Distributed Defense Against DDoS Attacks

Jelena Mirkovic, Max Robinson, Peter Reiher, George Oikonomou

***Abstract*— Distributed denial-of-service attacks represent a major security problem. The main task of defense systems is to accurately detect these attacks and quickly respond to stop the oncoming flood. It is equally important to recognize the legitimate traffic that shares the attack signature and deliver it reliably to the victim. Unfortunately, there is no single deployment point on the attack tree that successfully meets all three requirements. Detection of the attack is most accurate close to the victim, while the response and separation of legitimate traffic from the attack traffic is most successful close to the sources. Additionally, in partial deployment cases when many potential sources do not deploy a source-end defense, adequate victim protection can only be achieved by enlisting the help of backbone routers to constrain attack traffic. These factors clearly indicate that the DDoS problem requires a distributed cooperative solution.**

We propose a distributed system for DDoS defense, called DefCOM. DefCOM nodes span source, victim and core networks and cooperate via an overlay to detect and stop attacks. Attack response is twofold: defense nodes constrain the attack traffic, relieving victim's resources; they also cooperate to detect legitimate traffic within the suspicious stream and ensure its correct delivery to the victim. DefCOM design has a solid economic model where networks deploying defense nodes directly benefit from their operation. DefCOM further offers a framework for existing security systems to join the overlay and cooperate in the defense. These features create excellent motivation for wide deployment, and the possibility of large impact on DDoS threat.

***Index Terms*— System design, Experimentation with real networks/Testbeds**

I. Introduction

Distributed denial-of-service (DDoS) attacks commonly overwhelm their victims by sending a vast amount of legitimate-like packets from multiple attack sites. As a consequence the victim spends its key resources processing the attack packets and cannot attend to its legitimate clients. During very large attacks, DDoS traffic also creates a heavy congestion in the Internet core which disrupts communication between all Internet users whose packets cross congested routers.

The only way to completely eliminate the DDoS threat is to secure all machines on the Internet against misuse, which is unrealistic. Most large sites currently handle the problem by equipping critical systems with abundant resources. While this raises the bar for the attacker, any amount of resources can be exhausted with a sufficiently strong attack. The only remaining approach is to design defenses that will detect the attack and respond to it by dropping excess traffic. An important requirement for DDoS defenses is to recognize legitimate packets in the flood, separate them from the attack and deliver them safely to the victim.

A practical DDoS defense must meet three important goals: (1) accurate attack detection, (2) effective response (dropping or rerouting) to reduce the flood, and (3) precise identification of legitimate traffic and its safe delivery to the victim. These goals are best met at diverse points on the attack tree, illustrated with a simplified attack scenario in Figure 1. The Figure shows four attackers flooding the victim over a simplistic version of a well connected core (grey nodes).

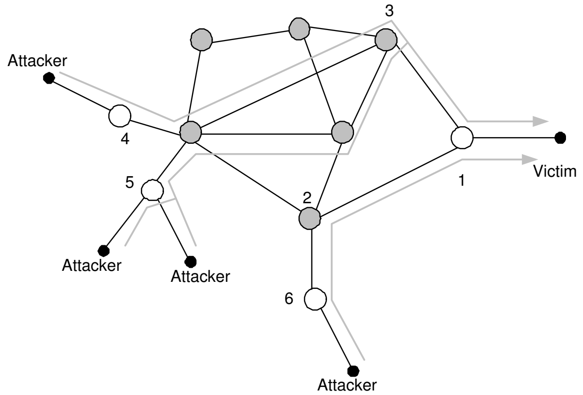


Fig. 1. Illustration of the attack scenario, and various defense deployment points. Grey nodes represent the Internet core, white nodes represent edge routers and black nodes represent end hosts.

Detection is most accurate when performed near the victim (e.g. at 1), since the defense system can closely observe the victim's performance and notice early signs of service degradation. As the deployment moves further upstream, the ability to detect attacks early (or to detect them at all) diminishes. The best strategy for *attack*

J. Mirkovic (sunshine@cis.udel.edu) and G. Oikonomou (oikonom@cis.udel.edu) are with the University of Delaware, M. Robinson (max@cs.ucla.edu) and P. Reiher (reiher@cs.ucla.edu) are with the University of California Los Angeles.

detection would then be to deploy a defense node in the vicinity of the victim.

To effectively control the flood, a defense must be able to see and police all traffic targeting the victim (illustrated by grey lines), regardless of the number and placement of the attacking machines. It also must be able to handle large floods. Victim-based defense (e.g., at node 1) easily meets the first requirement, but may be overwhelmed with high packet rates. Core-based and source-based defenses can handle large attacks but need distributed deployment to cover all attack paths to the victim. In our illustration we would need at least two defense nodes in the core[1] (for example at 2 and 3), or three source-based defenses (at 4, 5, and 6). It is worth noting that core-based defenses usually need significantly fewer deployment points to cover all attack paths than source-based defenses, due to highly interconnected topology of the Internet core. The best strategy for *effective flood control* would then be to deploy several defense nodes in the core.

Precise traffic identification is challenging due to high variability and amount of the attack traffic, and requires a lot of statistics gathering and per-packet processing. Victim-based defenses experience heavy traffic volumes during the attack, which limits their ability to profile traffic. Core routers handle high and diverse traffic continuously and have very scarce resources that cannot be dedicated to profiling. On the other hand, source-based defenses experience moderate traffic volumes, even during the attack, and thus need moderate resources for sophisticated profiling of this traffic [1], [2]. The best strategy for *traffic identification* would then be to deploy defense nodes close to the sources.

DDoS problem requires a distributed solution, in which defensive components deployed at multiple places in the Internet work together to stop the flood and deliver legitimate traffic to the victim. In this paper we propose a design and implementation of such a solution. Our system, called DefCOM (<u>Def</u>ensive <u>C</u>ooperative <u>O</u>verlay <u>M</u>esh), deploys defense nodes distributed in the Internet core and through the edge networks. All nodes form a peer-to-peer overlay to securely exchange attack-related messages. When an attack occurs, nodes close to the victim detect this and alert the rest of the DefCOM overlay. Core nodes and those in vicinity of attack sources then suppress the attack traffic through coordinated rate limiting. Source nodes are also tasked with traffic profiling, making sure that their share of limited bandwidth is fully dedicated to traffic they deem legitimate or important. They further work in concert with core nodes to ensure that this legitimate traffic is safely delivered to the victim. Like currently, an interested network would be able to guarantee continued operation during a DDoS attack by following two easy steps: (1) deploying an alert generator and (2) purchasing a rate-limiter service from each of its ISP providers. The novel contribution of DefCOM is that legitimate clients of this network can also achieve DDoS attack transparency and reach the victim anytime if they deploy a classifier node in their network.

While core nodes are likely to be offered as an infrastructure service to control flood from many attacks, victim and source nodes are deployed based on end-user threat assessments. Those networks that are interested in gaining protection from DDoS will deploy victim-side defense nodes and join the DefCOM overlay. Networks that would like to ensure good treatment of their users' traffic during an attack will deploy source-side defense nodes and also join the overlay. This yields a good economic model where networks deploying defense nodes gain direct benefit from this deployment or can charge for it (e.g., core nodes are ISPs who can sell flood control service to their customers), and promotes wide deployment.

To further facilitate wide deployment, DefCOM does not require homogeneity of defense nodes. DefCOM nodes are not a design of a new product to be purchased and deployed, but instead require specific functionalities that are either present in today's security systems or easily added. Existing systems can thus be enlisted as part of DefCOM overlay. This means that current DDoS solutions can be mobilized to work together and achieve better performance through cooperative defense.

II. Related Work

Many research projects and commercial products attempt to tackle the DDoS problem. Only those that provide some form of cooperative defense between different nodes or share other strong similarities to DefCOM are reviewed here. See [2] for a more complete survey of all DDoS defense approaches.

Local Aggregate-Based Congestion Control (Local ACC)[3] provides an entirely self-contained solution at a single router for detection and rate-limiting of DDoS attacks and other traffic spikes (like the Slashdot Effect [4]). Routers respond to early signs of congestion in their queue by identifying high-bandwidth aggregates that are responsible for the majority of packet drops and imposing a rate limit on each aggregate. Pushback [5] extends local ACC with communication and coordination

[1] This is because the illustrated victim network is multi-homed. If it were single-homed, the single core deployment point would be able to see and police all the traffic to the victim

capabilities and is most closely related approach to DefCOM. If the congested router cannot control the aggregate itself, it issues a rate limit request for a fair share of the total rate limit to its immediate upstream neighbors who carry the aggregate's traffic. A router receiving the rate limit request decides whether to honor it, and whether to issue further requests upstream. The request propagation process in Pushback has the same goal as DefCOM's traffic tree discovery, but it can only be performed in the contiguous space of Pushback-enabled routers. A single legacy router on the attack tree blocks the request propagation, reducing Pushback to local ACC at this point. Pushback further inflicts significant damage on legitimate traffic sharing the attack path [5].

Secure Overlay Service (SOS) [6], [7] protects victims very well by granting them "cover" from DDoS attacks, hiding their location in a large peer-to-peer overlay network. SOS's chief limitation is that it is not suitable for a service available to the public, such as a Web server, since clients of an SOS-protected server must be aware of and cooperative with SOS. This was amended by adding Turing tests to SOS, in WebSOS [8], but this approach will only work for human users accessing the service. Further, SOS routes traffic on suboptimal route on the overlay, while DefCOM simply uses overlay for communication while traffic routing is performed as usual, over the path chosen by BGP. SOS does not address or limit the damage from legitimate nodes that get subverted by the attacker. Those nodes could still overwhelm the victim. DefCOM limits the amount of damage that a misbehaving end node can inflict on other users through its distributed rate limit algorithm.

Active Security System (ASSYST) [9] supports distributed response with non-contiguous deployment. All ASSYST nodes are essentially the equivalent of classifier nodes and are deployed only at edge networks. In [10] a collaborative DDoS defense system is proposed in which routers act as gateways, detecting DDoS attacks locally and identifying and dropping packets from misbehaving flows. Gateways are installed and communicate only within the source and the victim domains, thus providing cooperative defense of a limited scope. Similarly, COSSACK [11] forms a multicast group of defense nodes which are deployed at source and victim networks. Each defense node can autonomously detect the attack and issue an attack alert to the group. Sources involved in the attack cooperate with the victim to suppress it. Since intermediate networks do not participate in defense, systems described in [9], [10], [11] cannot control attack traffic from networks that do not deploy proposed defense.

III. DefCOM Overview

DefCOM builds a distributed peer-to-peer network of cooperative defense nodes scattered throughout the Internet. Defense nodes exchange information and control messages to detect attacks, and collectively respond to them while ensuring good service to legitimate traffic. DefCOM nodes can roughly be classified into three categories, based on the functionality they provide:

- *Alert generator* nodes that detect the attack and deliver an alarm to the rest of the peer network
- *Rate limiter* nodes that rate limit a high volume of traffic destined for the victim, but cannot profile traffic to separate legitimate from attack packets
- *Classifier* nodes that perform *selective rate-limiting*. They differentiate between legitimate and attack packets, dedicate their available bandwidth to legitimate traffic and cooperate with other defense nodes to ensure good service for the legitimate clients. Note that classifier functionality encompasses rate-limiter functionality. Also note that the traffic differentiation does not need to be perfect. As long as the classifier node respects its rate limit, it can choose to send any traffic it deems important for its users.

Each node can embody one or multiple functionalities, depending on its resources and the authorization within the peer network. However, the placement of some nodes (whether they are located in the core or edge network) facilitates some functionalities better than others. Edge-based defense nodes are well suited to deploy alert generator and classifier functionalities, while core defense nodes are well suited to provide rate-limiter functionality.

Since not every router or gateway in the Internet will be a defense node, DefCOM is designed to be effective in partial deployment. This feature is supported by an overlay network topology in which only nodes that have established direct peering relationship are aware of each other. The system provides a significant level of defense for potential targets with only a few defense nodes deployed, and becomes more effective as more defense nodes are added, protecting a larger community.

DefCOM's responsive actions take place only after the attack has been detected. Under normal operation the system is quiescent and does not police traffic. DefCOM's operation is best explained by first describing a simplistic version of the system, depicted in Figure 2.

Consider a DDoS attack on a popular Web server *V*, where the victim network *NetV* has a DefCOM defense node, providing alert generator functionality (*AG*), between itself and the rest of the Internet. An ISP providing services for *NetV* is hosting rate limiter

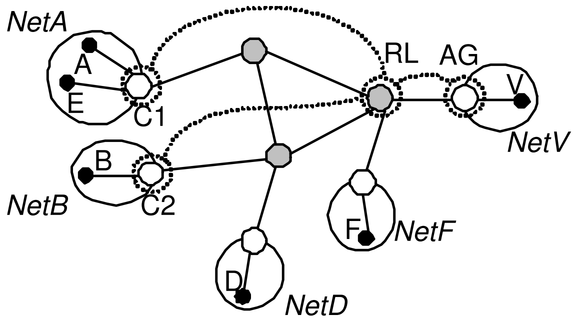


Fig. 2. Illustration of DefCOM operation

functionality (*RL*) at its core router. This router has abundant network resources, while the link leading to *NetV* is underprovisioned and can be overloaded by a DDoS attack. The victim communicates with legitimate clients, *A*, *B* and *D*, spread over three networks *NetA*, *NetB*, and *NetD*. *NetA* and *NetB* host classifier nodes *C1* and *C2*, while *NetD* does not have a classifier node. The alert generator, rate limiter and two classifier nodes (circled by dotted lines) form an overlay network. As mentioned before, nodes could contain more than one functionality but for simplicity this is not shown in the example.

After some time nodes *E* and *F* become subverted and start generating a DDoS attack on the victim that overloads its resources and degrades service to *A*, *B* and *D*. The victim's alert generator node is likely to detect this attack, and it sends the *attack alarm* message to the other nodes participating in the DefCOM overlay, informing them of the attack. In the next phase — *the traffic tree discovery* phase — DefCOM nodes cooperate to determine their relationships with the neighbors in the overlay. A node can be upstream, downstream or unrelated to its neighbor with regard to traffic flow to the victim. Nodes determine this by observing transit traffic that they relay to the victim and deploying *secure packet stamping*, which is described in section III-D. If some node *N* is upstream from its peer *P* (i.e. traffic is first flowing through *N* and then through *P* to reach the victim), then *P* will be *N*'s parent on traffic tree (and correspondingly *N* will be *P*'s child). In Figure 2 the traffic tree is depicted by dotted lines. Nodes *C1* and *C2* are children of node *RL*, which is a child of node *AG*, and *AG* is the root of the tree.

Once the traffic tree has been defined, a *distributed rate limit* algorithm controls the attack traffic. The desired rate limit is determined by the root node and propagated down the tree from parents to children. In this example, node *AG* will propagate rate limit requests to *RL*, who in turn will determine appropriate rate limits for its children and propagate rate limit requests to *C1* and *C2*. All rate limits are then enforced. *C1* and *C2*, being classifier nodes, will profile their traffic and dedicate their limited bandwidth to legitimate traffic from *A* and *B*. Classifier *C1* will drop most attack packets from *E*, while legitimate traffic from *B* will fall far under *C2*'s rate limit.

Using secure packet stamping, *C1* and *C2* stamp legitimate packets that they pass on to *RL*, thus informing *RL* that these packets should not be dropped. *RL* dedicates its limited bandwidth mostly to packets bearing *C1* and *C2*'s stamps (indicating that those packets have been vouched for by *C1* and *C2*), and drops with equal likelihood packets from *D* and *F*. The victim is relieved from a high volume of attack traffic by the joint rate-limiting action of *C1*, *C2* and *RL*. It continues to receive and serve requests from *A* and *B*, during the ongoing attack, as they are protected with stamps from *C1* and *C2*, and will safely reach the victim. Requests from *D* will compete for limited bandwidth with attack traffic from *F* and likely lose. This is unfortunate, but *D* can easily amend the situation by deploying a classifier node.

This example illustrates DefCOM's two major claims: (1) Attack traffic is controlled and the victim can resume its normal operation, and (2) Legitimate traffic from networks protected by classifier nodes continues to be served during the attack, while legitimate traffic from unprotected networks must compete with a smaller amount of attack traffic than it would be the case in the absence of DefCOM. Note that this has been achieved with deployment of only four defense nodes. Naturally, if more nodes are deployed, then the scalability and effectiveness of the system is improved, but even with sparse deployment DefCOM can provide significant benefits to its users. Further note that all three types of DefCOM nodes (alert generator, classifier, rate limiter) are necessary for complete protection against DDoS. Without an alert generator, small volume attacks could slip by core or source-end detection. Without classifier nodes, the attack would be suppressed but all legitimate clients would suffer collateral damage. Without rate limiter nodes, attack traffic from legacy networks could still reach the victim's network and overwhelm it. In our example, if *RL* were not deployed, traffic from malicious node *F* could still reach the victim and potentially overwhelm its defenses. Lastly, note that it is not necessary that *C1* and *C2* perfectly separate legitimate from the attack traffic — although it is to the best advantage of their customers if they do. As long as *C1* and *C2* obey their rate limit, they can send whatever traffic they deem important to the victim. To illustrate this, assume that *C1* mistakenly sends only attack traffic within its rate limit.

Since *C1* obeys the rate limit, this amount of attack will not be able to overwhelm the victim. The damage for *C1*'s malfunction is suffered mostly by its users who will likely take action to fix this. Since DefCOM enforces fair sharing of bandwidth, *C1* cannot claim more resources than its share, and thus cannot hurt legitimate traffic correctly detected and passed by *C2*.

A. DefCOM Overlay Network

DefCOM overlay network facilitates communication between nodes and is maintained at all times, regardless of the presence of attacks. When a new defense node decides to join DefCOM, it learns the locations (addresses) of several DefCOM nodes either by querying a public service (e.g., DNS) or from a published list. It then contacts one of the nodes in the overlay who either accepts it as a peer or redirects the join request to other DefCOM nodes. Once established, peering relationships may change over time; a node can acquire new peers and lose the old ones based on the flow of traffic and the node's interest.

DefCOM nodes construct secure, private, and authenticated communication channels between themselves and their direct peers in the overlay using standard PKI methods. Each DefCOM message is protected against replay and modification by adding a timestamp to the message and calculating a digest. This digest is signed by the originator's private key to guarantee authenticity. One way to establish a PKI infrastructure for DefCOM nodes is to have all nodes agree on several Certificate Authorities, and then distribute keys using certificates.

B. Detecting an Attack

Many security compromises are covert, and often occur without the knowledge of users of the compromised machine. In contrast, the DDoS victim can easily detect the occurrence of the attack by observing severe consumption of its resources. This simple attack detection method is implemented in current DefCOM prototype. On the other hand, it would be useful to have an early DDoS attack detection, *before* victim's resources have been severely consumed. There is significant body of research in this area that DefCOM can leverage as alert generators. Also note that severe resource consumption may occur as a result of perfectly legitimate activity — a *flash crowd* — when numerous legitimate users access a popular service simultaneously. Even though this is not a malicious activity, a necessary flood control response must unfortunately drop some clients' requests to favor of other clients rather than attempting and failing to serve all clients. DefCOM will achieve this effect by arbitrating the assignment of bandwidth to those legitimate users that are part of the overlay, making sacrifices from the non-overlay participants, and when necessary, even from a subset of the overlay participants.

C. Raising and Spreading an Alarm

Once an attack is detected, an alert generator will issue an *attack alarm* message, containing the IP address or IP address range of the target (victim) of the attack, and potentially more precise attack specification such as: (1) port or port range targeted by the attack, (2) attack type, for example: TCP SYN flood, (3) transport protocol specification, for example: IGMP, ICMP, UDP or TCP, (4) possible higher-level protocol information, such as RTSP, RTCP, HTTP. DefCOM nodes are quiescent until stimulated by an attack alarm, thus minimizing the danger of obstructing normal traffic flow in the absence of DDoS attacks.

Attack alarms propagate through the overlay network in a constrained and controlled flood. Nodes that observe traffic to the victim will become active upon the receipt of the alarm and communicate to build the traffic tree (explained in Section III-D).

A malicious outsider falsely reporting a DDoS attack could be a serious problem: if the false report is believed, and DefCOM limits traffic to the alleged victim, legitimate traffic could be dropped. To avoid this, DefCOM alert generators must posses an authorization to issue alerts for a given victim. A straightforward example of a possible solution would use a DefCOM certificate authority to issue certificates binding specific networks to alert generators that are allowed to issue alerts for them. The delegation of alert generators that are not residing in victim's network opens some interesting possibilities. For example, an ISP may have a remote monitoring facility located far from the victim, from which alert generator nodes probe the potential victim(s) they are protecting and determine if a poor response indicates a likely DDoS attack.

D. Building the Traffic Tree

The process of building and using the traffic tree is illustrated in Figure 3. The traffic tree for a specific attack contains only those DefCOM nodes that observe traffic to the victim and can thus help control it (nodes 1, 2, 4, 5, 6, 7, 8, and 9).

Defense nodes in the overlay cooperate to trace out the topology of the traffic tree and learn if they are upstream or downstream with regard to traffic destined for the victim and their peers on the tree. This cooperation is done through *secure message stamping*. Each active node

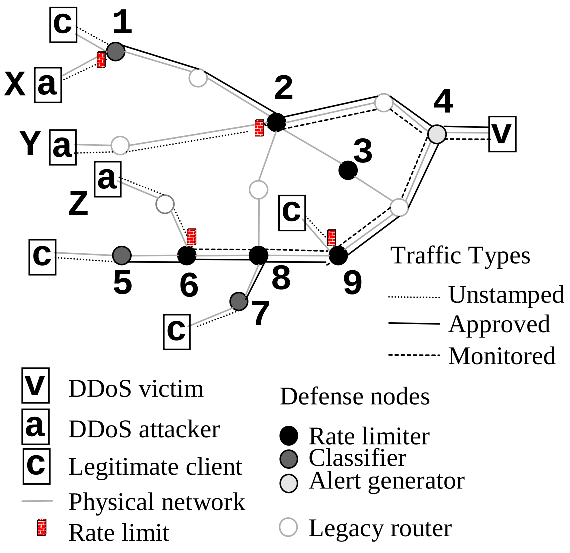


Fig. 3. Illustration of DefCOM traffic tree

picks two stamps for communication with each of its neighbors and securely delivers them to the neighbor. Stamps are short to facilitate packet marking by placing them in the packet header, and thus must be changed frequently to secure them against compromise. Further, stamps are only valid between two specific neighbors.

One possible field in IPv4 header that could carry DefCOM stamps is IP identification field. As discussed in [12] this field is used for assembly of fragmented packets, but those packets represent a very small portion of Internet's traffic. Thus overwriting this field should not interfere with normal traffic flows. Since DefCOM uses packet stamping only during attacks, this further minimizes likelihood of damage to legitimate traffic.

An active defense node places one of its stamps in the header of packets that it forwards to the victim. It also observes packets that it receives from its neighbors, looking for their stamps. A node becomes a parent of a neighbor whose stamped traffic it observes. A parent sends an explicit message to its children to inform them of their child status. In the Figure 3, node 4 is a root of the tree, with nodes 2 and 9 as its children. Node 9 has one child — 8, node 8 has children 6 and 7, and node 6 has one child — 5. Node 2 has node 1 as a child.

If a stamp gets compromised, the attacker would only be able to use it for a short period, before it gets changed by its owner. An attacker who is able to sniff traffic between two DefCOM nodes would be able to sniff each new stamp, as well, and inject bogus traffic with a correct stamp. This problem is discussed in the Section V. A discussion of the scalability of tracing traffic trees is covered in Section VI.

E. Controlling the Attack Flood

DefCOM controls the attack by propagating a rate limit request from the root of the traffic tree upstream towards the leaves. The original amount of the rate limit request is set by an alert generator. As the request propagates, this amount is split among nodes on the tree. Rate limit splitting is a distributed process. If a node on the tree has more than one child, it divides its bandwidth share among its children, generates corresponding rate limit requests and sends them to each child. The major concern is to guide the rate limiting process to assign a fair share of bandwidth to all legitimate users. This is challenging for several reasons: (1) the rate limit algorithm is distributed and each node has only the local knowledge, (2) a node may have non-uniform distribution of legitimate clients across node's children, and (3) traffic is dynamic so the rate limit must be adjusted accordingly.

Two basic designs for the distributed rate-limit algorithm are: (1) a proportional-share algorithm where the parent divides its bandwidth allocation amongst its children proportionally according to each child's reported need, and (2) an equal-share algorithm where the parent ignores the needs of each child and divides its allocation equally among all its children.

The proportional-share scheme is more fair in the sense that legitimate user's needs are more closely met. However, a subverted child can produce gigantic bogus requests for bandwidth that, if granted, result in tiny and incorrect allocations to its non-subverted siblings. The equal-share scheme is robust in face of subverted participants, but fails to properly handle the case when legitimate clients are not uniformly distributed among a node's children. DefCOM currently implements a loosely-enforced equal-share scheme (Explained in Section IV. In our future work we plan to investigate more sophisticated algorithms for distributed rate-limiting.

F. Traffic Classification and Separation

As mentioned in the Section III-D, each node has two stamps that it can use to mark traffic. One of those is a *legitimate* stamp and is used to mark traffic that has been vouched for by a classifier node. The other is a *monitored* stamp and is used to mark traffic that has not been vouched by a classifier node, but that has been policed according to the imposed rate limit at some defense node. In the Figure 3, classifier nodes 5, 7 and 1 would mark the traffic they deem legitimate or important with a legitimate stamp, and strive to pass as much of it to the victim as their rate limit permits. If there is some bandwidth left, classifier nodes will pass some of the traffic they cannot verify to be legitimate, and mark it as monitored.

A rate limiter node that has a classifier node as a child on the traffic tree, such as node 6 in the Figure 3, will receive three types of traffic: *legitimate* traffic marked by node 5 as important, *monitored* traffic also marked by node 5 as suspicious but already policed, and *unstamped* traffic from node Z. Node 6 then distributes its limited bandwidth to give as much of it as possible to the traffic carrying a legitimate stamp. The remaining bandwidth will first be offered to the monitored traffic, and any leftovers will be used to forward unstamped traffic. Node 6 also marks each packet it forwards with its own stamps, that it has exchanged with its parent 8. Legitimate and monitored stamps from node 5 will be simply overwritten by corresponding stamps from node 6. Unstamped traffic that gets forwarded will also be marked with monitored stamp by node 6. This indicates that the traffic has been policed and prevents double taxing in the upstream rate limiter nodes. The overall effect of packet stamping is the differentiation of three traffic classes and the service offered to those classes.

IV. Experimental Results

We implemented DefCOM in a Linux router and tested it with live traffic in Emulab testbed [13]. Linux router implementation consists of two parts: (1) the user-level implementation of DefCOM control message exchange and (2) the loadable kernel module implementation of traffic stamping and rate-limiting functionalities. Stamping and traffic tree discovery are fully implemented as described in DefCOM design. Alert generator functionality currently triggers an alert when the amount of the incoming traffic exceeds a predetermined limit. This alert contains only victim IP address so all traffic to the victim is subject to policing by DefCOM. Distributed rate-limit algorithm assigns an equal bandwidth share to all node's children. Unstamped traffic is strictly policed by rate limiter nodes, while packets bearing monitored and legitimate stamps are always allowed to pass. This enables DefCOM to successfully handle attack traffic but would lead to incorrect operation in face of malicious participants. We use D-WARD [1], [2] as our classifier node.

A. DDoS Attacks

To test DefCOM's response to DDoS attacks, we replicate experiments with Pushback as described in [3]. These experiments are extensive enough to demonstrate DefCOM effectiveness. At the same time they enable us to compare DefCOM performance with closely related Pushback approach.

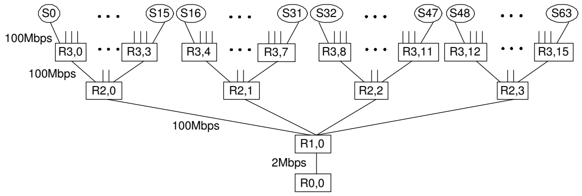


Fig. 4. Topology from Pushback experiment [3]

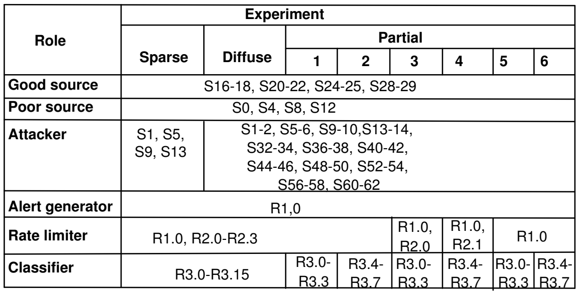
Figure 4 shows the experimental topology. There are four levels of routers — 0, 1, 2 and 3. The victim, router at level 0, is connected by a bottleneck link of bandwidth 2Mbps with a single router at level 1. This router connects to four routers at level 2, and each of these connects to four routers at level 3. Each router at level 3 also connects to four sources. In [3] links between sources and level 3 routers have 2 Mbps bandwidth, and links between routers at levels 3 and 2, and levels 2 and 1, have 20 Mbps bandwidth. This was difficult to replicate in Emulab testbed since any link with bandwidth lower than 100 Mbps requires an additional *delay* machine inserted on the link. This would approximately double the number of machines needed for the experiments. On the other hand, the experiments target only the bottleneck link and their results are not influenced by bandwidth of other links in the topology. In our experiments all links but the bottleneck link have 100 Mbps bandwidth as indicated in Figure 4.

Legitimate traffic occupies around 70% of the bottleneck link in the absence of the attacks. It consists of several consecutive telnet-like sessions between legitimate users and the victim, for the duration of 200 seconds. The attack starts 50 seconds after the start of legitimate traffic and lasts for 100 seconds.

There are four legitimate sources who share a third-level router with an attacker each – they are called *poor* sources in [3] and we borrow this terminology. There are also ten legitimate sources that do not share a third-level router with an attacker, called *good* sources [3]. Since DefCOM actions are not based on the link the traffic is coming from we fix the positions of poor and good sources and the attackers, unlike in [3] where positions were chosen at random. Poor sources' traffic occupies 25% of bottleneck bandwidth and good sources' traffic occupies 45%. Figure 5 provides details about each node's role in the experiments.

Sparse Attacks

In this experiment, DefCOM is fully deployed in the topology. First-level router `R1,0` is the alert generator and the rate-limiter, second-level routers `R2,0`—`R2,3` are rate-limiters and third-level routers `R3,0`—`R3,15` are classifiers running D-WARD. There are four attackers



Role	Experiment							
	Sparse	Diffuse	Partial					
			1	2	3	4	5	6
Good source	S16-18, S20-22, S24-25, S28-29							
Poor source	S0, S4, S8, S12							
Attacker	S1, S5, S9, S13	S1-2, S5-6, S9-10,S13-14, S32-34, S36-38, S40-42, S44-46, S48-50, S52-54, S56-58, S60-62						
Alert generator	R1,0							
Rate limiter	R1.0, R2.0-R2.3				R1.0, R2.0	R1.0, R2.1	R1.0	
Classifier	R3.0-R3.15		R3.0- R3.3	R3.4- R3.7	R3.0- R3.3	R3.4- R3.7	R3.0- R3.3	R3.4- R3.7

Fig. 5. Roles of nodes in experiments

in this scenario. The attackers each send 2 Mbps of UDP traffic to the victim R0,0.

Figure 6 shows the amount of bandwidth assigned to good, poor and attack traffic during the attack, in case of no defense, local ACC, pushback and DefCOM. Bold lines show baseline levels of good and poor traffic.

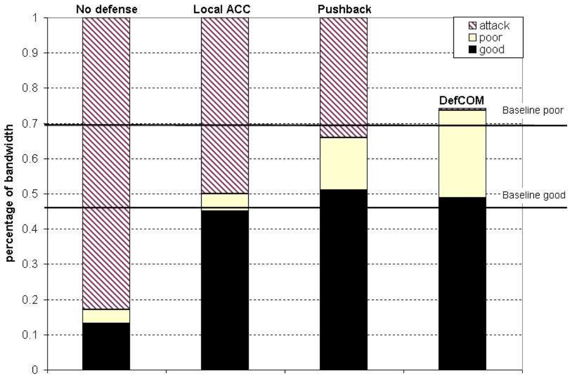


Fig. 6. Sparse DDoS experiment

Local ACC does pretty well in this case, letting through almost all of the good traffic. But it allows only a small amount of the poor traffic to get through. Pushback also allows the good traffic to get through, and some of the poor traffic. DefCOM successfully protects both traffic types. In fact, legitimate traffic receives a higher share of bandwidth than in the baseline case — this is because legitimate traffic loses some packets at the start of the attack and sends more aggressively to compensate for this loss, and DefCOM accommodates this excess bandwidth need. DefCOM further successfully suppresses the attack traffic letting through less than 1%, unlike Pushback that gives all the remaining bandwidth to the attack. We further note that "no defense" case shows larger damage to legitimate traffic than in [3]. The amount of damage inflicted on legitimate traffic by the attack depends on many factors that cannot be controlled in live experiments, such as aggressiveness of the legitimate traffic, client's and server's TCP implementation, path characteristics, etc.

Diffuse Attacks

In this experiment there are 32 attackers, who each send 0.25 Mbps UDP traffic to the victim. Figure 7 shows the amount of bandwidth assigned to good, poor and attack traffic during the attack, in case of no defense, local ACC, pushback and DefCOM. Bold lines show baseline levels of good and poor traffic.

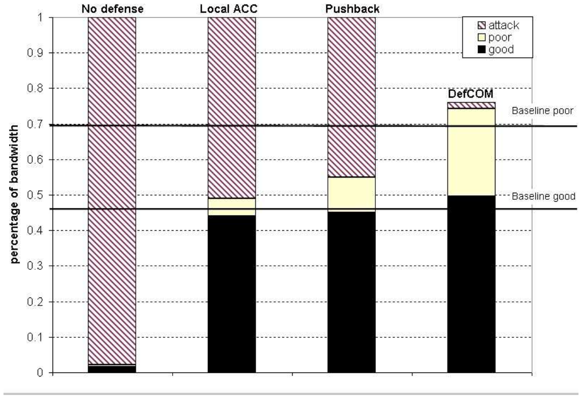


Fig. 7. Diffuse DDoS experiment

DefCOM exhibits same good performance as in sparse attack experiments. The only difference is that slightly larger amount of attack traffic — around 2% — reaches the victim in the case of a diffuse attack. Pushback still allows almost all the good traffic to get through, but the poor traffic receives a smaller bandwidth share than in the previous experiment. Again, "no defense" case shows much larger damage to legitimate traffic than in [3].

Partial Deployment

To test DefCOM performance under partial deployment, we investigate six partial deployment scenarios: (1) Classifier nodes are deployed only on third-level routers that connect to poor sources, all second-level nodes run rate-limiters, (2) Classifier nodes are deployed only on third-level routers that connect to good sources, all second-level nodes run rate-limiters, (3) DefCOM nodes are deployed only on a path from poor sources to the victim,(4) DefCOM nodes are deployed only on a path from good sources to the victim, (5) Classifiers at poor sources, rate-limiter and alert generator at R1.0, and (6) Classifiers at good sources, rate-limiter and alert generator at R1.0. Note that scenarios 5 and 6 illustrate non-contiguous deployment of DefCOM nodes.

The distribution of the attackers and legitimate clients, and the attack rate are the same as in the experiments with diffuse attacks. Figure 8 shows the amount of bandwidth assigned to good, poor and attack traffic during the attack, in case of partially deployed DefCOM. Bold lines show baseline levels of good and poor traffic.

We see that in all six cases DefCOM successfully pro-

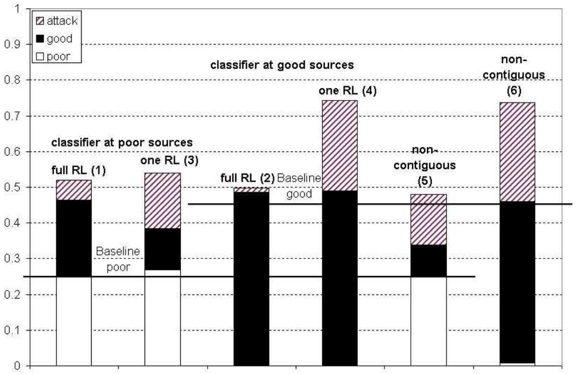


Fig. 8. Experiments with partial DefCOM deployment in case of a diffuse attack

tects traffic from networks that deploy classifier nodes. In scenarios 1 and 3 when DefCOM protects poor node's traffic, this traffic is delivered in full to the victim. In scenarios 2 and 4 when DefCOM protects good node's traffic, this traffic reaches the victim unharmed. Even in non-contiguous deployment scenarios 5 and 6 all protected traffic reaches the victim.

In cases where rate-limiters are fully deployed (scenarios 1 and 2) lower amount of attack traffic reaches the victim than in cases when rate-limiters are partially deployed (scenarios 3-6). This is because fully deployed rate-limiters police attack traffic at different aggregation points on the traffic tree and thus manage to control it well. In all scenarios, traffic from a network that does not deploy a classifier node suffers collateral damage since DefCOM cannot differentiate it from the attack traffic. This damage is smaller in case when good traffic is the one not protected by classifiers for two reasons: (1) good traffic does not share a third-level router with attackers so it competes with attack traffic at fewer spots, and (2) the amount of good traffic is almost twice the amount of poor traffic, which makes it more competitive when fighting for a limited resource.

B. Flash Crowd

To test DefCOM behavior in case of flash crowds we generated continuous FTP transfers of a 50 KB file from all 64 sources to the `R0,0` during 100 seconds. Without DefCOM, each transfer took on the average 5.0199 seconds and there were total of 1075 transfers. With fully-deployed DefCOM, this average was 5.0217 seconds and there were total of 1067 transfers. Maximum, minimum and median transfer times stayed the same. These experiments indicate that DefCOM does not interfere with normal network operation in case of flash crowds

V. DefCOM Security

DefCOM will likely be subject to various outsider and insider attacks attempting to bias or moderate its operation. Further, if the system offers new opportunities for attackers, the holes it opens could be worse than the holes it closes. This section discusses several potential attacks and offers possible countermeasures.

A. Attacks using Subverted DefCOM Nodes

Generally, the problem of having malicious participants exists in any distributed system (such as the existing unsecured routing and DNS infrastructure), and is yet unsolved in a general sense.

If DefCOM deploys proportional-share rate limit algorithm, malicious participants may attempt to monopolize limited bandwidth by marking all their traffic as legitimate. DefCOM nodes should devise monitoring and policing functions to ensure that rate limit requests are obeyed and resource requests are granted in accordance with negotiated policy. A node would assign a level of trust to each of its direct peers. Those peers that disobey rate limit requests would have their trust level reduced and would subsequently get lower share of the bandwidth.

A malicious node could further stamp all its malicious traffic as legitimate while still obeying the rate limit. This has the effect of denying service to the subtree rooted at malicious node but the limited amount of malicious traffic cannot do damage at the victim. Since a compromised router can already drop all its transit traffic, DefCOM does not create a new security problem.

A malicious parent could set a child's bandwidth limit to a very small quantity or zero, thus denying service to the subtree rooted at itself. Note that, since a parent is naturally downstream from all its children, this effect can be achieved without DefCOM by the malicious router dropping all transit traffic.

Traffic tree discovery is also subject to malicious participant attacks. One example of a potential malicious behavior could occur when a compromised node in the DefCOM network makes a false claim to be the parent of another node during the construction of a traffic tree, for a particular DDoS attack. As a result, legitimate defense nodes could be tricked into using an incorrect traffic tree. A potential defense against this attack would be to use a tool like traceroute to determine if an alleged parent is actually along the routing path to the victim, probably in cooperation with the overlay network topology construction mechanisms.

A malicious defense node could launch a DoS attack against another defense node. The attack would attempt to occupy the target with control traffic such as bogus encrypted messages or repeated peering attempts. Note here that control messages cannot be faked as they are cryptographically signed by the node's peers. However, they could consume a node's resources for processing bogus messages. Since a node only communicates with its peers, a possible solution to this problem is to limit the acceptance rate of control messages and to reject communication with nodes that exceed the rate. Client puzzles [14] should make a repeated peering attempt attack (performed by sending a flood of requests to join an overlay) more expensive for the attacker than the victim.

B. Interfering with a Classifier Node

In the general case, an attacker could attempt to interfere with a classifier node, fooling it into classifying the attack traffic as legitimate. This action will only harm legitimate users of the network deploying the classifier node. Since this node is owned by the same network its malfunction is likely to be promptly discovered and corrected.

Specifically, if DefCOM deploys D-WARD [2], [1] as classifier node, attacker could attempt to mislead D-WARD to classify attack traffic as legitimate. D-WARD classifies TCP connections as legitimate or attack based on the correlation of incoming and outgoing traffic. Aggressive TCP connections that send high traffic volume and receive few or no replies will be classified as potentially malicious, while well-behaved TCP connections that receive sufficient numbers of replies to their traffic will be classified as legitimate. An attacker could spoof large number of replies to his malicious traffic to force D-WARD to classify attack connections as legitimate and offer priority service to them. This would inflict damage to real legitimate traffic circulating through DefCOM overlay, which may receive a lower service level, as its bandwidth is stolen by attack traffic. D-WARD can detect this by sampling a few connections and delaying their outgoing packets for a second or two. If the system receives replies to packets that have not been sent, the sampled connections will be declared malicious.

C. Probing for Holes in the Defense

This is a next-generation attack we would expect to see after the wide-spread deployment and adoption of DefCOM. The attack occurs when a large set of zombies cooperate to find which zombies have a route to the victim that does not pass through a classifier or rate-limiter DefCOM node. Once this "hole" in the defensive mesh is determined, the attacker would attempt to aggressively recruit new zombies from these networks. To solve this problem, the victim should ensure that a rate limiter node is placed between its network and the rest of the Internet. The easiest way to achieve this is to purchase rate limiting service from the same ISP that provides the connectivity services. If rate-limiter functionality were deployed systematically, the optimal deployment would likely be on the points of vertex cover of core topology. Arguments that support this are similar to arguments presented in [15], for route-based filtering deployment. Briefly, highly connected core routers can police a large amount of traffic. Park et al. [15] prove that deployment of a traffic policing service at 18% of core AS domains would have a major impact on attack traffic. Thus, with a moderate core deployment it would be hard for attackers to find holes in the defense. We plan investigate this problem further in our research.

D. New Vulnerabilities

DefCOM requires that its participants send messages and take actions on others' requests. Particularly in core nodes, DefCOM essentially introduces new types of router behaviors that can be controlled remotely. This may open new vulnerabilities (e.g., new buffer overflows) if added protocols and services are not properly secured.

Such dangers can be minimized by careful design and implementation, and by proper use of cryptography to ensure that only trusted parties can access the system's new functionalities. Nevertheless, we must exercise extraordinary caution when adding new functionality to routers as part of DefCOM, and must perform careful testing and validation to give strong assurances that DefCOM does not add new flaws.

VI. Scalability

DefCOM nodes communicate only with direct neighbors in the overlay network. This feature promises good scalability if no node has a large number of peers. To control this effect, we need to control the overlay building algorithm, preferring those topologies in which each node has a balanced, small number of peers. This may not always be possible as the overlay topology depends also on the underlying physical topology and pattern of defense nodes deployment (i.e. in the case when only edge networks participate in the overlay a node may have a multitude of peers). In general, larger deployment of rate limiter nodes in the core lowers degree of the node (number of potential peers) and improves scalability. Should DefCOM be widely deployed, it would be necessary to provide a sufficient number of rate limiter nodes in the core to achieve satisfactory scalability. A node stores only a small amount of state information per peer — some traffic statistics data, peer stamps and a public key. The amount of storage space consumed depends again on the overlay topology.

The other factor that affects scalability is the number of attack reports, as a specific traffic tree is built for each report. In our future work, we will investigate strategies to combine traffic trees in cases when multiple attack reports coincide. This may be a case when worm propagation creates a wide-spread DoS effect on the Internet.

VII. DefCOM Deployment

Existing security systems can be augmented to provide required alert generator, rate limiter or classifier functionality. In this section we list the requirements that defense nodes must meet in order to claim one of these functionalities and we provide examples of existing defense systems that meet these requirements.

A. Alert Generator

Alert generator nodes must be able to determine when an attack is occurring. It is also desirable if they can generate at least a crude attack profile (e.g., identifying which protocol is used in the attack) to reduce collateral damage. Many networks already deploy security systems that provide alert generator functionality, such as firewalls, intrusion detection and monitoring systems. Those systems are prompt to detect the attack and frequently can characterize malicious traffic, at some level. A potential target network can complement functionality of its current security systems by enlisting its defenses as alert generator nodes in the overlay network. This membership does not preclude the network from making its own response to the attack, but it does offer better flood control and superior traffic profiling. A typical security system needs to be augmented to communicate with DefCOM, thus enabling the system to deliver authenticated alarms to the overlay network.

B. Rate Limiter

Nodes providing rate limiter functionality must be able to handle large traffic loads. They must have sufficient network resources that cannot be easily overwhelmed with high-volume traffic. Further, they must be able to apply selective rate-limits on traffic matching a given destination IP address, any stamps, if present, and potentially a protocol field. As rate limiter nodes do not incur high per-packet overhead, they need not possess many computational resources.

Current core routers already handle high traffic loads during regular operation. They also have the ability to install selective rate limits based on IP addresses and protocol fields, and can deploy those limits in response to external SNMP commands. To join DefCOM, core routers would have to be augmented with (1) a secure communication layer to enable them to exchange messages and authenticate information received from other DefCOM nodes, (2) ability to examine packet stamps, and (3) support for secure packet stamping that essentially involves overwriting parts of a packet's IP header. Assuming that the majority of traffic is not destined for a known attack victim, all of these functionalities could be implemented on a co-processor tasked to handle packets matching an attack signature, provided the co-processor could instruct the router about installing and removing rate-limits.

C. Classifier

Classifier defense nodes are likely to perform much more computation per packet than rate limiter nodes. Therefore, they are best located at network points that relay low traffic volumes. One such place would be a border router between an edge network and the core. Classifier nodes are crucial to fulfill the DefCOM guarantee of a good service level to legitimate traffic. They should be fairly successful and accurate in separating legitimate from attack traffic, so that legitimate traffic may be granted priority in resource sharing.

We are aware of two source-end DDoS defense systems that meet the above requirements: MANAnet Reverse Firewall [16] and D-WARD [1], [2]. MANAnet Reverse Firewall [16] is a commercial product that prevents DDoS attacks by limiting the rate of "unexpected" outgoing packets at a network's exit router. The evaluation of packets as expected and unexpected is performed only for outgoing TCP packets, and is based on information received in TCP acknowledgements from the foreign peer. The outgoing rate of other traffic types (UDP, ICMP) is limited to a fixed value. Reverse Firewall already provides traffic separation through expected/unexpected classification. In order to join DefCOM it would have to be augmented with (1) a secure communication layer, (2) a module that receives external attack alert signal, authenticates it and triggers response action, (3) a module that receives rate limit requests and deploys them in Reverse Firewall, and a (4) packet stamping capability.

D-WARD [1], [2] is a source-end DDoS defense system that prevents outgoing attacks from a deploying network while preserving good service guarantees to legitimate traffic. D-WARD can detect DDoS attacks either autonomously or by receiving an attack alert signal from another defense system. Once an attack has been detected, D-WARD installs a selective rate limit on the total outgoing flow to the victim, preferentially passing those packets that are deemed legitimate. D-WARD separates legitimate from attack packets by collecting statistics on each connection it observes (connection is defined as all traffic exchanged between an IP address and port on the deploying network side and an IP address and port in the foreign network) and classifying connections based on the predefined legitimate traffic models. D-WARD has been extensively tested both in large-scale experiments in Emulab [13], cooperative tests with other research groups and in real deployment. Test results are presented in [2] and they strongly support the claim that D-WARD can precisely separate legitimate from the attack traffic. D-WARD has very low level of false positives (less than 0.1%) for autonomous attack detection, and generates a few to none legitimate packet drops during attacks. To convert D-WARD to a classifier node we had to add secure packet stamping and DefCOM message exchange modules to current D-WARD prototype.

VIII. Future Work and Conclusions

The main area of future work on DefCOM is the investigation of security techniques that facilitate detection of malicious participants and limit the damage to DefCOM operation. We also plan to investigate dynamic overlay topology construction algorithms. DefCOM currently assigns an equal share of bandwidth to all children, which is suboptimal in case of unequal distribution of traffic among children. We plan to investigate approaches that would enable equal-share algorithm to converge to fair-share as traffic changes. As mentioned in section III-E such approaches will have to be secured against misuse by malicious participants.

DDoS is a complex problem involving hosts distributed all over the Internet and affecting numerous networks. While localized defenses alleviate damage from small-scale, easily characterizable attacks, more sophisticated threats can only be handled through a cooperative, Internet-wide defense. We have proposed a distributed framework called DefCOM that existing defense nodes can join to achieve a cooperative response to DDoS attacks. DefCOM enables participants to perform those defense actions they are most capable of, supplementing their weaknesses with strengths of others. Each participant directly benefits from DefCOM operation. Victim networks receive protection from the attack, while source networks receive a guarantee that their traffic will be delivered during attacks. Core networks deploying rate-limiter functionality can sell DDoS protection services to their customers. This economic model along with the ability to accommodate existing defenses should motivate wide deployment. Experiments with DefCOM prototype show promising results and warrant further research into cooperative DDoS defense.

References

[1] J. Mirkovic, G. Prier, and P. Reiher, “Attacking DDoS at the Source,” in *Proceedings of the ICNP 2002*, November 2002.

[2] Jelena Mirkovic, *D-WARD: Source-End Defense Against Distributed Denial-of-Service Attacks*, Ph.D. thesis, University of California Los Angeles, 2003.

[3] R. Mahajan, S. Bellovin, S. Floyd, V. Paxson, and S. Shenker, “Controlling high bandwidth aggregates in the network,” *ACM Computer Communications Review*, vol. 32, no. 3, July 2002.

[4] S Adler, *The Slashdot Effect: An Analysis of Three Internet Publications*.

[5] J. Ioannidis and S. M. Bellovin, “Pushback: Router-Based Defense Against DDoS Attacks,” in *Proceedings of NDSS*, February 2002.

[6] A. D. Keromytis, V. Misra, and D. Rubenstein, “SOS: Secure Overlay Services,” in *Proceedings of SIGCOMM 2002*, 2002.

[7] A. D. Keromytis, V. Misra, and D. Rubenstein, “Using Overlays to Improve Network Security,” in *Proceedings of SPIE ITCom Conference on Scalability and Traffic Control in IP Networks II*, July 2002.

[8] Debra L. Cook, William G. Morein, Angelos D. Keromytis, Vishal Misra, and Daniel Rubenstein., “Websos: Protecting web servers from ddos attacks,” in *In the Proceedings of the 11th IEEE International Conference on Networks (ICON).*, 2003.

[9] R Canonico, D Cotroneo, L Peluso, S P Romano, and G Ventre, “Programming routers to improve network security,” in *Proceedings of the OPENSIG 2001 Workshop Next Generation Network Programming*, September 2001.

[10] D Xuan, R Bettati, and W Zhao, “A gateway-based defense system for distributed dos attacks in high-speed networks,” in *Proceedings of 2001 IEEE Workshop on Information Assurance and Security*, June 2001.

[11] C Papadopoulos, R Lindell, J Mehringer, A Hussain, and R Govindan, “Cossack: Coordinated suppression of simultaneous attacks,” in *Proceedings of DISCEX III*, April 2003, to appear.

[12] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, “Practical Network Support for IP Traceback,” in *Proceedings of ACM SIGCOMM 2000*, August 2000.

[13] Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar, “An integrated experimental environment for distributed systems and networks,” in *In the Proceedings of the OSDI’02*, Boston, MA, Dec. 2002, USENIX Association, pp. 255–270.

[14] A. Juels and J. Brainard, “Client puzzles: A cryptographic countermeasure against connection depletion attacks,” in *Proceedings of the 1999 Networks and distributed system security symposium*, March 1999.

[15] K. Park and H. Lee, “On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets,” in *Proceedings of ACM SIGCOMM 2001*, August 2001.

[16] Cs3. Inc, *MANAnet DDoS White Papers*, http://www.cs3-inc.com/mananet.html.