A Proactive Distributed Denial of Service Protection Framework

M. Eyrich, A. Hess, G. Schäfer, L. Wartenberg
Telecommunication Networks Group, Technical University Berlin
Einsteinufer 25, 10587 Berlin, Germany
[eyrich, hess, schaefer, wartenberg]@tkn.tu-berlin.de
15 November 2004

Abstract

Securing communication networks against distributed denial of service attacks (DDoS) is still one of the most challenging network security issues. We propose a framework to protect network routers and hosts against resource starvation caused by DDoS attacks. We pro-actively build overlay groups of neighboring enhanced routers according to current traffic patterns. During ongoing attacks, the framework provides knowledge and mechanisms to forward alerts and mitigation rules towards the attacking traffic sources. Within the steadily growing protected area, we are able to release the mitigation rules in order to resume normal network operation. In this paper we present architecture and auto-configuration mechanisms of the framework including communication protocols and messages.

Keywords: DDoS, Traffic Control, Overlay Network.

1. Introduction

Denial of Service (DoS) attacks aim at denying or degrading a legitimate user's access to a service or network resource, or at bringing down the servers offering such services itself. In the last several years DoS attacks have increasingly become a major problem of computer security. Internet denial-of-service attacks have increased in frequency, severity and sophistication. Between the years of 1989 and 1995, the number of such attacks reported to the Computer Emergency Response Team (CERT) increased by 50 percent per year [2]. According to a 1999 CSI/FBI survey report 32 percent of respondents detected denial-of-service attacks directed against them [3]. To make things worse, already in the year 2000 reports [8] indicated that attackers are more and more developing tools to coordinate distributed attacks from many separate sites, which is also known as *Distributed Denial of Service (DDoS)*.

Current figures given by CERT [9] confirm this rather negative development in terms of numbers of network attacks. The number of registered incidents for the year 2000 was 21,756, for the year 2002 already 82,094, and in the year 2003 the CERT received 137,529 incident reports. Beyond this, also the diversity of software is increasing and still the quality of many software solutions is insufficient, especially in terms of security vulnerabilities resulting from programming errors. In the year 2000, the CERT registered 1,090 vulnerabilities, 2,437 vulnerabilities in 2001, and 4,129 vulnerabilities in 2002. The number reported in this year was slightly smaller with 3,784 vulnerabilities, which at least points to a change in trend, but nevertheless represents an alarming fact.

These problems are aggravated by an inappropriate security awareness of many network and system administrators as well as users which has (again) been clearly shown by the W32/Blaster worm [10]. The worm which started on August 11th 2003 exploited a vulnerability that has already been known four weeks earlier. Actually, since July 16th 2003 Microsoft had provided a patch in order to fix this flaw. But still, the worm could diffuse itself in a manner such that Symantec classed it as category 4 (severe threat, global distribution). As a consequence thereof, we can clearly see that the idea of quickly patching all vulnerable systems upon detection of a new security hole is not an appropriate measure to cope with the evolution of execution speed of computer attacks, and that, therefore, attackers will continue to be able to break into systems and deploy them for their purposes in the future.

DDoS attackers make use of the joint power of multiple systems when launching an attack. Therefore, even servers with high amount of resources and high bandwidth connections are vulnerable to such attacks. In addition to the attacker and the victim, a DDoS network consists of so called master- and slave-systems. The slave systems are the actual offenders. They are not controlled directly by the attacker, but by the master systems.

Detecting and defending against a DDoS attack at the target is difficult due to the fact, that the attacking traffic can hardly be differentiated from normal traffic and that the target system is usually not capable anymore of taking any protective measure, as it is completely overloaded with malicious packets — in severe cases even entire network links can get congested. In the worst case, the attacking traffic consists of correct IP-packets with spoofed source addresses and random payload - the only trustworthy message element is the target address.

In this paper, we therefore describe an approach to protect network routers and hosts against resource starvation caused by DDoS attacks, which allows to detect and mitigate occurring attacks in network elements (with added DDoS protection functionality). The paper is structured as follows: the next section discusses general DDoS defense strategies and related work, and points out some requirements for effective DDoS protection that have not been fulfilled by existing approaches. Section 3 starts with motivating additional properties of a DDoS protection framework and then explains our basic approach, its components and protocol operation. Section 4 describes a prototypical implementation of our framework and presents a preliminary performance assessment. In the final section 6, we draw some conclusions and outline potential directions for future work.

2. General DDoS Defense Strategies and Related Work

There are several means that have been proposed so far to effectively defend against DoS attacks and that can be classified in the following way:

- *General prevention:* these are all measures, that make the attacker's life more difficult. They comprise for example filtering rules against IP spoofing or patches that fix bugs and vulnerabilities in software implementations.
- *Recognition and reporting of DoS attacks:* is an obvious requirement in order to initiate actions against an ongoing DoS attack. In practice, however, this turns out to be much more difficult than it sounds.
- *Defending against occurring attacks:* comprises the actions to mitigate the harm while an attack occurs. Three steps can be distinguished: (i) means to make the attack ineffective (e.g. filtering packets, change of configuration), (ii) tracing back the attack to its source so that the attacker can be identified, and (iii) after the source is identified it might be possible to stop the attack(er), e.g. by denying network access.

Regarding *general prevention*, it should be ensured that vulnerabilities are removed as soon as they are discovered. The system software of all servers and network elements should be kept up-to-date in order to cope with newly discovered vulnerabilities of networking software and to resist against the more intelligent software attacks that enable attackers to take control over hosts and turn them into slave systems for DDoS attacks.

Concerning *recognition and reporting* of DoS attacks, it has to be stated that despite the considerable harm caused by flooding attacks, it is either not trivial for the victim to recognize that it is actually being attacked, i.e. to distinguish normal and malicious traffic, as in some cases the victim still can access services on the Internet (with degraded performance), whereas in other cases the victim's connectivity might be completely disrupted, making it easy to detect the DoS situation but impossible for the victim to report an alarm to any networking entity.

It is rather difficult to defend against DDoS attacks. Router configurations and filtering rules can possibly be adopted to mitigate the harm at the victim's side. However, because of the nature of flooding attacks with spoofed IP addresses and the brute force of Distributed DoS attacks it is often not sufficient to defend against the attack only in network parts that are close to the victim. Therefore, various approaches to trace back malicious traffic to its actual source and to defend against it close to the source have been proposed.

One of these DDoS mitigation techniques is the so-called *pushback mechanism* on top of enhanced routers [4]. In this approach, routers are successively informed to filter traffic destined to a specific address. However, the approach assumes that all routers inside a network are of the same type and it does not treat the difficulty of networks consisting of routers with different capabilities.

The Active Network Based DDoS Defense [7] is an approach that uses the possibilities provided by an underlying active networking infrastructure. In each domain a central management station is established to which alerts are sent. An alert is generated by a traffic rate monitoring application which is running on the host to be protected. According to the reception of an alert, a central management station sends an active program to the active router nearest to the victim. The active program filters the traffic and further on, it creates copies of itself. These copies move upstream towards the sources of the attack. However, the central management station presents a single point of failure inside a domain. In the case that the central management station gets attacked no helping mobile agents can be dispatched. Furthermore, if a well chosen router becomes the target of an attack it is possible to cut off the path between active routers and central management station.

Another frequently mentioned DDoS-defense technique is the so-called traceback technique. For example in [6] the authors describe a traceback mechanism which probabilistically marks packets with partial path information. In the case of an attack, a victim, after having received a sufficient amount of packets, is able to reconstruct the entire path of the attack traffic. A further traceback approach is the utilization of ICMP-traceback messages [1]. About every 20,000 packets a router creates an ICMP traceback message and addresses it to the same destination as the selected packet, inserting its own source address. In case of an attack these ICMP traceback messages are evaluated in order to reconstruct the path of the attack traffic.

Both mentioned traceback mechanisms are of a rather reactive nature, as they start acquiring mandatory DDoS mitigation knowledge when the attack is currently taking place or when the attack has stopped.

3. Active Distributed DoS Blocker Framework

An effective DDoS protection framework should ideally fulfill a couple of requirements: First, efficient DDoS protection mechanisms have to be located in the network itself, at a point where it is still possible to react while an end host or other parts of the network might already be disabled by an attack. Furthermore, a net-centric approach is detached from users and administrators. Second, a DDoS protection mechanism should be realized with a distributed architecture. This avoids the existence of a single point of failure — to our understanding a fundamental requirement — as it is much more difficult to attack a distributed mechanism consisting of numerous entities which have the same set of rights and capabilities. Third, goal of a DDoS protection mechanism must be to block the attacking traffic as close as possible to its actual origin. A distributed protection architecture should be realized such that it can provide backtracking information on demand that is needed for mitigating a DDoS attack. A further requirement is a system which configures itself automatically according to the current circumstances. In order to quickly react upon the detection of a DDoS attack, it is mandatory that the protection system works without input of a network administrator. Moreover, the system should not require initial knowledge in network elements besides common routing tables so that no expensive and manual configuration process is needed for deployment. Furthermore, the mechanism must be realized as efficient as possible such that it does not noticeably influence the network performance, and it should be possible to deploy the mechanism in a heterogenous architecture comprising of *enhanced* and *traditional* routers.

In order to fulfill the abovementioned requirements, we designed our approach for distributed denial of service protection to: (a) proactively providing a structure of the main network paths surrounding a participating system, (b) enabling upstream enhanced routers to recognize a running attack, even if it is not in the main path of the attack, (c) quickly disburdening an attacked system, (d) at the same time tracking down the real sources of an attack, (e) quickly returning to normal operation for areas of the network, where no (more) attack traffic is flowing, and finally, (f) keeping access restrictions as limited as possible.

Our proposed framework attains these goals with the following mechanisms. Ad (a): During normal, unsuspicious operation of the network we analyze traversing traffic in order to detect endpoints that start to communicate. Based on these destinations we build groups of neighbored enhanced routers—whose overlay distance is limited to a single hop—by sending *search* messages towards those targets, which will be caught by the next enhanced router on the propagation path. Within each of those groups, information about an ongoing attack — including the target addresses or address ranges — can quickly be distributed. As a result, enhanced routers are able to immediately block or mitigate the respective traffic. Ad (b and c): The center of a group and its members mutually exchange pulses to provide its current state of aliveness. If an enhanced router fails to receive such pulses within the expected intervals it throttles the targets serviced by it (this is the reason why we need to proactively collect the serviced address ranges). Ad (d): An enhanced router which recognized an attack, propagates adaptable traffic mitigation rules step by step upstream to be placed nearby the attackers. The propagation can fol-

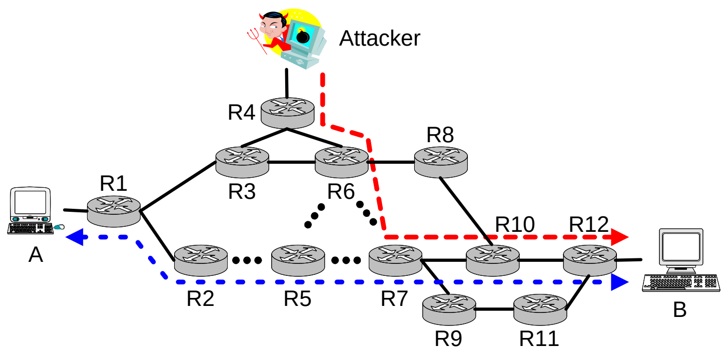


Figure 1: An Example Scenario

low several diverging paths. Ad (e): Resuming sending of pulses without mitigation rules quickly leads to normal network operation in the areas where an attack was initially detected, while the mitigation rules are still propagating towards the sources of the attack. Figuratively spoken, think of a wall moving like circles away from a stone thrown into water, where the stone is the target of an attack. Ad (f): Initial reaction on a lost neighbor is the mitigation of all traffic the neighbor is in care of. As soon as the neighbor is reachable it is able to refine the initial broad mitigation rules to the attacked target addresses only. Additionally, our proposed framework has the following features:

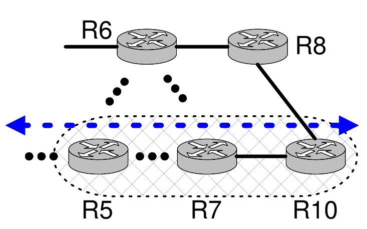
- it does not rely on any initial knowledge about neighbors besides local routing tables;
- it can be established on top of current routers if there are certain interfaces available (a sniffing interface to catch certain packets; a control interface to set traffic mitigation rules; an interface to get the current routing table (e.g. SNMP)); putting the active components besides the main router provides fail safeness if the control system gets broken;
- there are no changes to existing protocols necessary (like changing the fragments field. . .);
- our demand driven approach further:
  - allows the network to react autonomously on changing network conditions,
  - creates a minimal network load during ongoing attacks,
  - allows traffic mitigation to the attacked network even if it cannot send any more messages,

3.1. The Components

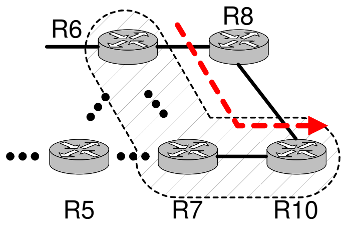
Our Active Distributed Denial of Service Blocker (ADDOSBLOCKER) framework basically consists of enhanced routers which are distributed over the Internet (see Figure 1). To give a concise overview of our approach, the following protocol functionalities can be distinguished:

- detecting the need to discover a neighborhood relationship regarding a specific (aggregated) network flow, achieved with the so-called Demand Driven Overlay Neighborhood Establishment (DDONE) process;
- identifying neighborhood relationships using search messages;
- keeping neighborhood knowledge up-to-date using pulse messages;
- actually triggering defense mechanisms with ALERT messages.

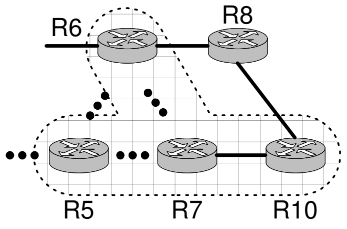
In order to reduce pre-configuration within the network, enhanced routers do not have knowledge about each other. Instead they use the DDONE mechanism in order to create and configure inter-network dependencies (section 3.2). By watching passing traffic an enhanced router (e.g., R1) recognizes a previously unknown target (host B). It starts to discover “neighbors” using a special active message, a so called search, directed to the unknown target. A search is recognized by the next enhanced router on its way (say R2), which registers with its predecessor R1 to build a group. R2 and all following enhanced routers behave the same way such that, for instance, R2 recognizes R1 and R5 as its neighbors and, therefore, as members of its group. Each one of the enhanced routers is finally the center of a group with its own view of a group such that no two groups are identical. A Last Mile Active Router (LMAR) has an extended functionality such that it is



(a) First step



(b) Second Step



(c) Final Group

Figure 2: Group Establishment Process (●●● Intermediate Routers, ▬▬ Overlay Path)

able to further observe the traffic volume that is transmitted to specific end hosts and services (see also sections 3.1.1 and 6). Summarizing the enhanced routers build a dynamically adaptable overlay network which is used for DDoS detection and mitigation.

3.1.1. Last Mile Active Routers and Enhanced Routers An enhanced router is the basic unit of the DDoS mitigation framework. More precisely, two slightly different variations exist: Last Mile Active Router (LMAR) and Enhanced Router (ER). Looking at figure 1 on the page before, the enhanced routers R1, R4 and R12 are LMARs whereas the others are ERs. LMARs and ERs both have the following set of responsibilities. Each enhanced router communicates its individual state to its neighbors by sending, according to a defined scheme, pulse messages to all members of its group. The pulse mechanism is both a means for mutual observation of participating enhanced routers and a means to detect traffic thresholds in the network. A detailed discussion is given in section 3.2.

Next, each enhanced router is responsible for invoking on demand the neighborhood generation process. A demand arises through the event of a new connection target which is not registered yet or due to an alteration of the local routing table. Thus, it is mandatory that each enhanced router analyzes the passing traffic and observes its local routing table. Moreover, each enhanced router keeps track of its neighborhoods in the neighborhood table. An example for the neighborhood table of router R1 is given in table 1. The first column memorizes the aggregated destinations for which a next neighbor has already been determined. From the perspective of an enhanced router which is in charge of protecting network resources against DDoS attacks, the only data field to be trusted is the destination IP. The second entry stores the address of the neighbor who routes packets to the aggregated targets. The level field informs about the load of the corresponding interface of the neighbor. Finally, the options field contains an extensible list of required entries. For example a timestamp provides the time when the last packet has been routed to an aggregated destination.

The difference between an LMAR and an ER is the fact that an LMAR is the access router for a specific group of hosts and services. More precisely, an LMAR observes the rate of traffic destined to these hosts and services. In case that a specified traffic volume exceeds the critical threshold the LMAR launches an alarm. An alarm consists of sending an alert message to the upstream neighbor and of starting a packet filtering service.

Currently, the individual threshold data-rates for end-systems and services are stored in a local security policy on the corresponding LMAR. Consequently, a network administrator — who is the only authorized person to edit the security policy — is still required. A more sophisticated registration mechanism is scope of future work and is shortly outlined in section 6. Further on, the network administrator specifies how to respond to a detected DDoS attack, by defining the packet filtering service which must be started in order to mitigate the attack. As the response mechanisms are realized as active networking services, it is possible to dynamically latch different packet filtering responses on the enhanced routers.

3.1.2. The Mitigation Initiation API The mitigation initiation API provides the capability to dynamically plug-in a given mitigation rule. Its main mechanism is the ability to communicate addresses or ranges of addresses to participating routers. Therefore, our framework provides the service to distribute objects between group members. These objects, for instance, contain an address range and the function to be applied to

Table 1: neighborhood table of R1

Routing Entry of Neighbor	Neighbor IP	Level	Options
162.67.201.0/24	R2	199	12:53:13.564533, ...
58.46.12.0/16	R3	213	12:53:13.832566, ...
192.101.133.0/24	R2	61	12:53:11.832286, ...

the packets belonging to the given address range. The function may be just dropping packets up to a given rate or to drop every other connection establishment attempt. The framework selectively transmits mitigation requests to its neighbors, as long as there are packets of the given address range traversing the current router.

3.2. Demand Driven Overlay Neighborhood Establishment

The Demand Driven Overlay Neighborhood Establishment (DDONE) is the a mechanism to establish specific groups of enhanced routers for the purpose of a mutual exchange of state information and an automatic and mutual monitoring of its (active) neighbors. This, enables an enhanced router to recognize an overloaded successor node and to throttle traffic destined to targets behind this node, even if the node is completely unable to actively request such a throttling.

The basic instruments of a DDONE are the abilities to (i) scan passing traffic to trigger a DDONE, (ii) to recognize active messages, and (iii) to get knowledge of the local routing table. These instruments will be explained in the following sections.

A DDONE is generally triggered by new connections, for instance, a TCP message with the SYN flag set, or by a new target for a UDP connection. The enhanced router compares the triggering event with its routing table and its neighbor table to decide if there is already a known neighbor. If no neighbor is found, the enhanced router (say R1) starts a neighbor discovery by sending search messages.

Search Messages are generated to discover a potential neighbor. The main principle here is that different types of packets usually follow the same path to their destination. If a new destination is recognized, a search message is sent to the same target as the original packet with a maximum Time To Live (TTL) of 255. A search message is expected to be inspected by any enhanced router along the path and to be evaluated by the next enhanced router on the subsequent path of a packet (e.g., R2 in figure 1). Usually, the next enhanced router (R2) should take the evaluated search request off the network.

All members of the overlay group are in distance of a single hop to the center of the group (see figure 2. Common, non-overlay protocols such as Open Shortest Path First (OSPF) [5] use an approach which utilizes (in the context of IP multicast) the IP TTL field to restrict packets to a known distance. For the establishment of an overlay network the distance to the next member is not known in advance, i.e., it is not known whether a TTL of, say, five, or 15 is necessary. Therefore, the TTL cannot be used to limit messages to a single hop in the view of the overlay network. Instead, another mechanism is needed. Recalling the ability of enhanced routers to change packets on the fly it can be used to construct such a behavior by just refraining from forwarding the packet.

Search(es) are sent using UDP packets on a special port. In order to get the same path as the initial trigger packet, searches are sent to the same destination as the trigger. Thus, enhanced routers are required to inspect every passing UDP packet that is addressed to the special port.

After receiving a search request it is the duty of R2 to contact R1. R1 expects replies (pulse messages) within a fixed amount of time, and repeats the search three times in short intervals if there is no response. Otherwise, if the destination is still active, a search is repeated in longer intervals to catch rerouting or new systems in the subsequent part of the network. The lack of a reply is a sign for one of the following situations. Either there is no other enhanced router within the subsequent path to the destination or there is an asymmetry in the visible part of the Internet leading to responses not being returned to the querying enhanced router.

Both results indicate that the search originator is the last enhanced router towards the target system. Yet, an asymmetrical routing of packets—as opposed to the asymmetry in the visible part above—is easily caught by the proposed architecture due to the direct addressing of neighbor enhanced routers in search replies and pulse messages.

Responses can be sent via UDP as well as via TCP. However, UDP is preferred due to lower network load and lower requirements in system resources. The main advantage of TCP is the implicit three-way handshake. Yet, the sequence of request, response and pulse message (section 3.2) is also a three-way handshake but does not require a node to hold a huge number of TCP-endpoints in case of a DDoS attack against the framework itself (the enhanced routers cannot be put into having many sockets in TIMEWAIT state).

Search Reply A search reply is used to initially transfer local state information such as relevant parts of the routing table to neighbors. The type of information is similar to the format of pulse messages, although the initial information transfer will require larger objects while pulse messages are used to transfer state deltas between two events. Therefore, a search reply is just a special case of a pulse message (see also the following section).

Pulse messages are the basic message format to update each enhanced router's neighborhood table. These messages are directed to all members of a group using a group communication mechanism (e.g., IP multicast in the access network, end system multicast in non-multicast-capable parts of a network). Its main components are the IP address of the sending router, a time stamp for a possible re-ordering of messages and an authentication facility. These fields are necessary to recognize over-flooded destinations. In the additional data area the current state of the router is communicated either as a complete table (on request or due to a new neighborhood), or in a compressed form as a delta to previous values, where deltas are numbered to recognize lost pulse messages (this is especially necessary for the removal of table entries). Deltas should be included in several consecutive pulse messages in case one of them was lost. The pulse interval depends on the current network situation. In case of high network loads a higher frequency of pulse messages allows for quicker responses in case of a network overloading. Pulse messages are always exchanged between enhanced routers in both directions. However the frequency of pulses is predefined by the sending router, thus it is not identical for both directions.

Pulse messages are required to be sent with a constant initial TTL. The TTL of received packets is evaluated to catch changes in transit networks between two neighboring enhanced routers. Changes in the routing to a destination will most probably result in a different hop count for sent packets, which allows the receiving enhanced router to re-initiate a DDONE.

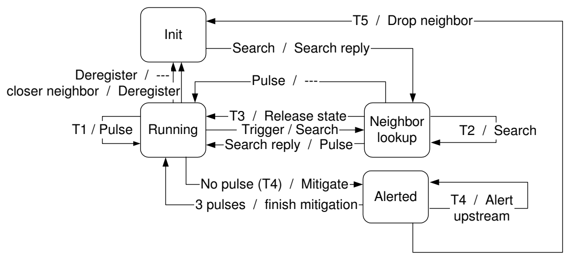


Figure 3: DDONE State Machine (Event / Action)

ALERT Messages Alert messages are sent containing the relevant alerts as objects in the data area. The object's parameters are: (i) The destination address and prefix length either as given by the neighborhood table of the initial requesting router or, if provided by the LMAR, the address of the attacked end host including protocol and port (zero values indicate wild-cards); (ii) the requested mitigation level in percent as a suggestion for other enhanced routers; (iii) The live-time of the alert; and finally, (iv) a unique identifier in order to avoid alert loops. An upper bound in the number of objects is given by the size of the UDP packets.

Alert execution is started by first briefing the mitigation initiation API to plug in a proper traffic mitigation rule. At the same time, it creates and sends the above described alert messages to its group members. Each group member starts the corresponding mitigation plug-in and next, verifies whether it is in charge of delivering traffic to the victim. Those who do so, forward the alert message to their group neighbors, such that the alert notification propagates towards the traffic sources.

3.3. State Machine

Every member of the DDoS mitigation framework maintains a state machine to every of its group members as illustrated in figure 3. After booting, an enhanced router is in state *idle*. Reception of a trigger leads to sending a search, putting it in state *neighbor lookup*. Timer T2 triggers repetitions of search messages up to the given number of times. Either the reception of a search reply or a timeout of timer T3 puts the system back to state *running*. Timer T1, which is shared for all neighbors of the enhanced router, triggers the periodical sending of pulse messages to the neighbors.

Failures to receive pulse messages from the neighbor are caught by timer T4. It results in putting the neighborship in state *alerted*. A second event of T4

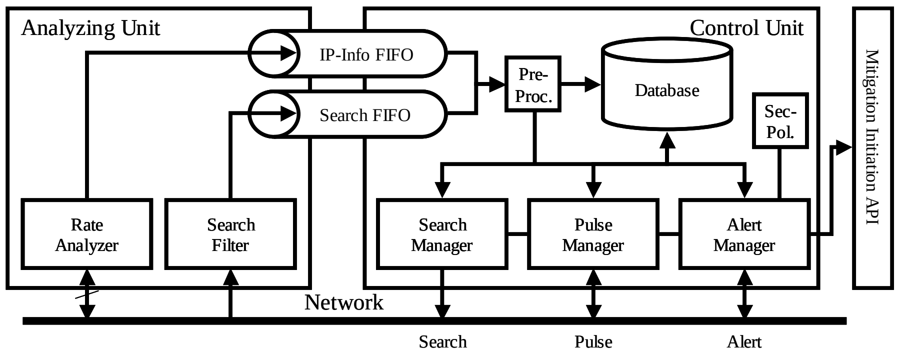


Figure 4: Prototype Architecture (↮ sniffing interface, ↔ capturing interface)

results in sending alert messages to (according to the neighborhood table) upstream neighbors. These alerts are sent as long as there are no pulses received. After a reasonable amount of time — controlled by timer T5 — the neighbor is dropped and alert messages are no longer sent. This allows the network to return to normal packet forwarding. This extension is provided to be able to remove an enhanced router from the framework. Of course, such a router should send a suitable deregistration message.

We assume that a failure to receive pulse messages due to overrunning of the subsequent router will be solved quickly after initiating mitigation. Therefore, further pulse messages will arrive soon. Otherwise, the error is most probably caused by some other event, such as an unexpected reboot or a hardware failure, due to which a further mitigation is not suggestive.

4. Prototype Implementation

The prototype architecture depicted in figure 4 consists of an analyzing and a control unit. The rate analyzer sniffs the network in order to keep track of the amount of data that is delivered to a specific or aggregated destination by extracting destination IP address and packet size from every IP packet. The extracted information is transferred to the control unit via the IP-Info First-In-First-Out (FIFO). In contrast to the rate analyzer, the search filter captures search messages from the network and inserts them into the search FIFO.

Considering the control unit, the database consists of neighborhood table and local destination table. The latter contains information about how much data, addressed to an aggregated destination (ER) respectively to a specific end-host (LMAR), has been routed by this enhanced router within a defined period of time. The preprocessor unit dequeues the IP-info FIFO, prepares the information and inserts it properly into neighborhood table respectively local destination table. Furthermore, it also watches for packets destined to addresses which are not contained in the neighborhood table yet. In case of such an event, it informs the search manager. Finally, it forwards search messages to the pulse manager who is in charge of further processing.

The search manager is responsible for sending search messages and, additionally, keeping track of open search requests in order to avoid repeated sending of identical search packets. Finally, it monitors and regulates, if necessary, the rate of search requests in order to keep traffic overhead under a specified limit.

The pulse manager sends and receives pulse messages. A received pulse message is either a response to a search request or a regular pulse. In case that a search message was received by the pulse manager, it checks with the help of the search manager whether the response matches an open request and if so, it updates the neighborhood table. In case that the received message was a regular pulse message, it adjusts the level of the corresponding entry in the neighborhood table. Besides this, the pulse manager also creates and sends pulse messages, either triggered by an internal timer which elapsed (regular pulse), or triggered by the search manager (search response). A timer is running for each neighbor in order to monitor regular reception of pulse messages. In the event that a timer elapses, the pulse manager triggers the alert manager to launch an alert.

The alert manager exclusively possesses a socket which is used to receive and send alert messages. In case that an alert message is received, it is analyzed and the corresponding countermeasures are initiated by briefing the mitigation initiation API. Besides this,

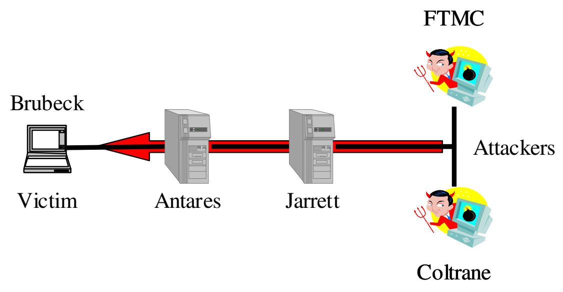


Figure 5: Testbed

an alert manager that is running on an LMAR is also in charge of monitoring the traffic volume that is routed to specific end-hosts. In case that the traffic volume exceeds the threshold that is defined in the security policy, the alert manager launches an alert. It sends alert messages to the next neighbors upstream and it briefs the mitigation initiation API.

5. Proof-Of-Concept

With the described prototype we conducted a proof-of-concept experiment. Therefore, we setup the testbed which is depicted in figure 5 and that consists of two attacking DDoS slaves (*fmtc* and *coltrane*), two enhanced routers (*antares* and *jarrett*) and a victim (*brubeck*).

Jarrett automatically became a neighbor of *antares* with the detection of the first packet that was sent to *brubeck*. *Antares* supervises the traffic volume that is sent to the victim and in addition, we specified a maximum traffic rate for *brubeck* of $1MByte/s$. If this rate is reached *antares* triggers an alarm and sends it to *jarrett*. Finally, both enhanced routers start to drop all packets that are destined to the victim.

Figure 6 depicts the interface-specific traffic volume that is observed by *antares*. The ”bandwidth endhost” curve—which equals the ”bandwidth out” curve in this scenario—shows the amount of traffic that is sent from *antares* to the victim. The ”bandwidth in” curve represents the amount of traffic that arrives at *antares* and which is addressed to *brubeck*. It can be seen that both enhanced routers drop the packets. In detail, *antares* starts earlier dropping as it is the last mile active router which informs the enhanced routers further upstream. The time difference between both curves represents the time that is required for the alert notification and the attack mitigation initiation. We emphasize here that the purpose of the conducted experiment was to demonstrate the operativeness of the concept.

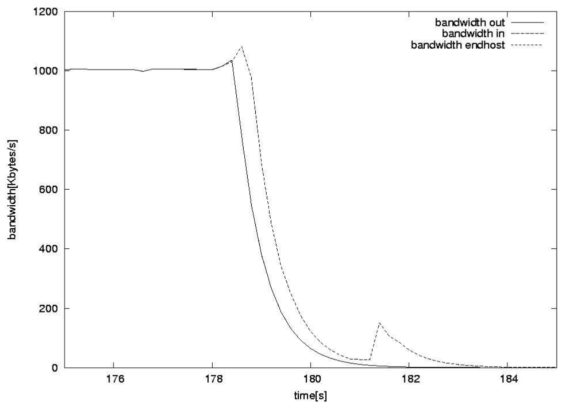


Figure 6: Attack mitigation

6. Conclusions and Outlook

In our paper we have presented an proactive DDoS mitigation framework purposed on the ability to quickly disburden hosts or networks. We have outlined the basic required protocol mechanisms to interconnect enhanced routers within the network without requiring administrative configuration. Furthermore, we demonstrated the operativeness of our approach on the basis of a proof-of-concept experiment.

Future work includes the development of a registration mechanism to allow end hosts to communicate their specific traffic expectations to the nearest enhanced router. The current framework is partially able to handle routing abstractions as given by connection tunneling or QoS routing. Tunneled traffic can only be mitigated in total and mitigation cannot be restricted to certain targets within a tunnel. QoS routing may lead to search messages not following the same path as the usual traffic. As a result, mitigation rules can only be installed near entry and exit points of the QoS routed area.

Furthermore, research is required how detect triggering events in order to match most of the passing traffic while still not overloading routers. This is one of the reasons, why, in our opinion, our framework is not installed on backbone routers but in the less burdened edge area, where it is still able to efficiently reduce DDoS attacks.

Apart from the aforementioned aspects also security issues, as for example a secure authentication mechanism between the enhanced routers, must be considered. Future work also includes a simulative evaluation of our approach.

References

[1] S. Bellovin, M. Leech, and T. Taylor. ICMP Traceback Messages. `http://www.ietf.org/internet-drafts`. Internet Draft, work in progress, October 2001.

[2] J. D. Howard. An Analysis of Security Incidents on the Internet. PhD thesis, Carnegie Mellon University, August 1998.

[3] Computer Security Institute and Federal Bureau of Investigation. 1999 CSI/FBI Computer Crime and Security Survey. Computer Security Institute Publication, March 1999.

[4] John Ioannidis and Steven M. Bellovin. Implementing Pushback: Router-Based Defense Against DDoS Attacks. In *Proceedings of Network and Distributed System Security Symposium, Catamaran Resort Hotel San Diego, California 6-8 February 2002*, 1775 Wiehle Ave., Suite 102, Reston, VA 20190, February 2002. The Internet Society.

[5] J. Moy. OSPF Version 2. Request for Comments 1247, Internet Engineering Task Force, July 1991.

[6] Stefan Savage, David Wetherall, Anna Carlin, and Tom Anderson. Practical Network Support for IP Traceback. In *ACM SIGCOMM*, pages 295–306, October 2000.

[7] D. Sterne, K. Djahandari, R. Balupari, W. L. Cholter, B. Babson, B. Wilson, P. Narasimhan, A. Purtell, D. Schnackenberg, and S. Linden. Active Network Based DDoS Defense. In *DARPA Active Networks Conference and Exposition*, 2002.

[8] Computer Emergency Response Team. CERT Advisory CA-2000-01: Denial-of-Service Developments. `http://www.cert.org/advisories/`, January 2000.

[9] Computer Emergency Response Team. CERT/CC Statistics 1988-2003. `http://www.cert.org/stats`, 2003.

[10] Computer Emergency Response Team. W32/Blaster Worm. `http://www.cert.org`, August 2003.