Journal of Network and Systems Management, Vol. 12, No. 1, March 2004 (© 2004)

Distributed Management Architecture for Cooperative Detection and Reaction to DDoS Attacks

G. Koutepas,[1,2] F. Stamatelopoulos,[1] and B. Maglaris[1]

We propose a cooperative intrusion detection framework focused on countering Distributed Denial-of-Service (DDoS) attacks through the introduction of a distributed overlay early-warning network. Our goal is to minimize the detection and reaction time and automate responses, while involving as many networks as possible along the attack path. The proposed approach relies on building a "community" of trusted partners that will cooperate by exchanging security information so that inclusion in the attack path is detected locally and without traceback procedures. The main building block is the Cooperative anti-DDoS Entity, a modular software system deployed in each participating network domain that supports secure message exchanges and local responses tailored to individual sites' policies. We discuss the operation and the implementation of a prototype, and we provide a survey of the methodologies against DDoS and compare our approach to related work.

KEY WORDS: Network security; distributed management; inter-domain; multicast; automated reaction.

1. INTRODUCTION

Distributed Denial-of-Service (DDoS) attacks have evolved in complexity and sophistication and pose a serious threat to the modern IT infrastructure. Their distinguishing characteristic is that they do not attempt to break into the target computer systems, like intrusive attacks, but rather aim at the disruption of normal operations through flooding of network links. The sources of the attacks are usually trojaned computer systems, which when instructed will initiate the flow of malicious traffic. Although small in scale and difficult to detect near the sources, the flows have a cumulative devastating effect when they reach their target.

[1]Network Management and Optimal Design Laboratory, Electrical and Computer Engineering Department, National Technical University of Athens, Zografou, GR 157 80, Athens, Greece. E-mail: {gkoutep, fotis, maglaris}@netmode.ntua.gr
[2]To whom correspondence should be addressed.

1064-7570/04/0300-0073/0 © 2004 Plenum Publishing Corporation

Management-wise, DDoS attacks present an interesting challenge since their nature makes them difficult to stop by the efforts of a single site. Factors that contribute to this are: (a) attackers most of the time spoof packet source IPs address; (b) the possibility of the attack initiating from a wide range of networks worldwide; and (c) the inability of a domain to enforce incoming traffic shaping; detected malicious flows can be blocked locally but the assistance of the upstream network is still needed in order to free the resources occupied on the incoming link. Adding to these problems, single Intrusion Detection Systems have the task to distinguish between malicious packet flooding and legitimate traffic surges.

Consequently, any effective response procedure requires cooperation between sites. When attempting to counter a DDoS, the specific attack characteristics have to be determined locally and communicated to networks on the attack path (possibly through attack-congested lines) in order to take appropriate measures. Currently, this is a manual, nonautomatic and time consuming procedure. It heavily depends on the administrator's availability and good will, as well as the service policies at the upstream networks. According to the site's security policies the typical reactions implemented usually consist of setting up tailor-made blocking or throttling filters on active network components. Still, no matter how effective this response will be, the bandwidth penalty is present on all the domains along the attack path. To alleviate the resulting congestion extra steps must be taken and contacts must be made between these networks. The further we move from the victim, the more dispersed this procedure becomes and there is less immediate interest from the domains to help.

In summary, the requirements for an effective response to a DDoS attack are: (a) early detection at the victim site and at as many upstream domains as possible; (b) the ability to exchange incident information between domains without inflicting significant additional network load [1]; and (c) timely and effective response in as many domains on the attack path as possible.

Our work aims to satisfy these requirements by introducing a framework for countering DDoS attacks through the cooperation of network domains. The proposed architecture is built around the concept of the *Cooperative Counter-DDoS Entity*. This is a modular software platform to be deployed within each participating domain, and it offers communication and response coordination services. Our approach allows different administrative realms to cooperate during a DDoS attack with a group of trusted partners, without releasing any authority within their own domain or compromising their security. Through this cooperation within the trusted community, each Entity will try to detect its position on the attack path (*possible source*, *transient node*, *target*, *not on path*) without performing traceback procedures, and then react accordingly and in parallel to the other Entities. The Entity is viewed as the gateway to a trusted overlay network, where all members share security notices [Intrusion Detection Systems (IDS) alerts] and automated assistance (active filtering) against DDoS attacks. The Entity behavior within this

trusted overlay network is guided by the local policy (simple rule-based system) that defines what information to share and how to react to shared security information.

The organization of this paper is as follows. In Section 2 we present recent methodologies and research on countering DDoS attacks and put our approach in perspective. In Section 3 we introduce the proposed framework and outline the cooperative operation. Section 4 discusses the Entity's software architecture, its internal operation and how information coming from different sources is processed to produce understanding of the attack's path and reaction results. In Section 5 we compare our methodology to other counter-DDoS proposals. Finally, conclusions and future work directions are presented in Section 6.

2. ANALYSIS AND SURVEY OF DDoS SOLUTIONS

DDoS attacks are multifaceted problems and present challenges in every phase of the effort to counter them. We attempt here to give an overview of the main DDoS solution methodologies, presently proposed, in order to demonstrate the positioning of our proposal, the "Cooperative Anti-DDoS Entity" in relation to them. The various solutions can be separated in the following three broad categories: *detection*, *traceback* (source and path discovery), and *reaction*. The most comprehensive analysis of DDoS attack characteristics is provided by Moore *et al.* [2] with some additional information on actual DDoS events by Sung *et al.* [3]. A systematic review and evaluation of existing DDoS defense mechanisms is presented by Chang [4] while a simple taxonomy of DDoS attacks and the types of defence mechanisms is also presented by Mirkoiv *et al.* [5].

2.1. Detection

Figure 1 presents a layout of the main detection methods employed for discovering DDoS attacks. The simplest detection process can take place at the victim's network with the straightforward observation of line congestion especially in small bandwidth links. However, additional analysis effort is required in order to distinguish DDoS attacks from normal traffic surges. The method of monitoring router load [6] or analyzing the discarded packets [7] refine first observations and will initiate the process of acquiring the attack's characteristics.

Current traffic analysis methodologies are based on traffic sampling [8] or on the usage of specialized router traffic monitoring facilities (e.g., Cisco's Netflow [9]) that perform off-line analysis on a separate machine; both methodologies rely on anomaly detection. A tool called Panoptis [10] attempts to perform traffic analysis at the border routers of big sites using traffic traces provided by Netflow and produces simple reports to the management console. Network Intrusion Detection Systems (NIDS) perform a similar task within local networks.

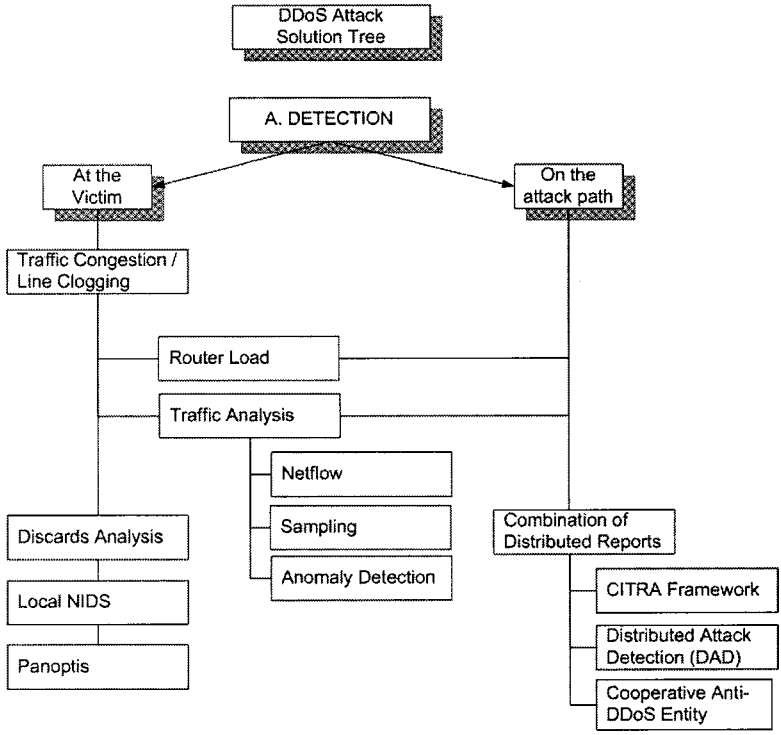

Fig. 1. Detection methodologies for DDoS attacks.

Router load monitoring, as well as traffic analysis in general, can be employed at any stage along the attack path. However, the methods just described don't expand further than a single site; their centralized nature prevents distributed operation, unless supported by an infrastructure that will assume the tasks of communications and report fusion and analysis. Solutions like the Distributed Attack Detection (DAD) system [11], and the Cooperative Intrusion Traceback and Response (CITRA) framework [12,13], as well as our own system, offer such distributed detection frameworks as part of a general attack countering infrastructure. Being distributed frameworks, DAD and CITRA are compared to our approach and presented in more detail in Section 5.

2.2. Traceback

The vast majority of attacks cover their traces by using spoofed IP source addresses on their packets, which makes identification of the real source and the attack route (traceback) extremely difficult. Also, as deduced from Moore *et al.*

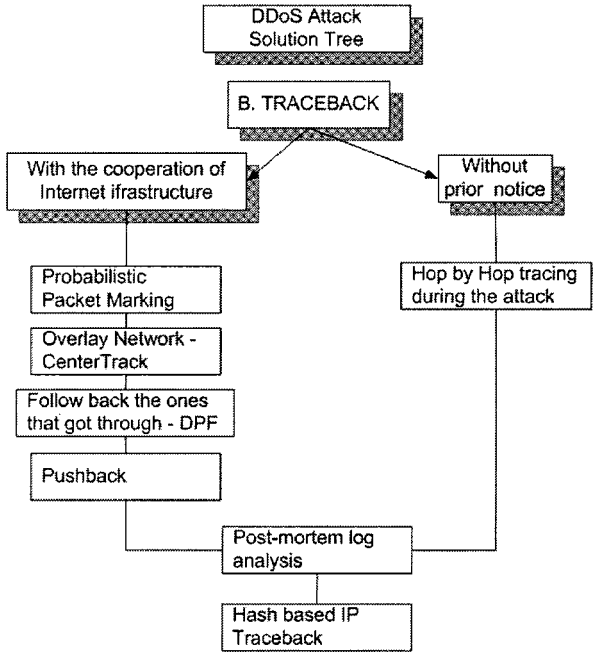

Fig. 2. Solutions proposed for the traceback problem.

[2], usually a DDoS attack finishes in a very short period of time making manual tracing useless. Figure 2 outlines current traceback proposals. Post-mortem traffic log analysis yields better results; although it does not affect the ongoing attack, it can identify machines infected by DDoS agents and vulnerable parts of the network. A post-mortem approach of Snoeren *et al.* [14] uses a hash method for generating traffic audit trails within a network that can trace the origin of any recent IP packet.

Automatic traceback methodologies may require a tracing infrastructure prepared before the attack. The Pushback method [15] sequentially notifies the routers on the attack path in the reverse order of the attack propagation, but it requires special router software. In Savage *et al.* [16], packet marking (probabilistic or deterministic) at various points on the Internet infrastructure is suggested and analyzed. The victim should be able to deduce the true path taken by incoming traffic by examining packet marking. A different approach of packet marking based on algebraic coding of paths traversed improves noise elimination and multiple path reconstruction [17]. A number of other such approaches trace the offending packets through an overlay networking infrastructure. In the CenterTrack [18] methodology, implemented within ISP networks, traffic to the victim of a DDoS

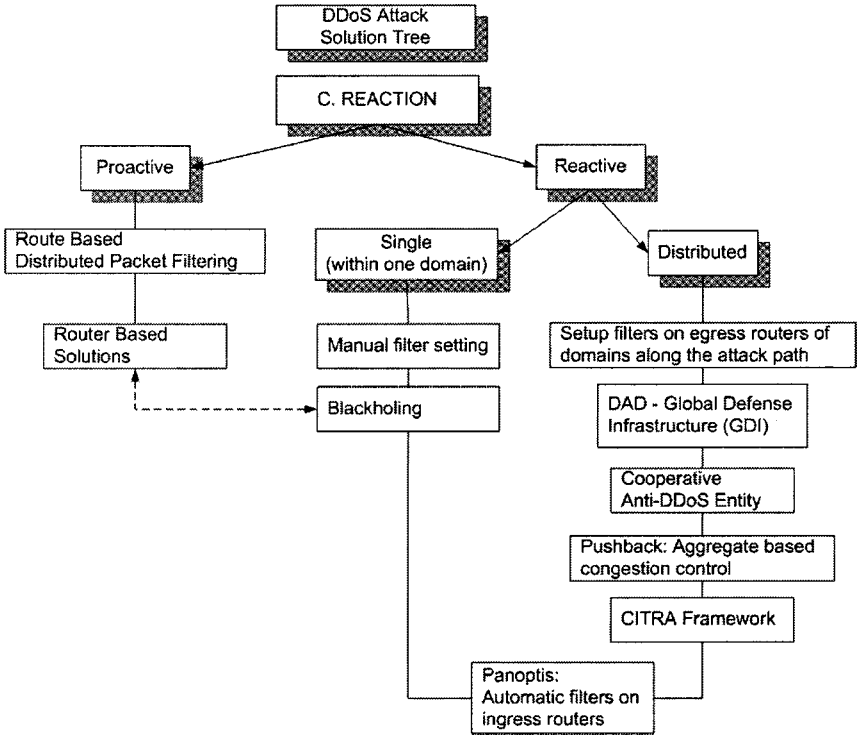

Fig. 3. Main reaction methods to DDoS attacks.

is tunneled through alternative routes in the network until detecting the ingress routers used by the attackers. In Route Based Distributed Packet Filtering (DPF) [19] (a reaction/filtering solution) it is demonstrated that if enough Internet core routers employ the proposed filtering method it would be fairly straightforward to discover the sources of the attacks that do manage to get through the protection infrastructure.

2.3. Reaction

As presented in Fig. 3, the reaction solutions can be categorized as proactive and reactive. A very typical proactive approach is to block traffic with source addresses that do not appear correct according to the routing policies in force. The Route Based DPF [19] suggests doing this on the whole Internet, thus configuring an "Internet-firewall," and proves that the anti-DDoS result will be more effective with the greater number of core routers participating. The method has a number of weaknesses: the need for efficient implementation and the fact that the route

to the source of a packet is not always known. Route Based Filtering can also be implemented in every network as part of the security policy. The measure's effectiveness diminishes with the positioning of the domain closer to the Internet periphery.

The reactive efforts can be based on single or multiple sites. Single domains positioned upstream to the victim (e.g., ISPs) can limit attack effects by setting up tailor-made blocking or throttling filters on egress links close to the victim. Another option, when the attack is targeting only a specific machine, is for the ISP to "blackhole" all the traffic destined for that node, thus allowing room for the rest of the traffic on the victim's link. The Panoptis system [10] automatically sets up the necessary filters on domain ingress routers based on its traffic analysis findings mentioned earlier.

Our proposed solution falls in the category of distributed reaction. Other similar approaches that will be described in Section 5, include the DAD System [11], the CITRA framework [12,13], and the multi-domain mechanism of the Pushback mechanism [15].

3. ARCHITECTURE AND OPERATION

3.1. Building a Network of Cooperating Domains

The proposed approach relies on building a "community" of trusted partners that will cooperate by exchanging security information so that inclusion in the attack path is detected locally and without traceback procedures. A *Cooperative Counter-DDoS Entity* is assigned to each participating network domain. This is a modular software platform that implements automatic exchange of security messages and response coordination services according to the proposed framework.

3.1.1. Community Organization

Enrolling domains in an agreement for cooperation and the propagation of response actions mimics Computer Emergency Response Teams' (CERT) style of operation: each CERT has an area of responsibility ("constituency") and cooperates with other such teams, considered trusted partners and usually passing through a certification procedure [20]. Similarly, in our approach, the community of the cooperating domains is set up off-line and all managerial details (encryption keys, notification frequency, etc.) are arranged before the proposed architecture is deployed. Many domains are already discussing security matters between them and make efforts to cooperate. Building a cooperating community consisting of a limited number of domains does not require global consensus and Internet steering bodies' approval like in the case of Internet-Firewalls [10]. The established credibility of CERT teams can play an auxiliary role in the arrangements as a trusted party for key exchanges, dispute resolving etc.

The effectiveness of the proposed methodology is very much dependent on the extent of community participation. Obviously, the distributed nature of the solution requires at least two participants. Even a small community of a few participating domains along the attack path will improve detection and reaction; broader participation is expected to increase the framework's accuracy and resulting reaction speed. The key benefit of our proposal is based on broad participation, and thus fast correlation of alerts and detection data, as well as the coordination of distributed reaction.

3.1.2. Communications within the Framework

One of the main problems encountered in every cooperative scheme is how to prevent communications between nodes from escalating and causing extra traffic in a network already overloaded by an attack. In our approach we have chosen Multicast as the transport method because it solves this drawback and offers some extra organizational advantages as well. The Entities depend on Multicast connectivity existing between the sites. At each domain there is a limited and well specified number of subscribers to the service, so Multicast Tree maintenance (inactive branch pruning etc.) is de-facto simpler than the standard on-off client registration. Domains that are not participating in the Framework and not supporting this type of communications can be bypassed by tunnels.

The multicast inter-domain connectivity of the Cooperative Framework can be built using any of the currently available methods [21] according to what is more effective and efficient for the partner networks. One approach can be to set up permanent tunnels between Multicast routers (the MBONE method). A number of currently available dynamic connection protocols are also appropriate: PIM-Dense Mode, Multi-protocol extensions to BGP4 (MBGP), MBGP/Multicast Source Discovery Protocol (MSDP)/PIM Sparse Mode.

Having a Multicast communication base makes it possible to organize and scale the architecture by the use of groups and variable TTL values. Messages exchanged can be distributed to neighboring nodes within a certain radius using different TTL values. The domains participating in the anti-DDoS Framework agree preliminary on the communication specifics and multicast connectivity through their network (or arranging tunnels through third nonparticipating and nonmulticast supporting domains).

As explained later in paragraph 3.1.3, an important security concern is protecting the Entity from direct attacks. This can be addressed by exploiting multicast in order to maintain a "stealthy" presence for the Entities. After registering with the local multicast router they can passively listen to notifications and transmit "to the group" without revealing their real address. Likewise, more than one system, even on different subnets of the domain, can work in parallel, monitor the same multicast groups, with all of them maintaining the same state and prepared for fail-over procedures. Dispersing the Entities reduces the possibility of a single attack impairing all of them simultaneously.

The messages exchanged by the Entities are composed according to the, XML based, Intrusion Detection Message Exchange Format (IDMEF). The IETF is developing this protocol, currently in draft state, to enable cooperation between Intrusion Detection Systems [22]. Following the IDMEF paradigm, messages exchanged between Entities are divided in two categories: Heartbeats and Alerts. Heartbeats are periodic notifications that a node is in operation, with connectivity to the Framework. Alerts are sent by Entities when they detect and identify security events.

3.1.3. Security Considerations

An attacker orchestrating a DDoS attack could "tune into" the right Multicast group (the one used by the Entities) and listen for signs of detection and response communications. Knowing such information could make it possible to direct the hostile machines to new attack patterns so that the malicious traffic can elude any newly installed filters. Another concern is that fake alert messages describing nonexistent events could initiate Entity responses to hinder legitimate traffic. These security concerns indicate that Entity communications need to be secured. Communication security in our system uses symmetric (single key) encryption (in the prototype we used DES), message signing and time-stamping. The Cooperative Framework requires exchanges between limited numbers of participants so key management is limited to periodical off-line key exchanges and precautions in their usage. To avoid message spoofing or duplication, each entrusted Entity also digitally signs its messages (using PGP and public-private key pairs for each domain, again with off-line exchanges) and includes a time stamp in the body of the message (the Entities are time-synchronized.)

The Entities support the operation of the whole cooperative scheme, so they can be considered critical points of failure. We have tried to address this concern by making the design lightweight, modular and portable. In the current solution the Entity is easily transferable to another machine in the event of a malicious or accidental failure. As mentioned earlier, it is also possible for a secondary Entity to operate in parallel with automatic fail-over when the main one stops performing. This weakness, however, benefits the detection procedure. Failure to transmit "Heartbeat" messages at regular time intervals indicates a possible security event. Heartbeats not received are one of the parameters considered in the assessing algorithm of the Entity inference engine.

Finally we have to consider the cases when the Multicast backbone itself may fail, either at the leaf level because of DDoS traffic congestion or at the core level due to direct attacks against the routers maintaining the multicast tree. This could certainly affect the connectivity of the overlay network. Although this is a real weakness of the networking infrastructure it presumes existing router vulnerabilities and well informed attackers that will specifically target them. However, such

attacks are identified by the proposed framework through periodical "heartbeat" messages (disruption of their steady flow), so that the algorithm is aware that a possible attack is in progress. Also, this weakness can be addressed more effectively through the use of level 2, out-of-band multicast connectivity (e.g., via MPLS tunnels).

3.2. Deployment of the Framework

3.2.1. Entity Operation

The Entity is designed to offer communication and response functionality, for the Framework of cooperative domains. At each domain it is the receiving point for messages about the security of the community members and the operational status of neighboring Entities. It also transmits info about local security events and assessments. Locally the Entity acts as the highest-level node of the operating IDS hierarchy. Using info from peer nodes and the local Intrusion Detection (ID) Systems the Entity evaluates the security status and infers about the presence of an on-going attack.

The second important function of the Entity is to offer limited response capability. This departs from the usual IDS informational-only functions. Our design goal is to automate DDoS response, even to a minimum extent. The Entity is fitted with the capability, after concluding on the security event evaluation, to interact with local network components in a limited and tightly controlled manner. The intended action is the temporary configuration of DDoS attack suppressing filters that will fit the characteristics of the specific security event.

Security sensitive administrators will be quick to point out the risk associated with an automated unit configuring network components. To answer these concerns there are several safeguards in the design. Firstly, the changes are made temporarily, with notifications sent to the management console. Secondly, through the Entity's policy configuration the administrator can exactly define the effects of these changes to the network.

The various Entity operational states and transitions between them are presented in Fig. 4:

Down: Not operating or the administrator has suspended the communications module. It is the only state that the Entity is not transmitting Heartbeats.

Normal: stays in this state while it is normally receiving Heartbeats from the other Entity group members and has not received any Alert messages. The timeout for receiving a Heartbeat is the same with their transmition period.

Suspicion: The Entity comes to this state because of a number of occurrences; either it fails to receive a Heartbeat, or has received an Alert. The latter may be from another Entity or from local IDS sources. Each distinct attack registered at the Entity is receiving an exclusive Notification Counter. Each incoming Alert is

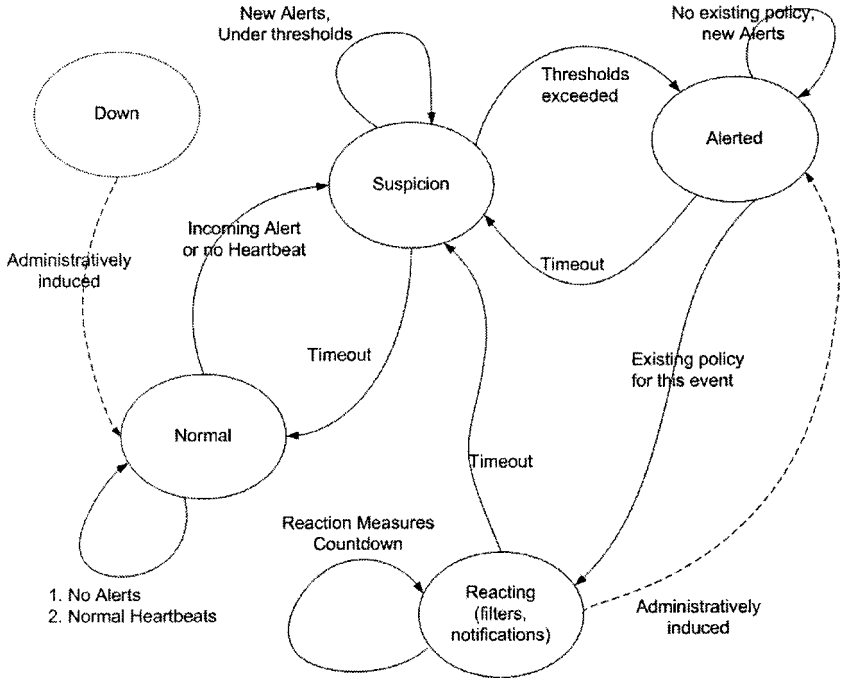

Fig. 4. State transitions of the counter-DDoS entity.

examined for matching any of the on-going events. A positive match will increase the corresponding Notification Counter.

Local IDS messages can result in a multiplied increase to their matching Notification Counter by a factor of **Local_value**. This way they can be treated as higher-value, more trusted than remote Alerts. This feature is useful in networks that become targets of DDoS attacks, to accelerate state changes and broadcast of Alerts for community reaction. Additionally, we expect a higher probability of local IDS notifications in cases that an attack is traversing the domain.

Continuing failure to receive Heartbeats does not increase any of the counters. If new Alerts have not been received for an event and for a **Suspicious_timeout** period the Entity returns to Normal operation and the corresponding Notification Counter is reset.

Alerted: the Entity passes to this state if any attack Notification Counter exceeds the value of **Alert_threshold**. Upon reaching this state a notification is sent to the management console and the policy rules are searched for any entry matching the current attack characteristics. If a corresponding policy entry is not

found then this state is retained for a period of **Alerted_timeout** seconds. The countdown is renewed if during this period another similar Alert arrives. After that the Entity returns to the Suspicion state with the matching Notification Counter set to the value of **Alert_threshold**, decreased by 1.

Reacting: we pass to this state if there is a policy entry matching the detected event. This state is similar to the Alerted state but signifies that reaction measures are being taken. The **Reacting_timeout** parameter of this state is specified in the policy entry and is equivalent to the time that any actions will be applied to network equipment; if the manager monitors the state of the Entity he can know immediately whether an action is currently being taken. New Alerts coming while the entity is in this state don't affect the countdown to the **Reacting_timeout**. Once the timeout is reached we return to the Suspicion state, with the counter set to the value of **Alert_threshold**, decreased by 3 or to the value of 1, whichever is less. We took this decision to prevent the Entity staying in this "active" state indefinitely in the case that Alerts. The Entity may also pass manually back to the Alerted state by the manager which would prompt re-reading the policy and applying any newer rules.

The values of the Entity Parameters: **Suspicious_timeout, Local_value, Alert_threshold, Alerted_timeout,** and **Reacting_timeout**, are set by the administrator and configure the operation of the Entity through the state transition process.

3.2.2. Framework Operation

Entity deployment creates a security focused overlay network. Typical Entity deployment and the Cooperative Framework operation are shown in Fig 5.

During a developing attack the flows of malicious traffic target a network, passing through other domains. DDoS discovery may occur at any of the domains that the attack traverses depending on the detection methods employed and the sensitivity of the ID systems. If the attack is not discovered before, the target network will experience immediate problems with its available bandwidth. This will prompt the local Entity to transmit an Alert or, if loosing connectivity, will result in the stop of Heartbeat messages.

Each Entity that will receive this Alert (or fail to receive the Heartbeat) will combine this information with its local IDS readings to establish whether an attack flow is going through its domain. The combination of arriving Alert notifications about a discovered on-going event with the stop of Heartbeats from parts of the network will cause the Framework Entities to pass to Suspicion and Alerted states. The closer a network is to the victim (if it's on the attack path) the strongest irregularity it will sense in its IDS readings that will produce more local IDS Alerts. Consequently the Framework Entities will be changing to the Alerted state with higher probability as we come closer to the victim. This also ensures that measures will be taken mainly along the attack path where the Alerts will be more

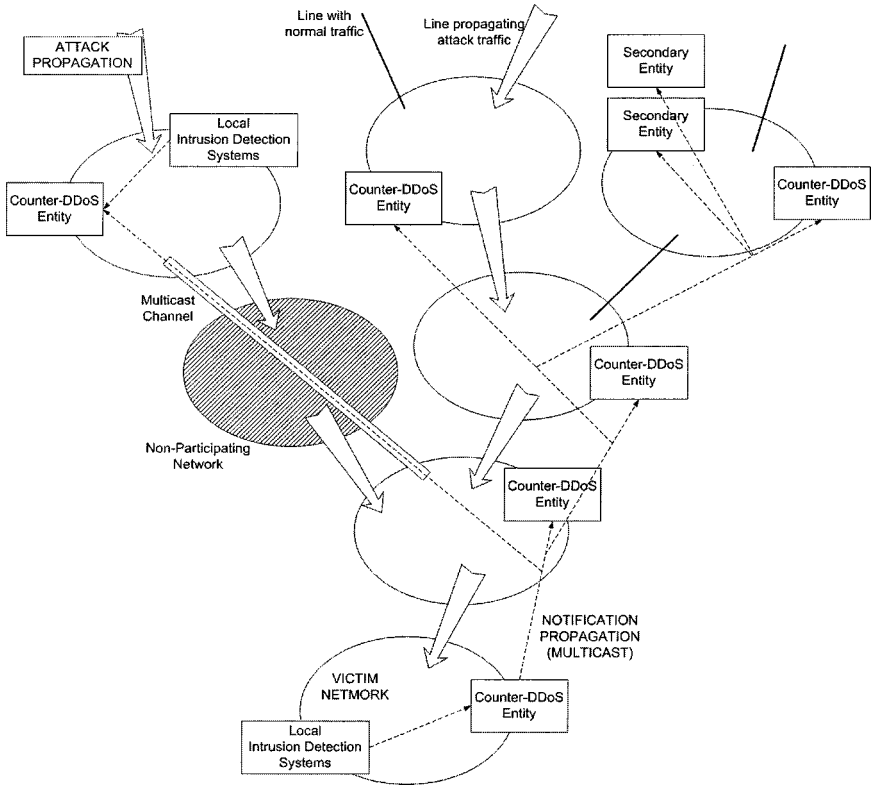

Fig. 5. Typical entity deployment and notification message flow during an attack.

persistent. If the network is indeed a passageway for the attack then, according to the response policies, filters may automatically be set at the border routers and reduce the malicious traffic effects.

4. IMPLEMENTING A PROTOTYPE

4.1. Software Architecture

The software architecture of the Cooperative Counter-DDoS Entity is light-weight and modular. In the prototype we have based the implementation on the Java framework for portability and platform independence. The software architecture of the Entity with message flows is shown in Fig. 6.

The design comprises of a number of independent modules bind together under the administrative features of the Java Management Extensions (JMX) API [23]. Each unit is implemented as an MBean. JMX provides the communication

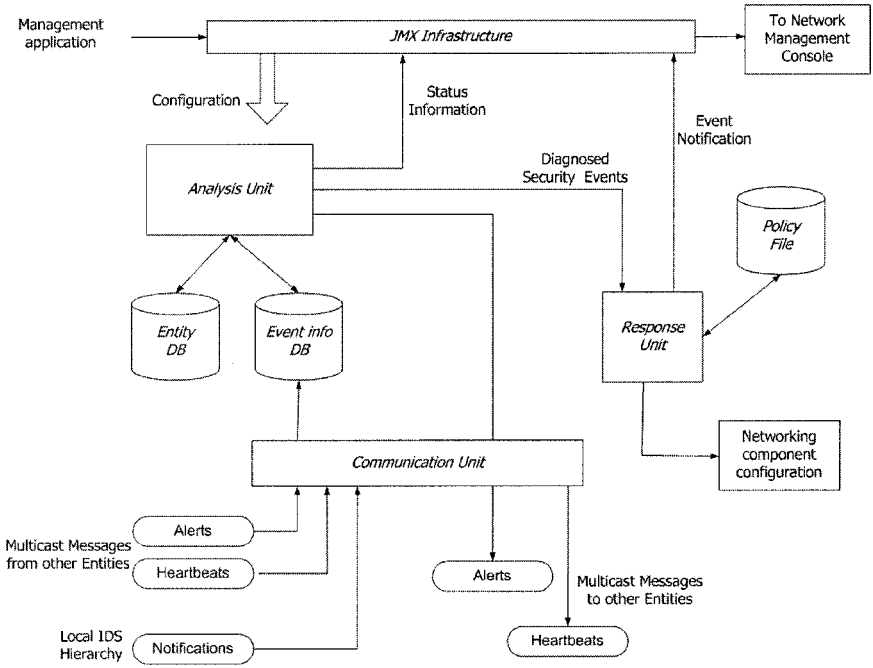

Fig. 6. The software architecture of the entity.

facilities between them and the management infrastructure. A number of different interfaces are available for access to the modules, e.g., a secure HTTP/SSL connection and SNMP. The administrator can change the configuration parameters of whole Entity or control each module individually. The JMX infrastructure allows a number of management actions upon the modules: they can be installed, activated, or deactivated, at run time without affecting the whole Entity.

4.1.1. Message Contents

In order to better understand the functioning of the Analysis and Reaction parts of the Entity we provide here some notes on the Messages exchanged and the specific IDMEF attributes we utilize.

Also, in order to better illustrate the internal operation of the Entity we use the example network, shown in Fig. 7, with an on-going attack targeting one of its leaf domains.

The white ovals represent domains, members of the Cooperative Framework (A, B, C, D, and E) with operating Entities and the gray ovals (X, Y, Z, and W) nonmember domains. The thick arrows represent attack flows against domain D.

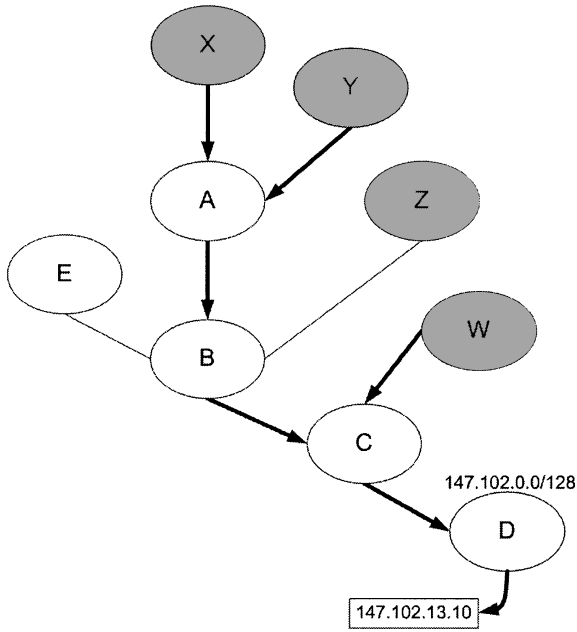

Fig. 7. The Simple Network used in the example.

Both Heartbeats and Alerts are composed of XML tags, structured hierarchically according to the IDMEF Document Type Definition. Heartbeats are coming only from other Entities whereas Alerts may come either from Entities or as local IDS notifications. IDS components may be also transmitting Heartbeats to the Management Console or elsewhere but the Entity is not involved in any system management/maintenance. Code Box 1 presents a typical Heartbeat message sent from the domain A Entity. Code Box 2 is a DDoS Alert message sent from the same Entity. The DDoS is targeting a server (with IP 147.102.13.10) in domain D. The messages may contain more elements but we have included only the ones utilized in the design.

The Heartbeat and Alert *ident* attributes are only used for filing purposes. The unique code for each Entity of the Framework is sent in *analyzerid* and

```
<Heartbeat ident="1234">
<Analyzer analyzerid="Domain_A_Entity-123"></Analyzer>
<CreateTime ntpstamp="0x12345678.0x87654321">2003-04-01T03:44:56,01+02:00
</CreateTime>
</Heartbeat>
```

Box 1. A Typical Heartbeat message.

```
<Alert indent="67890">
<Analyzer analyzerid="Domain_AEntity-123"></Analyzer>

<CreateTime ntpstamp="0x12344321.0x87655678">2003-04-01T03:45:57,03+02:00
</CreateTime>

<Source ident="X">
<Service>
<Portlist>25, 80</Portlist>
</Service>
</Source>

<Source ident="Y">
<Service>
<Portlist>25, 80</Portlist>
</Service>
</Source>

<Target ident="D">
<Node>
<Address category="7">147.102.13.10</Address>
</Node>
</Target>

<Classification origin="1">SYN Attack</Classification>
<AdditionalData type="8", meaning="Description">DoS</AdditionalData>
<AdditionalData type="8", meaning="Next-Hop Domain"><AdditionalData>

</Alert>
```

Box 2. A Typical alert message.

allows identifying the domain that is sending the Alert. It also distinguishes ones coming from a local IDS source. The *ident* code in the Source and Target elements represents domains the Entities know about, members of the Cooperative Framework or nonmembers neighboring. The Entities keep a common list of all these domains, their interconnections, and their codes used in the Messages. Alerts can have multiple *Source* elements but only if they both refer to the same attack. Specific information about the attack is included within the *Service* element and can be a port, a port list and/or a protocol name. The *Classification* and *AdditionalData* elements provide extra information about the attack. Extending the standard IDMEF it is mandatory to have the *AdditionalData* element that shows the "next-hop Domain." This is deduced from IDS attack traffic flow analysis. The *Source*, *Target*, and *AdditionalData* elements provide enough information to identify the positioning of the domain in relation to the attack path.

4.1.2. The Communications Unit

It handles Multicast messaging and incoming message parsing. Messages arriving may be Heartbeats or Alerts. Using the common group key (DES in the

prototype) they are decrypted and then checked for validity by PGP signature and timestamp both of which are part of the XML message. If they do not match the expected values a notification is sent to the management console but no immediate action is taken. In the opposite case (a positive match) the actual IDMEF XML payload passes through a parser. The data, including date info, is stored in the Event Database where it is available to the Analysis Engine and to the manager for further examination. Alerts with more than one *Source* element are stored as separate items. Heartbeats are renewing their time registration in the Database. The Communications Unit is also broadcasting local Heartbeat messages (with the Entity's characteristic identification) within the defined periodic intervals. It also transmits appropriately composed Alert messages using data sent by the Analysis Unit.

4.1.3. The Analysis Unit

This is the inference engine for correlating event reports of the local and Framework sensors. Its operation and its sensitivity are configured by the manager set parameters described in Section 3.2.1 (**Suspicious_timeout, Local_value, Alert_threshold, Alerted_timeout, and Reacting_timeout.**)

The Analysis Unit is responsible for initiating outgoing Alerts. The Framework must avoid Alert replication that would result in traffic buildup. Alerts are produced from the local domain IDS notifications under the following conditions:

1. The Entity must be in the Alerted (or Reacting) state to send an Alert.
2. The notifications must be about a (D)DoS attack

3a. Either the Entity has come to the Alerted state exclusively by local IDS notifications, or

3b. Local notifications about an event have been more than the **Notification_threshold** value within a time period of **Notification_min_time** even if the Entity has received external Alerts about it. The quick accumulation of local notifications indicates the higher possibility the domain to be on the attack path. These two extra configuration parameters do not effect the operation of the Entity itself but rather regulate Alert creation.

4.1.4. The Response Unit

In this Unit events are checked against existing entity Policies. The received Alerts are stored in the Events database as Table I shows on the Entity of Domain B in the example. The table shows information received from domains A, C, and D. All the Alerts have been identified as belonging to the same attack by the combination of Attack Type (derived from the **Service** and **AdditionalData** elements) and final destination (derived from the **Target** element). Examining this table a domain can be in one of the following cases concerning the attack route: (1) it is either the source of the attack or on its path (e.g., as line 3 indicates); (2) it

Table I. Representation of Event Starage in the Events Database

	Alert Sender	Source Domain	Target Domain	Next-hop Domain	Event Type
1	A	X	D	B	1 (SYN attack)
2	A	Y	D	B	1
3	C	B	D	D	1
4	D	C	D	None	1

is on the attack's path (e.g., what lines 1 and 2 indicate); (3) it is the target of the attack; or (4) it is out of the attack path (e.g., indication of line 4)

Once a particular event has raised the Entity's state to Alerted (as described in Section 3) the Response Unit searches the database entries referring to it and establishes the positioning of the local domain on the attack path, as described before. Summing up all the entries, the positioning Case can be any of 1, 2, 3, or 4, or a combination of Cases 1 and 2 (possible source of the attack and on its path), or Cases 2 and 3 (on the path of the attack and its target). This attribute (positioning in the path of the attack), the attack type, and attack destination are used for matching the event with a configured policy entry (e.g., throttling traffic of these characteristics, this on the ingress router in the direction of the victim.)

The policy entry refers to the type of action that will be taken. Currently this can be traffic blocking or throttling/shaping. These traffic restrictions referring to the type of attack traffic and its destination will be applied on domain border routers. Each policy entry that refers to an action also includes the period of time it will be in force which is the corresponding **Reacting_timeout** configuration parameter. The Entity does not have to choose identify the components to act upon and does not even need to have the topology of the domain's internal network.

The Response Unit undertakes messaging to the Management Console (through JMX) each time the entity reaches the Alerted state irrespective to any reaction measures.

5. RELATED WORK

The Global Defense Infrastructure (GDI) proposed by Wan and Chang [11] (also see Wan [24]) is an approach against DDoS attacks most close to ours. In their proposal "Minimally" and "Fully" configured Local Detection Systems (LDSes) are to be placed at various strategic locations in the entire Internet, like Network Access Points, etc. constituting the Distributed Attack Detection System (DAD). The distributed architecture is supported by messaging between the LDSes. The LDSes have modular design, with each one of the modules (for traffic analysis, attack detection, attack response etc.) occupying a different computer system. The FLDS perform misuse and anomaly based DDoS detection based both on their own

findings and indicative alerts coming from neighboring systems. Suspicious attack alerts are communicated within the GDI using a reliable flooding mechanism. The messages are formed in IDMEF. Once attack detection has been established they install rate-limiting filters at the Internet core router they are connected to. MLDSes are a lighter implementation of the system used only for installing rate-limiting anti-DDoS filters based on information they receive from FLDSes.

Although there are many similarities in the design and operation of GDI with our approach (IDMEF messages, modular systems, communicating findings in the community, and setting up rate-limiting filters) the two solutions are different in their management goals, scale of deployment, and detection methods. Our Cooperative Framework constitutes a system for the management and coordination of anti-DDoS efforts within small and trusted groupings of domains. Another difference is that the Entity used in our design is part of the domain's administrative realm, permitting high level of trust and configuration according to local policies. We do not perform any detection on our own but rather rely on the findings of the local ID Systems at each site. It could be argued that the different IDSs would result in non-uniform detection rates in the Framework. We believe that the different approaches used by the deployed IDSs compensate for this handicap by allowing a variety of approaches in the detection procedure. Alert exchanges make all the Infrastructure's members aware of their findings.

The Cooperative Intrusion Traceback and Response (CITRA) framework [12,13] is also work close to ours. It uses the concept of communities (administrative domains) and all of them organized in neighborhoods. At each community it performs low-level intrusion detection, focusing at the boundaries. The detectors distribute attack reports to their neighbors who can then trace the attack path and initiate intrusion responses. CITRA employs device independent response directives and uses centralized reporting and coordination. The communications take place using the Intruder Detection and Isolation Protocol (IDIP). This approach also has a number of features similar to ours like response policies and the possibility of using multicast. The main differences of our proposal are that (a) we operate on a higher level and focus in automation and acceleration of enterprise cooperation; (b) in the communication part we use the functionality of the multicast method as the prime element of our architecture; (c) we have based our implementation on the IDMEF protocol for a standardized and easy integration with existing and future IDS components; and (d) we take a liberal approach to domain participation, not depending in the seamless integration of every one to achieve an effective solution.

Ioannidis and Bellovin [15] present their solution of controlling the high bandwidth aggregates that comprise a DDoS attack. The approach utilizes routers both for attack detection and response. DDoS attack traffic aggregates are established by monitoring discarded (overflow) traffic at router level that includes the attack packets. The system then moves on to block these traffic aggregates at the

routers using tailored filters. The findings are communicated between cooperating routers using the special Pushback protocol. As a result the malicious aggregates are traced step-by-step closer to their sources and their bandwidth allocation controlled. The difference in our work is the higher level approach to the problem. We focus our effort in providing a cooperation-enabling infrastructure for domains and we are independent from the intrusion detection components. Another significant difference of the Pushback approach is that it requires administrative access to each individual router, which causes problems when expanded further than the border of a domain. Furthermore, tracing back on a low-level requires the collaboration of every intermediate device in the path from the victim on the way to the source. In our approach we have tried to minimize and overcome the problem of nonparticipating domains.

The Indra system [25], also utilizes multicast communications for distributed intrusion detection and response operations but this is an approach within a single domain. Specifically, IDS hosts that have identified an attack attempt, use secure multicast messages, to communicate its characteristics to other nodes in the network notifying them in advance. Having faced similar problems to ours in the security authentication of multicast messages they have developed a secure messaging API offering cryptography.

Finally many concepts discussed here are also presented by Frincke *et al.* [26], where the prerequisites for forming a cooperative intrusion detection framework are discussed. Although that work does not deal with the problems of countering denial-of-service attacks or usage of multicast communications for building an overlay network, there are many similarities with our approach in the elements that form a secure and effective basis for cooperation, like the trust concerns between domains.

6. CONCLUSIONS

We have presented a distributed framework that introduces a cooperative inter-domain approach in countering the DDoS problem. We have also presented the implementation details of the main enabling Entity (based on open and accepted standards) and discussed its operation. Finally, we have provided an overview of related work and compared it to our approach.

The operation of our approach relies on building a “community” of trusted partners, each deploying a local software Entity. Entities exchange security information (Heartbeat and Alert messages) so that inclusion in the attack path is detected locally and without requiring traceback procedures. Reaction is activated in parallel, controlled at each domain by local policies. The proposed architecture is not an intrusion detection system, but rather a “detection message management system and coordinated reaction framework” independent of the underlying detection technologies.

Currently we have implemented a prototype Entity and we intend to test it in real-world conditions. Our experiments and analysis of the prototype have identified the following strong points of the proposed framework: (a) the Entities demonstrate a high degree of autonomy and automate manual cooperation activities which in an operational situation would greatly minimize reaction time; (b) local reaction is supported and based on locally-defined policy rules, which can achieve the parallel operation of Entities without exposing the network to threats outside the domain; (c) the Entity communicates effectively via multicast without having to know or exposing the topology of the "overlay network"; (d) Entity messages (Heartbeats and Alerts) were successfully relayed to an Entity operating in an adjacent LAN thus demonstrating the concept of no single point of failure; the second Entity could be interchanged with the "primary" one offering services within the same domain; and (e) under normal circumstances there is a low message rate but when needed there is an escalation of event notifications, mimicking security organizations cooperation.

Future work includes the large-scale validation through WAN emulation and the experimental deployment within the Greek Research Network. We also consider investigating the use of reserved bandwidth (level 2) tunnels between cooperating Entities for avoiding "multicast tree collapse" problems.

REFERENCES

1. G. Koutepas, F. Stamatelopoulos, and B. Maglaris, Efficiency and performance issues in distributed intrusion detection systems, Applied Telecommunication Symposium 2002 (ATS 02), San Diego, California, April 2002.
2. D. Moore, G. Voelker, and S. Savage, Inferring Internet denial-of-service activity, Proc. Tenth USENIX Sec. Symp., 2001.
3. Minho Sung, Markus Haas, and Jun Xu, Analysis of DoS attack traffic data, 2002 FIRST Conference (www.first.org), Hawaii, June 2002.
4. R. Chang, Defending against flooding-based distributed denial-of-service attacks: A tutorial, *IEEE Communications Magazine*, pp. 42–51, October 2001.
5. J. Mirkovic, J. Martin, and P. Reiher, A Taxonomy of DDoS attacks and DDoS defense mechanisms, University of California, Technical Report#020018 2002. (Also available at *http://www.lasr.cs.ucla.edu/ddos/ucla_tech_report_020018.pdf*)
6. M. Behringer, Tracing DoS attacks, Hi Tech 2002 Workshop, Limmerick, IE, June 2002.
7. R. Manajan, S. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, Controlling high bandwidth aggregates in the network, *ACM SIGCOMM Computer Communication Review*, Vol. 32, No. 3, pp. 62–73, 2002.
8. C. Estan and G. Varghese, "New directions in traffic measurement and accounting, Proceedings of the 2001 ACM SIGCOMM Internet Measurement Workshop, pp. 75–80, (San Francisco, California), November 2001.
9. Cisco IOS NetFlow, *http://www.cisco.com/warp/public/732/Tech/nmp/netflow/index.shtml*
10. C. Kotsokalis, D. Kalogeras, and B. Maglaris, Router-based detection of DoS and DDoS attacks, HP OpenView University Association (HPOVUA) Conference '01, Berlin, Germany, June 2001.
11. K. K. Wan and R. Chang, Engineering of a global defence infrastructure for DDoS attacks, Proc. of IEEE International Conference on Networking, August 2002.

12. D. Sterne, K. Djahandari, B. Wilson, B. Babson, D. Schnackenberg, H. Holliday, and T. Reid, Autonomic response to distributed denial-of-service attacks, Proceedings of the Fourth International Symposium on Recent Advances in Intrusion Detection, RAID 2001, Davis, California, pp. 134–149, October 2001.
13. D. Schnackenberg, K. Djahandari, and D. Sterne, Infrastructure for intrusion detection and response, Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX II), Anaheim, California, January 2000.
14. A. Snoeren, C. Partridge, L. Sanchez, C. Jones, F. Tchakountio, S. Kent, and W. Strayer, Hash-based IP traceback, Proceedings of the ACM SIGCOMM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, August 2001.
15. J. Ioannidis and S. Bellovin, Implementing pushback: Router-based defense against DDoS attacks, Network and Distributed System Security Symposium, NDSS '02, San Diego, California, February 2002.
16. S. Savage, D. Wetherall, A. Karlin, and T. Anderson, Practical network support for IP traceback, Proceedings of the ACM SIGCOMM Conference, Stockholm, Sweden, pp. 295–306, August 2000.
17. D. Dean, M. Franklin, and A. Stubblefield, An algebraic approach to IP traceback, Network and Distributed System Security Symposium, NDSS '01, February 2001.
18. R. Stone, CenterTrack: An IP overlay network for tracking DoS floods, Ninth USENIX Security Symposium, Denver Colorado, August 2000.
19. K. Park and H. Lee, On the effectiveness of route-based packet filtering for distributed DDoS attack prevention in power-law internets, Proc. of the ACM SIGCOMM, 2001.
20. Trusted Introducer for CSIRTs in Europe, *http://www.ti.terena.nl/*
21. K. Almeroth, The evolution of multicast: From the MBone to inter-domain multicast to Internet2 deployment, *IEEE Network*, January /February 2000.
22. D. Curry and H. Debar, Intrusion detection message exchange format data model and extensible Markup Language (XML) document type definition, IETF Internet Draft, *draft-ietf-idwg-idmef-xml-10.txt*, January 2003.
23. Sun Microsystems, Java Management Extensions Instrumentation and Agent Specification, v1. 2, February 2002. *http://jcp.org/aboutJava/communityprocess/final/jsr003/index3.html*
24. K. Wan, An infrastructure to defend against distributed denial-of-service attack, M.Sc. Thesis, The Hong Kong Polytechnic University, June 2001.
25. Q. Zhang and R. Janakiraman, Indra: A distributed approach to network intrusion detection and prevention, Washington University Technical Report # WUCS-01-30, 2001.
26. D. Frincke, D. Tobin, J. McConnell, J. Marconi, and D. Polla, A framework for cooperative intrusion detection, Proceedings of the 21st National Information Systems Security Conference, pp. 361–373, October 1998.

G. Koutepas holds a BA in Electrical and Computer Engineering from the National Technical University of Athens (NTUA), Greece. He is currently pursuing a Ph.D. at NTUA with research work on effective inter-domain DDoS detection and containment.

F. Stamatelopoulos is a researcher at the Institute of Communication and Computer Systems at NTUA. He holds a Ph.D. in Electrical and Computer Engineering from NTUA and an M.Sc. in Computer Science from University of Maryland. His current research interests include network and system management and security, distributed systems and architectures.

B. Maglaris is a professor at the Dept. of Electrical and Computer Engineering at NTUA. He holds a Ph.D. degree in Electrical Engineering and Computer Science from Columbia University, New York. He is teaching and performing research on communication networks, with focus on operations and management, queuing theory, performance models, and optimal design.