Oblivious DDoS Mitigation with Locator/ID Separation Protocol

Kazuya Okada, Hiroaki Hazeyama, Youki Kadobayashi
Nara Institute of Science and Technology, Japan
{kazuya-o, hiroa-ha, youki-k}@is.naist.jp

ABSTRACT

The need to keep an attacker oblivious of an attack mitigation effort is a very important component of a defense against denial of services (DoS) and distributed denial of services (DDoS) attacks because it helps to dissuade attackers from changing their attack patterns. Conceptually, DDoS mitigation can be achieved by two components. The first is a decoy server that provides a service function or receives attack traffic as a substitute for a legitimate server. The second is a decoy network that restricts attack traffic to the peripheries of a network, or which reroutes attack traffic to decoy servers. In this paper, we propose the use of a two-stage map table extension Locator/ID Separation Protocol (LISP) to realize a decoy network. We also describe and demonstrate how LISP can be used to implement an oblivious DDoS mitigation mechanism by adding a simple extension on the LISP MapServer. Together with decoy servers, this method can terminate DDoS traffic on the ingress end of an LISP-enabled network. We verified the effectiveness of our proposed mechanism through simulated DDoS attacks on a simple network topology. Our evaluation results indicate that the mechanism could be activated within a few seconds, and that the attack traffic can be terminated without incurring overhead on the MapServer.

Categories and Subject Descriptors

C.2.0 [**General**]: Security and protection; C.2.2 [**Network Protocols**]: Routing protocols

General Terms

Security

Keywords

DoS/DDoS, Mitigation, Routing, LISP

1. INTRODUCTION

CFI '14 Tokyo, Japan

http://dx.doi.org/10.1145/2619287.2619291.

DoS and DDoS attacks can halt the functioning of critical online services. These attacks have the simple objective of wasting network or host resources by inundating servers with inordinate numbers of requests. In the worst case, the service becomes unavailable for legitimate users as the host connection resources are exhausted. At the same time, such attacks consume bandwidth between the attack sources and the target host. Recently, attack scales and their impacts have become insurmountable. In March 2013, there were over 200 Gbps of DDoS attacks against Spamhaus, which is an e-mail blocking list provider [7].

In order to protect hosts and network resources against DDoS attacks, a wide variety of defense methods have been proposed. However, existing blocking methods cannot either 1) reduce network traffic load while keeping services active, or 2) terminate DDoS traffic without being detected. For example, attack traffic can be blocked in a victim's autonomous system (AS) by an access control list (ACL), a firewall, or by an intrusion detection system/intrusion protection system (IDS/IPS) [1]. However, these blocking methods work only on the victim side network, so they cannot eliminate a load shared among various networks. Furthermore, such blocking methods can be detected by observation of packet drops on the attacker's side. In contrast, black hole routing [14, 8] can reduce the load at the ingress point of backbone networks, but legitimate services on the victim server become unavailable when this method is used, and it is also easily detectable.

Therefore, if at all possible, attackers should be kept oblivious to an attack mitigation effort, because they will change attack methods or sources if the attacker nodes determine that their attack is being thwarted. Additionally, the defense methods should be implemented as close to the attacker nodes as possible in order to reduce traffic loads.

To tackle this challenge, we developed a new DDoS mitigation method that can reduce loads on victim servers and networks while protecting ongoing legitimate services. In this paper, we propose a DDoS mitigation method that utilizes a two-stage map table extension of LISP [4, 6] in a way that keeps attackers oblivious to the defense efforts. While LISP was originally designed with a single map stage for IP based routing, we built an additional map table for the LISP architecture, which is called a mitigation table. This table is used to forward attack traffic to a decoy server that has the same IP address as the legitimate server. From the attacker's point of view, therefore, it is extremely difficult to distinguish whether the destination is legitimate or a decoy. Additionally, since simply adding a new mapping entry to

the mitigation table can trigger the mitigation, this solution does not require the reconfiguration of network devices or legitimate servers. We implemented the two-stage extension on MapServer, with hundreds of additional codes.

The rest of this paper is organized as follows: Section 2 discusses works related to DoS and DDoS attack defense. Section 3 describes our proposed method and its implementation. Section 4 shows the results of our experimental approach. Section 5 discusses characteristics of our proposed method. Section 6 concludes this paper.

2. RELATED WORKS

Before describing our proposed method, we will discuss the difficulties of DDoS defense and existing defense methods. Existing DDoS defenses are categorized into the following phases: prevention, detection, identification, and routing-based mitigation. There are a variety of solutions used in each of these individual phases.

Next, we will discuss difficulty of defending against DDoS attacks. In DDoS attack traffic, each packet has a different source IP address, that is mechanically generated by the attack programs. Therefore, victim hosts or security devices have difficulty identifying the attack packets from the amount of traffic. Additionally, there are often DDoS-like short term access trends on current services, known as *flash crowds* that produce traffic patterns that are similar to a DDoS attack when vast numbers of unique users attempt to access a service. However, such traffic is not offensive and must not be blocked. Even if a security device unequivocally detects and identifies the sources of an attack, the IP addresses are dynamically changed. Additionally, according to [1], 77% of all DDoS attacks are less than one hour in duration, which means that the defense has a limited time to adapt to the changes.

This prevention method aims at blocking DDoS traffic in a victim network. Ingress/Egress Filtering [5, 3] involves blocking packets based on an ACL maintained at a router or a switch. Ingress filtering blocks attack packets that attempt to enter the network by filtering lists on an L2 switch or a router. In contrast, egress filtering blocks outgoing packets from the network. Such filtering operations have been accepted as a best current practice (BCP) and are in widespread use on commercial networks. However, this type of blocking cannot reduce the attack traffic volume between the attacker's networks and a victim's network.

In general, DDoS attacks are detected by IDS or IPS appliances in commercial networks [12]. However, IDS/IPS appliances are prone to state table problems when detecting such attacks. In fact, a recent report [1], states that DDoS attack detection often fails due to state table depletion. There are also a number of dedicated products aimed at DDoS detection that utilize flow analysis [9], and numerous researchers have attempted to resolve the problem of anomaly-based attack detection. However, all of these detection algorithms still produce a numbers of false positives.

The identification of attack sources or attack paths is important when narrowing down the range of mitigation method drawbacks. If attack source IP addresses are unequivocally identified from attack traffic, those addresses can be reported to network operators who can then use egress filtering to block the traffic at a border or a periphery router where the attack sources are connected to the Internet. Additionally, source identification technologies provide

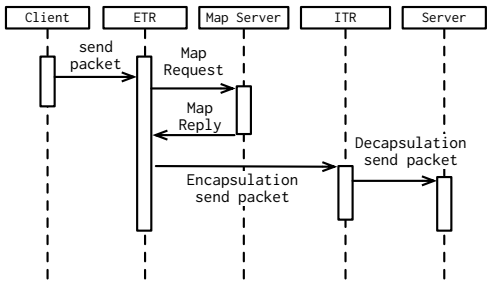

Figure 1: Packet forwarding sequence among LISP routers and the MapServer

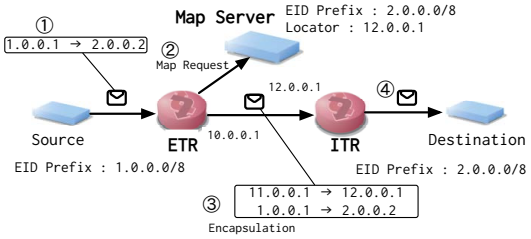

Figure 2: Packet forwarding on LISP networks

a strong deterrence force against attackers. Hash-based IP traceback [13] enables source identification by storing packet information in a router. In contrast, packet marking [11] works to incrementally encode the path information of the packet itself, which means the router does not need to retain that information. As a practical traceback method, NTT communications developed and operates a flow analysis system against DDoS attacks named SAMURAI [10], which is capable of tracking IP packets back to their ingress routers on a backbone network.

Blackhole routing [8,14] is a routing-based mitigation method that forwards malicious traffic to a null router device by dedicated routing information. However, this approach has the potential to disrupt legitimate traffic.

Another potential solution is supplied by Arbor Networks, which provides Peakflow [2] to protect a network against DDoS attack. This product monitors network traffic and injects new IP routes that divert malicious traffic into a filtering device when an attack is detected. Such products work on enterprise and ISP networks, but they only function on the victim's network itself, and cannot eliminate malicious traffic between an attacker source and a victim network.

3. OVERVIEW OF LISP-BASED MITIGATION

In this section, we describe our proposed LISP-based DDoS attack mitigation method, which is based on a simple extension of the LISP map table. Ideally, DDoS attack countermeasures should work near the attacker nodes in order to reduce unnecessary traffic loads on the network and to keep services available for legitimate clients. At the same time, attackers should be kept oblivious to the mitigation effort in order to deter them from changing attack methods or sources. However, current DDoS attack countermeasures work on the victim's network, and thus often block

services for legitimate clients. We resolve these problems using a LISP map-table extension. In the next section, we will briefly discuss LISP specifications, after which we will outline our proposed mitigation method.

3.1 Overview of LISP

LISP is a new Internet routing architecture with specifications standardized as RFC [4,6] by the Internet Engineering Task Force (IETF). With the traditional IP architecture, IP addresses work as both the device and network identifier.

Contrastingly, in LISP, the address role is separated into an end point identifier (EID) and a locator on the network. The EID is used to uniquely identify a device on the network. A routing locator (RLOC) is a routing point address for EIDs on the Internet.

EID and RLOC expressions can be used in the existing IPv4 and IPv6 address formats. In the separation architecture, network deployments can be made scalable because a device's EID addresses are aggregated by a few RLOCs. In addition, because separation architecture enables device mobility on the network, we can move an EID network location by changing the RLOC without the need for any configuration modifications to the device.

A packet forwarding sequence on LISP is displayed in Figure 1, 2. In these cases, a client attempts to communicate with a server via the LISP infrastructure by sending a packet to the server. This packet is then forwarded to the client side LISP router, known as an egress tunnel router (ETR). When the ETR is receiving a packet from inside the network, it sends a MAP request to a MapServer in order to determine the next hop/gateway router for a destination EID. The MapServer retains the binding information between the RLOCs and EIDs. The MapServer then replies with the RLOC of a destination EID to the ETR. After receiving the RLOC, the ETR encapsulates the packet and forwards it to the router that has the RLOC. An RLOC router, which is called an ingress tunnel router (ITR), then decapsulates the packet and forwards it to the chosen server located in the network.

3.2 Two-stage Map Table Extension

Existing DDoS defense methods have the following three disadvantages. The first is defense location. Almost all defense methods block attacks in the vicinity of the victim hosts. In methods of this type, host loads such as network bandwidth or CPU usage can be reduced, but the methods cannot eliminate the network traffic loads between the attackers and the victim. The second disadvantage is that several mitigation methods adversely impact legitimate services by filtering or dropping packets.

The last disadvantage is lack of obliviousness against attackers, who can easily recognize the mitigation effort from parameter changes measured by the attacker nodes. For example, firewall or ACL blocking can be recognized by connections from multiple nodes located in other networks.

The two-stage map table extension overcomes such drawbacks of existing mitigation methods. In our proposal, we assume the attacker's location or the ingress routers of the attack traffic have been detected by IP traceback or some other method. Based on that preposition, our system leads the attack traffic to one or more decoy servers, which behave as victim servers. Since the decoy server is located close to the attacker's network, the attacker's traffic does not leak

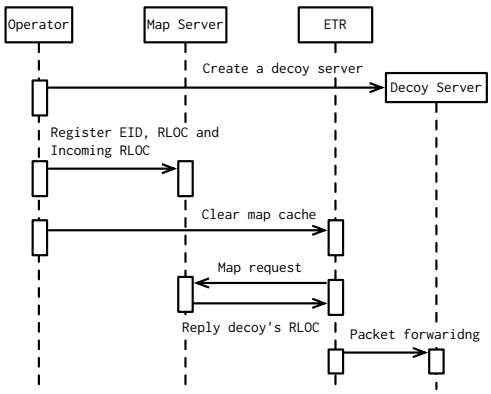

Figure 3: Mitigation sequence on two-stage map table

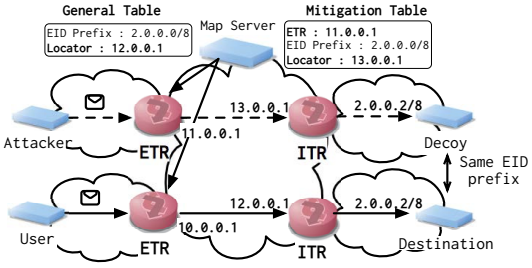

Figure 4: Packet forwarding on two-stage map table

to outside networks. Additionally, since the decoy servers have the same EID (IP address) as the targeted legitimate server, attackers are unable to distinguish between them.

In order to divert attack traffic to decoy servers located in network borders, it was first necessary to extend the LISP mapping system. To accomplish this, we begin by assuming that the original and additional map tables are manually configured by network operators. In this paper, we will refer to the additional map table as the "mitigation table", while the original map table will be called the "general LISP map table". The sequence diagrams are shown in Figure 3, 4. The mitigation sequence with the map table proceeds as follows. When a network operator detects a DDoS attack on the network, he or she registers a new map entry to the mitigation table. The entry has three fields: a border router's address (incoming router), a destination address (EID prefix) and a router address that hosts a decoy server (locator). After the map entry registration, the operator frees up a map cache for the incoming router or the ETR. Next, the router needs to identify the address of attack packets via the MapServer. If a packet is coming from the attacker, the router obtains an RLOC that hosts a decoy server from the MapServer. Finally, the packet is forwarded to the decoy server. If a packet is not sent from the attacker, the MapServer sends a legitimate RLOC to the ETR. Accordingly, only attack packets are routed to the decoy server. We call this two-map-tables-based system a "two-stage MapServer"

The mitigation entry remains valid in a timer in much the same way as a domain system name (DNS) time to live (TTL) hop limit. When the timer expires, the ETR transmits requests for solving the destination IP addresses of the

DDoS packets to a MapServer. After that process, the DDoS packets will again be forwarded to the decoy network.

3.3 Mitigation Table

A LISP MapServer entry is constructed using the following four fields: **EID Prefix** is expressed by an IPv4 or IPv6 address. **Priority** is used for a RLOCs section. If multiple RLOCs bind to an EID prefix, a lower value RLOC is preferred. **Weight** value use to traffic load balance between RLOCs. **Locator (RLOC)** is an IPv4 or an IPv6 address assigned to an ETR.

In a mitigation table, we add an incoming router's IP address field. This is the source address of the router that originates the DDoS attack traffic.

Algorithm 1 shows the RLOC selection pseudo code. When a MapServer receives a map request from an xTR (Ingress/Egress Tunneling Router), the server searches the mitigation table. If the xTR matches a table entry, the server responds with a decoy router's RLOC. If there is no matching entry, the server expands the search to the general LISP map table.

```
if Search mitigation table (incoming router address)
then
  | Respond with a decoy router's RLOC
else
  | Search general table (incoming router address)
  | Respond with a legitimate router's RLOC
end
```

Algorithm 1: RLOC selection via two-stage map tables

3.4 Decoy Network and Server

The minimum set of a decoy network consists of a decoy router and a decoy server. The decoy router does not require special features since it works the same as a general LISP router. Ideally, the decoy server and network should operate identically to the legitimate server. This means the server should have the same IP address, contents, and responses. Therefore, the server needs to be kept synchronized with the configuration and contents with the legitimate server. It is believed that server side virtualization techniques could be adapted making a decoy server.

3.5 Advantages

In this subsection, we will discuss the advantages of our method. To begin with, our method can reduce the traffic loads on both the legitimate server and the network by forwarding attack traffic to decoy servers located on the periphery of a LISP-enabled network. Additionally, our two-map LISP extension does not require any configuration changes on a victim host because our extension can control attack traffic using only an EID and a RLOC binding on an extended MapServer. Furthermore, it is not necessary to change or add configurations and routes to routers. In contrast, other routing-based mitigation techniques, such as black hole routing, require network operators to modify configurations and install routes of the routers. Therefore, the mitigation can be applied by network service providers. Moreover, since the decoy server has the same IP address as the legitimate server, attackers face extreme difficulty in recognizing whether or not their target is defended.

4. EXPERIMENT

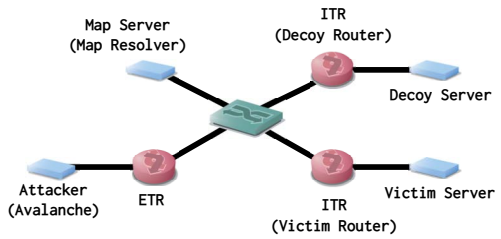

Figure 5: The experiment topology

In this section, we evaluate the performance of our software implementation. The purpose of this experiment is to evaluate throughput and the time required to switch routes using our extension.

4.1 Methodology

Figure 5 shows the network topology used in our experiment, which consisted of the attacker's network, the victim's network and the decoy network. We implemented the proposed method by modifying a LISP MapServer implementation on Linux [15]. The software has a LISP router and a map table module written in C code. We then added a mitigation table and its related functions in the map table module. The module code consists of a mere 350 lines of code. In total, the MapServer source has 2126 lines. In this experiment, each server and router was equipped with a 12 GHz Intel Xeon X2270 M2 central processing unit (CPU), 12 GB of memory and a 1000Base-T Ethernet port. The MapServer is running on the Linux server (Debian Squeeze). Each LISP router used [15]'s LISP implementation.

During our experiment, we measured the bandwidth of the victim and decoy server. To accomplish this, we collected received data volume from */proc/net/dev* per second in the servers. After starting the traffic generation on attacker node, we manually freed the map-cache on the ETR and the route of the traffic using our proposed method.

4.2 Basic Throughput

First, we tested basic performance using *iperf*, which is a common throughput testing method on Linux. A single *iperf* source can send user datagram protocol (UDP) packets to the victim server at a rate of 900 Mbps. In this experiment, we manually changed the map entry at approximately 100 [sec]. Figure 6 shows a result of the experiment. After a few seconds, the traffic was moved directly to the decoy server. This result shows that our implementation can quickly change routes despite high traffic rates.

4.3 UDP Traffic Mitigation

We then evaluated the performance of our method against a realistic UDP DDoS attack. The real DDoS traffic has spoofed massive source. For the attacker role, we used Avalanche 290, which is a commercial test traffic generator. Each link has a bandwidth of 1 Gbps. During the experiment, we placed a load on the victim host using the traffic generator. To simulate an UDP-based DDoS attack, we sent DNS A query packets to the victim server from the generator. We were able to generate approximately 700 to 900 Mbps of traffic using the generator, along with 0.18 million pps and about 105 million unique IP address sources.

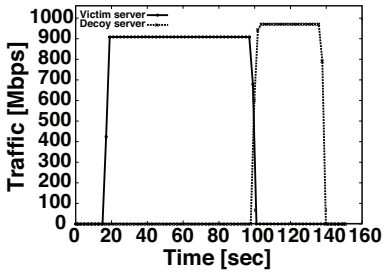

Figure 6: Traffic volume on victim and decoy server (iperf UDP)

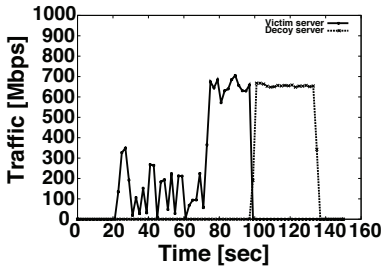

Figure 7: Traffic volume on victim and decoy server (DNS)

Figure 7 shows the received traffic volume on the victim and decoy servers. At approximately 100 seconds on the x-axis, the traffic was smoothly forwarded to the decoy server. The total transition time was just a few seconds long, even though the traffic had massively different source addresses.

4.4 TCP Traffic Mitigation

We measured how quickly our method could mitigate a TCP based-attack by conducting the following experiment. Here, we generated massive *HTTP Get Request* queries for the victim server where virtual tester clients downloaded 1 GB of data per connection. We then measured traffic bandwidth on the victim server and the decoy server using the same measurement methodology adopted during the UDP experiment.

Figure 8 shows the forwarding volume to the victim and the decoy on the server. In the figure, each line has a zigzag shape caused by TCP congestion between clients and the server. After 100 [sec] the traffic was smoothly mitigated to the decoy server. Even though the destination changed, the attacker's TCP connections were smoothly established between the simulated clients on the traffic generator and the server. This indicates that an attacker would be unlikely to recognize the mitigation provided by the decoy server.

4.5 Experimental Summary

The abovementioned experimental results demonstrate that our LISP two-stage map extension has the potential to mitigate wire speed traffic. Since the mitigation only takes a few seconds, both TCP and UDP DDoS attacks would be quickly mitigated. Additionally, attackers would find it difficult to recognize the mitigation was in process because the server responses do not change when the mitigation takes affect. These are ideal features for keeping the mitigation undetected by the attackers.

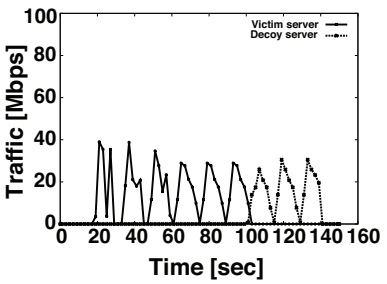

Figure 8: Traffic volume on victim and decoy server (HTTP GET)

5. DISCUSSION

Next, we will discuss considerations related to our LISP-based mitigation method.

5.1 Scalability

In our approach, there are scalability issues between a MapServer and a map cache. While the MapServer has to hold mitigation entries, which are registered by operators, the entry size depends solely on the unique incoming router. Accordingly, there is no special memory requirement for the MapServer and xTRs because attack filtering is based on the incoming router. Furthermore, since the number of incoming routers is much smaller than the unique source addresses of a DDoS attack, the defense actually has high scalability in relation to the number of attack sources on the Internet.

5.2 Installing Mitigation Entries

For our proposed architecture, the ETR's map cache must be free, or it will be necessary to await TTL expiration of an entry, after the mitigated entry. This means that if a network operator hosting a victim node in their network cannot explicitly clear a map cache on an attacker's router, attack mitigation will be delayed. Setting a short TTL for a map entry has the potential to prevent this delay, but it makes traffic for map resolves, and could potentially waste router and MapServer resources including CPU, memory, and bandwidth between an ITR and MapServers. Accordingly, it is necessary to consider a threat information sharing system among network operators. In such a system, when an operator receives threat information that impacts their network, they manually or automatically clear the map entry. This operation makes it possible to reduce response times when defending against an attack.

This mitigation operation function works by simply controlling map entries on the MapServer. In contrast, existing routing-based mitigation such as blackhole routing must inject new routes with accompanying parameters into the routers. When this occurs, those routes should be avoided in order to prevent legitimate traffic from being forwarded to the black hole interface. After the attack, the routes must be manually eliminated from the routers.

5.3 Attack Localization

In our two-stage map extension scheme, attack traffic affects the attacker and decoy networks, and the traffic is localized in these networks. Therefore, if we can set up a decoy network near an attacker's network, the attack traffic will not affect the Internet backbone. Accordingly, the

decoy network location is an important factor of attack localization. An ideal situation would be for all border routers to have LISP capability and a hosting environment suitable for a decoy network. This would allow any attack traffic to be forwarded to the decoy network on an AS border router before going out to the Internet.

5.4 Obliviousness

By using two-stage MapServer, we lead malicious traffic to a decoy server. The decoy server has the same IP address as a legitimate server and works in the same way as the legitimate server against the attacker. Therefore, defensive activities cannot be observed by the attacker based on server behavior alone. As mentioned in section 3.4, a decoy server could be made from a legitimate server using by virtualization techniques. However, even if a copy of the server instance is successfully made, it is still possible for attackers to detect the circumvention by measuring application level behaviors. For example, the attacker could be checking query responses. Accordingly, it is still necessary to consider ways to make more realistic copies of legitimate servers.

Additionally, the routing path should follow along the same route from the attacker node to the legitimate server even if the attack is forwarded to the decoy server. If the path deviates from the original routing path, the attacker might see through the defense mechanism by comparing the current routing path with original path based on traceroute results. Currently, however, many ISP network devices do not reply to Internet Control Message Protocol (ICMP) request packets, so even if the decoy network or server does not reply to the packet request, it would not be suspicious in itself. Furthermore, even if replies are required, imitative replies could be generated by simple codes.

6. CONCLUSION

Taking into consideration the need for an unnoticed mitigation method against DDoS attacks, we proposed a simple extension of the LISP map table. This extension separates the LISP map table into general and mitigation tables. While legitimate traffic is transferred to the legitimate servers along with the general table, attack traffic is routed to decoy servers by edge LISP routers that are activated based on the mitigation table. Because of the Locator and ID separation characteristics, decoy servers have same IP addresses as the legitimate servers. These characteristics contribute to complicating the attacker's chore of determining whether the servers are legitimate or decoys.

To accomplish this, we modified an existing MapServer implementation and evaluated its preliminary performance. The evaluation results in a simple network topology show that the proposed mitigation can be activated within a few seconds, regardless of the type of traffic. Additionally, the decoy server treated attack traffic the same way, and provided the same throughput, as the legitimate server. Consequently, it has been shown that the proposed approach can achieve oblivious DDoS mitigation.

Acknowledgment

This research was supported by the Strategic International Collaborative R&D Promotion Project of the Ministry of Internal Affairs and Communication, Japan, and by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 608533 (NECOMA). The opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the Ministry of Internal Affairs and Communications, Japan, or of the European Commission.

7. REFERENCES

[1] D. Anstee, D. Bussiere, and G. Sockrider. Worldwide Infrastructure Security Report 2012 Volume VIII. Arbor Special Report, Jan. 2013.

[2] Arbor Networks. Peakflow. `http://www.arbornetworks.com/products/peakflow`.

[3] F. Baker and P. Savola. Ingress Filtering for Multihomed Networks. RFC 3704 (Best Current Practice), Mar. 2004.

[4] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis. The Locator/ID Separation Protocol (LISP). RFC 6830 (Experimental), Jan. 2013.

[5] P. Ferguson and D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC 2827 (Best Current Practice), May. 2000. Updated by RFC 3704.

[6] V. Fuller and D. Farinacci. Locator/ID Separation Protocol (LISP) Map-Server Interface. RFC 6833 (Experimental), Jan. 2013.

[7] Q. Jenkins. Answers about recent ddos attack on spamhaus. `http://www.spamhaus.org/news/article/695/answers-about-recent-ddos-attack-on-spamhaus`, Mar. 2013.

[8] W. Kumari and D. McPherson. Remote Triggered Black Hole Filtering with Unicast Reverse Path Forwarding (uRPF). RFC 5635 (Informational), Aug. 2009.

[9] A. Lakhina, M. Crovella, and C. Diot. Mining anomalies using traffic feature distributions. In *Proc*, SIGCOMM '05, pages 217–228. ACM, 2005.

[10] J. Murayama, A. Kobayashi, H. Kurakami, T. Kuwahara, K. Ishibashi, and N. Miyake. Traffic monitoring and analysis technologies. *NTT Technical Review*, 8(7), Jul. 2010.

[11] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical Network Support for IP traceback. In *Proc*, SIGCOMM'00, pages 295–306, Aug. 2000.

[12] K. Scarfone and P. Mell. Guide to Intrusion Detection and Prevention Systems (IDPS). *NIST SP800-94*, 2007.

[13] A. C. Snoeren, C. Partridge, L. A. Sanches, C. EJones, F. Tchakountio, S. T. Kent, and W. T. Stayer. Hash-based IP traceback. In *Proc*, SIGCOMM'01, pages 3–14, Aug. 2001.

[14] D. Turk. Configuring BGP to Block Denial-of-Service Attacks. RFC 3882 (Informational), Sep. 2004.

[15] Y. Ueno, K. Horiba, and K. Kataoka. Design and Implementation of Software LISP Router. *Internet Conference 2011 (IC2011) - Work in Progress*, 2011.