for transmission. The RTS packet is used for competing the wireless transmission channel that is shared with other computers.

Scan In a network, scanning refers to a technique that a computer sends a number of packets with various destination addresses and then gathers information of computers at the destination addresses according to received response packets.
Spoof A technique to replace the originator's identity with another identity. Normally, in a network attack, an attacker puts another (any) address as the source address field in the IP header of an attack packet.
TTL Time-To-Live field in the IP header of a packet. It is decremented by a router when the packet goes through it. It is used to prevent the packet from staying forever in the Internet because the packet will be discarded when its TTL is reduced to 0.
Worm A type of malicious program that can be propagated by itself from one infected host to another vulnerable host. When a host is infected by a worm, the worm starts to scan other vulnerable hosts. Once a vulnerable host is found, it compromises the host by exploiting the vulnerability, and copies its code to the host. Then, the worm in the new infected host starts its own propagation.

IX. Cross References

1. Aad, I., Hubaux, J.P., and Knightly, E. (2004). Denial of service resilience in ad hoc networks. Proceedings of ACM Mobicom. ACM Press, New York.
2. Aljifri, H., Smets, M., and Pons A. (2003). IP Traceback using header compression. Computers & Security, Vol. 22(2), pp. 136-151.
3. Andersen, D. G. (2003). Mayday: distributed filtering for Internet services. Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems. USENIX Press, Berkeley, CA.
4. Anderson, T., Roscoe, T., and Wetherall, D. (2003). Preventing Internet Denial of Service with Capabilities. Proceedings of HotNets-II. ACM Press, New York.
5. Argyraki, K., and Cheriton, D. R. (2005). Active Internet traffic filtering: real-time response to denial-of-service attacks. Proceedings of the USENIX Annual Technical Conference. USENIX Press, Berkeley, CA.
6. Aura, T., Nikander, P., and Leiwo, J. (2000). Dos-resistant authentication with client puzzles. Proceedings of Security Protocols Workshop, Lecture Notes in Computer Science, 213(33), 170-181.
7. Bellardo, J., and Savage, S. (2003) 802.11 denial-of-service attacks: real vulnerabilities and practical solutions. Proceedings of USENIX Security Symposium, 15-28. USENIX Press, Berkeley, CA.
8. Bernstein, D. J. (1996) SYN cookies. Available at: http://cr.yp.to/syncookies.html. (Date of access: October 31, 2006)
9. Bro (2006) Bro Intrusion Detection System. Available at: http://bro-ids.org/. (Date of access: October 31, 2006)
10. CERT (1996). CERT Advisory CA-1996-21 TCP SYN flooding and IP spoofing attacks. Available at: http://www.cert.org/advisories/CA-1996-21.html. (Date of access: January 2, 2006)
11. CERT (1997). CERT Advisory CA-1997-28 IP Denial-of-Service Attacks. Available at: http://www.cert.org/advisories/CA-1997-28.html. (Date of access: August 20, 2006)
12. CERT (1998). CERT Advisory CA-1998-01 Smurf IP denial-of-service attacks. Available at: http://www.cert.org/advisories/CA-1998-01.html. (Date of access: January 2, 2006)

13. CERT (2004). Technical Cyber Security Alert TA04-028A, W32/MyDoom.B virus. Available at: http://www.us-cert.gov/cas/techalerts/TA04-028A.html. (Date of access: January 2, 2006)
14. CERT (2006). CERT/CC Statistics 1988-2005. Available at http://www.cert.org/stats/#incidents. (Date of access: January 2, 2006)
15. Cheswick, B., Burch, H., and Branigan, S. (2000). Mapping and visualizing the internet. Proceedings of USENIX Annual Technical Conference. USENIX Press, Berkeley, CA.
16. CISCO (2001). Dealing with malloc fail and high CPU utilization resulting from the code red worm. Available at: http://www.cisco.com/warp/public/63/ts_codred_worm.shtml. (Date of access: April 15, 2006)
17. CISCO (2006a). CISCO IOS NetFlow. Available at: http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html. (Date of access: August 20, 2006)
18. CISCO (2006b). CISCO PIX 500 Series Security Appliances. Available at: http://www.cisco.com/en/US/products/hw/vpndevc/ps2030/index.html. (Date of access: August 20, 2006)
19. CISCO (2006c). Cisco Guard DDoS Mitigation Appliances. Available at: http://www.cisco.com/en/US/products/ps5888/index.html. (Date of access: August 20, 2006)
20. Cowan, C., Beattie, S., Johansen, J., and Wagle. P. (2003). PointGuard: Protecting pointers from buffer overflow vulnerabilities. Proceedings of the 12th USENIX Security Symposium. USENIX Press, Berkeley, CA.
21. Crosby, S. A., and Wallach, D. S. (2003). Denial of Service via Algorithmic Complexity Attacks. Proceedings of the 12th USENIX Security Symposium, 29-44. USENIX Press, Berkeley, CA.
22. Dean, D., and Stubblefield, A. (2001). Using client puzzles to protect TLS. Proceedings of the 10th Annual USENIX Security Symposium. USENIX Press, Berkeley, CA.
23. Dean, D., Franklin, M., Stubblefield, A. (2002). An algebraic approach to IP traceback. ACM Transactions on Information and System Security, 5(2), 119-137.
24. DETER (2005). Cyber Defense Technology Experimental Research. Available at: http://www.isi.edu/deter/. (Date of access: January 2, 2006)
25. Dittrich, D. (1999). The DoS Project's "trinoo" distributed denial of service attack tool. Available at: http://staff.washington.edu/dittrich/misc/trinoo.analysis. (Date of access: January 2, 2006)
26. EMIST (2005). Evaluation Methods for Internet Security Technology. Available at: http://www.isi.edu/deter/emist.temp.html. (Date of access: January 2, 2006)
27. Ferguson, P., and Senie, D. (1998). RFC 2267 - Network ingress filtering: defeating denial of service attacks which employ IP source address spoofing. Availabe at: http://www.faqs.org/rfcs/rfc2267.html. (Date of access: January 2, 2006)
28. Gil, T. M., and Poletter, M. (2001). Multops: a data-structure for bandwidth attack detection. Proceedings of USENIX Security Symposium. USENIX Press, Berkeley, CA.
29. Gu, Q., Liu, P., and Chu, C. (2004). Tactical bandwidth exhaustion in ad hoc networks. Proceedings of the 5th Annual IEEE Information Assurance Workshop, 257-264. IEEE Press, New York.
30. Gu, Q., Liu, P., Zhu, S., and Chu, C. H. (2005). Defending against packet injection attacks in unreliable ad hoc networks. Proceedings of IEEE Globecom. IEEE Press, New York.

31. Guirguis, M., Bestavros A., Matta I., and Zhang Y. (2005). Reduction of Quality (RoQ) Attacks on Internet End-Systems. Proceedings of INFOCOM. IEEE Press, New York.
32. Hamadeh, I., Hart, J., Kesidis, G., and Pothamsetty, V. (2005). A preliminary simulation of the effect of scanning worm activity on multicast, Proceedings of the 19th Workshop on Principles of Advanced and Distributed Simulation. ACM Press, New York.
33. Handley, M. and Rescorla, E. (2006). Internet Denial of Service Considerations. Available at: http://tools.ietf.org/html/draft-iab-dos-05. (Date of access: October 31, 2006)
34. Honeynet (2005). Know your enemy: tracking botnets. Available at: http://www.honeynet.org/papers/bots/. (Date of access: January 2, 2006)
35. Houle, K. J., and Weaver G. M. (2001). Trends in denial of service attack technology. Available at: http://www.cert.org/archive/pdf/DoS_trends.pdf. (Date of access: January 2, 2006).
36. Hu, Y. C., Perrig, A, Johnson, D. B. (2002). Ariadne: a secure on-demand routing protocol for ad hoc networks. Proceedings of ACM Mobicom, 12-23. ACM Press, New York.
37. Hu, Y. C., Perrig, A, Johnson, D. B. (2003). SEAD: secure efficient distance vector routing for mobile wireless ad hoc networks. Ad Hoc Networks, 1(1), 175-192.
38. Ioannidis, J., Bellovin, S. M. (2002). Implementing pushback: router-based defense against DDoS attacks. Proceedings of NDSS. The Internet Society, Reston, VA.
39. Jian, W. (2000). A possible LAST_ACK DoS attack and fix. Available at: http://www.ussg.iu.edu/hypermail/linux/kernel/0004.1/0105.html. (Date of access: October 31, 2006)
40. Jin, C., Wang, H., and Shin, K. G. (2003). Hop-count filtering: an effective defense against spoofed DDoS traffic. Proceedings of ACM CCS, 30-41. ACM Press, New York.
41. Johnson, D., Maltz, D., Hu, Y. C., and Jetcheva, J. (2002) The dynamic source routing protocol for mobile ad hoc networks (DSR). IETF Internet draft, draft-ietf-manet-dsr-09.txt.
42. Juels, A., and Brainard, J. (1999). Client puzzle: a cryptographic defense against connection depletion attacks. Proceedings of NDSS, 151-165. The Internet Society, Reston, VA.
43. Kandula S., Katabi D., Jacob M., and BergerA. (2005).Botz-4-Sale: surviging organized DDoS attacks that mimic flash crowds. Proceedings of the 2nd Symposium on Networked Systems Design and Implementation. USENIX Press, Berkeley, CA.
44. Keromytis, A. D., Misra, V, and Rubenstein D. (2004). SOS: an architecture for mitigating DDoS attacks. IEEE Journal on Selected Areas of Communications, 33(3), 413-426.
45. Kuzmanovic, A., and Knightly, E. W. (2003). Low-rate TCP-targeted denial of service attacks, Proceedings of ACM SIGCOMM, 75-86. ACM Press, New York.
46. Li, J., Mirkovic, J., Wang, M., Reiher, P., and Zhang, L. (2002). Save: source address validity enforcement protocol. Proceedings of IEEE Infocom, 3, 1557-1566. IEEE Press, New York.
47. MAZU (2006). MAZU Enforcer. Available at: http://www.mazunetworks.com/products/mazu-enforcer.php. (Date of access: January 2, 2006)
48. Mahajan, R., Bellovin, S. M., Floyd, S., Ioannidis, J., Paxson, V., and Shenker, S. (2002). Controlling high bandwidth aggregates in the network. ACM SIGCOMM Computer Communications Review, 32(3), 62-73.

49. Mirkovic, J., Prier, G., and Reiher, P. (2002). Attacking DDoS at the source. Proceedings of IEEE ICNP, 312-321. IEEE Press, New York.
50. Mirkovic, J., Robinson, M., Reiher, P., and Kuenning, G. (2003). Alliance formation for DDoS defense. Proceedings of the New Security Paradigms Workshop in ACM SIGSAC. ACM Press, New York.
51. Mirkovic, J., and Reiher, P. (2004). A taxonomy of DDoS attack and DDoS defense mechanisms. ACM SIGCOMM Computer Communication Review, 34(2), 39-53.
52. Mirkovic, J., Dietrich, S., Dittrich, D., Reiher, P. (2005). Internet Denial of Service: Attack and Defense Mechanisms. Prentice Hall.
53. Pappalardo, D. and Messmer, E. (2005). Extortion via DDoS on the rise. Available at: http://www.techworld.com/security/features/index.cfm?featureid=1452. (Date of access: October 31, 2006)
54. Park, K., and Lee, H. (2001). On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law Internets. Proceedings of ACM SIGCOMM. ACM Press, New York.
55. Paxson, V. (1997). End-to-end routing behavior on the Internet. IEEE/ACM Transactions on Networking, 5(5).
56. Perkins, C.E., Royer, E.M., and Das, S. R. (2002) Ad hoc on-demand distance vector (AODV) routing. IETF Internet draft, draft-ietf-manet-aodv-11.txt.
57. Perrig, A., Canetti, R., Tygar, D., and Song, D. (2002). The TESLA broadcast authentication protocol. Cryptobytes, 5(2), 2-13.
58. Savage, S., Wetherall, D., Karlin, A., andAnderson, T. (2000). Practical network support for ip traceback. Proceedings of ACM SIGCOMM, 295-306. ACM Press, New York.
59. Shannon, C., Moore, D., and Claffy, K.C. (2002) Beyond folklore; observations on fragmented traffic. IEEE/ACM Transactions on Networking, 10(6), 709-720.
60. Snoeren, A. C., Partridge, C., Sanchez, L. A., Jones, C. E., Tchakountio, F., Kent, S. T., and Strayer, W. T. (2001). Hash-based ip traceback. Proceedings of ACM SIGCOMM. ACM Press, New York.
61. Song, D., and Perrig, A. (2001). Advanced and authenticated marking schemes for IP traceback. Proceedings of IEEE INFOCOM. IEEE Press, New York.
62. Staniford, S., Paxson, V., and Weaver, N. (2002). How to own the internet in your spare time. Proceedings of the 11th USENIX Security Symposium, 207-213. USENIX Press, Berkeley, CA.
63. Thomas, R., Mark, B., Johnson, T., and Croall, J. (2003) NetBouncer: Client-legitimacy-based High-performance DDoS Filtering. Proceedings of DARPA Information Survivability Conference and Exposition, Vol. I, pp. 14.
64. Walfish M. Vutukuru M., Balakrishnan H., Karger D., and Shenker S. (2006) DDoS Defense by Offense. Proceedings of ACM SIGCOMM. ACM Press, New York.
65. Wang, H., Zhang, D., and Shin, K. G. (2002). Detecting SYN flooding attacks. Proceedings of IEEE Infocom. IEEE Press, New York.
66. Wang, X., and Reiter, M. K. (2004). Mitigating bandwidth-exhaustion attacks using congestion puzzles. Proceedings of ACM CCS, 257-267. ACM Press, New York.
67. Wullems, C., Tham, K., Smith, J, and Looi, M. (2004). Technical summary of denial of service attack against IEEE 802.11 DSSS based wireless LANs. Available at: http://www.isi.qut.edu.au/research/publications/technical/wlan.php. (Date of access: January 2, 2006)

68. Xu, W., Trappe, W., Zhang, Y., and Wood, T. (2005). The feasibility of launching and detecting jamming attacks in wireless networks. Proceeding of ACM Mobihoc, 46-57. ACM Press, New York.
69. Yaar, A., Perrig, A., and Song, D. (2003). Pi: a path identification mechanism to defend against DoS attacks. Proceedings of IEEE Symposium on Security and Privacy.
70. Yaar, A., Perrig, A., and Song, D. (2004). SIFF: a stateless internet flow filter to mitigate DDoS flooding attacks. Proceedings of IEEE Symposium on Security and Privacy. IEEE Press, New York.
71. Yaar, A., Perrig, A., and Song, D. (2005) FIT: Fast internet traceback. Proceedings of IEEE Infocom. IEEE Press, New York.
72. Yang, X., Wetherall, D., and Anderson, T. (2005). A DoS-limiting network architecture. Proceedings of ACM SIGCOMM.
73. Ye, F., Luo, H., Lu, S., and Zhang, L. (2004). Statistical en-route detection and filtering of injected false data in sensor networks. Proceedings of IEEE INFOCOM. ACM Press, New York.
74. Zhang, R., and Chen, K. (2005). Improvements on the WTLS protocol to avoid denial of service attacks. Computers & Security, Vol. 24(1), pp. 76-82.
75. Zhu, S., Setia, S., Jajodia, S., and Ning, P. (2004). An interleaved hop-by-hop authentication scheme for filtering false data in sensor networks. Proceedings of IEEE Symposium on Security and Privacy. IEEE Press, New York.