Journal of Network and Information Security
6 (1), June 2018, 01-11
http://www.publishingindia.com/jnis

Layered Migrating Overlay for Effectively Sieving Internal DoS/DDoS Attackers - Its Designs and Effectiveness

Hiroshi Fujinoki
Department of Computer Science, School of Engineering, Southern Illinois University Edwardsville, Edwardsville, Illinois, United States of America. Email: hfujino@siue.edu

Abstract: Several overlay-based solutions have been proposed to protect network servers from DoS/DDoS attacks. The common objective in the existing solutions is to prevent the attacking traffic from reaching the servers by hiding the location of target server computers. The recent evolutions in DDoS attacks, especially in the increase in the number of bots involved in a DDoS attack and in the degree of control such bots have to the hijacked host computers, pause serious threats to the overlay-based solutions. We designed and assessed the potential of new overlay-based security architecture that addresses the recent evolutions in DDoS attacks. The new security architecture, called "Layered Migrating Overlay (LMO)", is designed to protect cloud servers (a) when their legitimate users convert to DoS/DDoS attackers or (b) when DDoS attacks are launched from the legitimate users' host computers that are hijacked by DDoS coordinators. LMO copes with the situations by sieving attacking traffic from the hijacked legitimate users' host computers using dynamic binary user splits over the migrating entry points to an overlay network. Our discrete event driven simulation suggested that LMO will efficiently sieve DDoS attacking hosts in many different situations, when a small number of attacking hosts hide behind a large legitimate user group, or when a stampede of DDoS attacking hosts occupy the majority of incoming traffic, without requiring a large number of migrating entry points. We also found that how quickly each migrating entry point can detect excess traffic is a key to keep convergence delay short.

Keywords: Network management, Overlay networks, Security management, Denial of services, Insider threats.

I. Introduction

The primary reason that makes denial of service (DoS) and its distributed counterpart, distributed DoS (DDoS), hard to prevent is that the attacking traffic is hard to be distinguished from legitimate one. We focus on DDoS, instead of DoS, mainly due to the higher difficulty in preventing DDoS [1]. To deal with the challenge, many solutions have been proposed [2]. Major existing solutions are egress filtering, ingress filtering, router push-back, reducing server-side bottleneck, and resource hiding using overlay networks. Despite the effort, DDoS attacks are still on the rampage in the Internet today. The major weaknesses in each approach are summarized below:

Egress filtering [3]: The gateway router in a network domain forwards outgoing packets only if they are destined to permitted destinations. This approach provides limited protection, if attacking traffic is generated to the permitted destinations from the host computers that are hijacked by DDoS coordinators.

Ingress filtering [4]: The gateway router in each network domain makes sure that the source IP address in every outgoing packet is valid (i.e., it matches with the domain address). Any packet that carries a spoofed source domain address will be dropped before they are injected to the public Internet. Similar to the problem in egress filtering, if the hijacked host computers generate attacking traffic using their valid network addresses, the effectiveness in this approach is limited.

Router push-backing [5]: After the gateway router or a core router on an attacking path detects an ongoing DDoS attack, it recursively contacts the next core router in its upstream of the attacking path, requesting the routers to drop or rate limit the attacking traffic. This approach requires every core router on an attacking path to coordinate, implying that a standard protocol must be developed and that the core routers must support such a protocol. This has not happened yet because of multiple different reasons, such as the development cost (in financially and timely) and even politics in the industry.

Reducing performance bottleneck: Many DDoS attacks target the server-side activities that require high overhead to effectively deplete the processing and storage resource at a victim server. Reducing such overhead at the victim server will mitigate the impact of DDoS attacks. For example, SYN-cookie [6] was proposed to reduce the processing overhead in TCP three-way handshaking. The primary weakness in this approach is that it reduces the overhead, but does not eliminate it. When the volume of attacks can be easily inflated by a DDoS coordinator, such as more than 10,000 coordinated bots [7],

"reduction" may not be enough for preventing the attacks. This is a serious problem when network speed is improving faster than the performance of end host computers.

Resource hiding by overlay networks: This approach prevents DDoS attacks by hiding the ways legitimate users reach servers from attackers using an overlay network. Such overlay networks hide their internal topology from any user, including legitimate ones, by performing secret routing over hidden overlay nodes, as well as by hiding the location of the destination servers. To accept only the legitimate users to an overlay network, each overlay node performs user authentication and/or traffic rate control. This approach prevents the attackers from depleting the network bandwidth to reach the servers as well as processing resource at the victim server. Shi proposed a solution, OverDoSe [8], which uses "puzzles" to limit the rate of new connections from users at entry points to an overlay network. Connection rate for each user is controlled by adjusting the complexity of puzzles. Other existing solutions in this category are described in Section 2.

To overcome the weaknesses in the existing solutions, we propose new overlay-based security architecture. The proposed architecture protects network servers for both the network bandwidth to reach servers and the server side resources, such as processor cycles and memory capacity. Logic bombs (those that disable servers by exploiting known/unknown bugs in server processes and the network protocols) are not a scope of this work. The new security architecture is designed for protecting network services that require each user to register in advance.

The new security architecture is designed especially to cope with the situations that have not been considered at least well enough. DDoS attacks can be launched by legitimate users' host computers that are hijacked by DDoS coordinators. In the situation, DDoS code planted in the legitimate users' host computers can generate attacking traffic to a target server after the users establish a connection to a server through an overlay network, which ruins the protection by an overlay network. Similarly, legitimate users converted to a DoS attacker can cause the same situation.

The rest of this paper is organized as follow. Section 2 describes the major existing overlay-based solutions. The approaches used in the existing overlay-based solutions are described and their strengths and weaknesses are analyzed. Section 3 describes the proposed new security architecture, called Layered Migration Overlay (LMO). Section 4 presents the performance evaluation on the potential of LMO in terms of the convergence delay and the number of overlay nodes needed to identify attacking users. Section 5 summarizes the conclusions and the future work, followed by the selected references.

II. Existing Overlay-Based Solutions

In hiding servers from DDoS attackers, many of the existing overlay-based solutions "migrate" the servers to prevent attackers from reaching the target servers [9, 10, 11, 12]. These solutions can be classified to two groups. The first group physically migrates a server when a DDoS attack is detected at the server [9, 12]. When an attack is detected, the server process is physically migrated to another host computer. The second group logically migrates the server [10, 11]. The latter does not migrate the server processes to another host computer. Instead, they reassign a new network address to the server host computers and the new network address is announced only to the legitimate users.

Each of the above two strategies has their own strengths and weaknesses. The first group (physical migrations) entails high cost, high runtime overhead, and increase in security risks. The high cost stems from the fact that physical migrations necessitate multiple host computers. Most of the commercial network host computers for servers are costly, since they require a high standard for response time, throughput, data capacity, and reliability. For the servers that deal with large volume of data, the overhead to migrate such data will cause significant overhead. Additional runtime overhead will be caused by maintaining data consistency. High volume of data migration to many host computers will increase attack surface. Its primary advantage is robustness. Since each server process is physically migrated to another host, attacks to the access path to the new server location will be effectively prevented as long as attackers do not easily track the location of the new server locations.

The logical migration approach has the opposite characteristics. Because it is only the network address or the server identifier that is changed, the attacking traffic may be able to effectively deplete the transmission and router bandwidth to reach the server. Since a server is never migrated to another host computer, this approach does not require additional major hardware resource or network transmission bandwidth to maintain data consistency.

To solve the trade-off problem between the physical and logical migrations of servers, some solutions use overlay networks and they migrate the overlay nodes instead of servers [13, 14, 15, 16, 17, 18]. Possible design options for overlay-based solutions are discussed and analyzed in [19]. Pingali's solution uses an overlay network, in which the overlay nodes, called "safe houses", migrate to hide the entry points to an overlay network [13]. It is assumed that servers can be accessed only through safe houses. Safe houses are dynamically migrated and only legitimate users should be able to discover them. A weakness in this solution is that there is not much protection if some of the legitimate users convert to attackers, or if legitimate users' host computers are already hijacked by DDoS coordinators. It can be that the attackers may share the information they stole from legitimate users to coordinate a sufficiently large number of hijacked hosts to effectively disable an overlay network or a server.

Wang proposed an overlay-based solution using multipath routing to mitigate the impact from DDoS attacks that can disable

one or a few overlay nodes. It spreads payload transmissions from each legitimate user through multiple overlay nodes in a similar way spread spectrum wireless signal transmission works [14]. One possible weakness is that this approach requires a large memory buffer in the server since "spread spectrum" is performed using packets. The overhead to reorder the arriving packets at a server may not be negligible especially for servers that use a high speed uplink. Since the list of overlay nodes is advertised to each user in advance, attackers will be able to obtain the list if the host computers owned by legitimate users have been hijacked by attackers. If it is the case, attackers may disable a sufficiently large portion of the overlay nodes, which will effectively disable an overlay.

Kurian proposed another overlay-based solution "FONet" [15]. FONet constructs secret reverse path forwarding channels that carry network traffic from overlay nodes (called "Access FONet") to a server. Similar to other overlay-based solutions, its main idea is to hide overlay nodes and their paths to reach the server. FONet uses a class-D IP address to build secret reverse paths to a server and it requires an overlay node at each network domain where there is a user to the server.

Khattab's solution uses a honeypot at each node in an overlay network where each honeypot migrates its location in such a way that attackers cannot predict the way the overlay nodes are migrated [16, 17]. In [16], honeypots are migrated to the upstream of an attacking path to shut down the attacking traffic as early as possible. Although this solution will be logically effective, migrating honeypots to the upstream of an attack path may not be always possible, especially if attacking traffic spans several different network careers because of the same reasons for "router push-back" approach.

The core of WebSoS [18], proposed by Cook, is hiding the location of the destination servers by shuffling entry points to an overlay network. WebSoS is developed as an extension to Chord [20], which is a hash-based routing service for distributed firewalls. By hiding the location of the target server, WebSoS prohibits any end users, including the legitimate ones, from directly forwarding network traffic to a server, which will effectively prevent DDoS attacks. WebSoS assumes that each legitimate user possesses a secret key that authenticates the identity of the user and authorizes the access right. This leads to a few implications regarding WebSoS's robustness in terms of recent growing security concerns of malicious insiders and the host computers hijacked by DDoS coordinators in similar ways to Starvou's solution. For example, if a sufficiently large number of legitimate users convert to attackers, or if a sufficiently large number of the legitimate users' host computers are hijacked by a DDoS coordinator, attacks can be effectively launched through a secure Chord overlay to a target server.

As the conclusion of this section, many of the existing overlay-based solutions [13, 14, 15, 18, 20] are not equipped with a protection against legitimate users who convert to attackers or legitimate users whose host computers are hijacked by DDoS coordinators. The essential security information owned by legitimate users, the one necessary or even enough to get through a secure overlay network to reach a server, can be stolen from the legitimate users' host computers possibly by infection of malwares. Once such incidents happen, the existing solutions have limited effect in protecting servers from DDoS attacks.

A few solutions monitor traffic rate of each user at each overlay node, such as [8]. They distinguish attackers by monitoring network traffic from each user at overlay nodes. However, such per-flow traffic monitoring entails considerable overhead [21], making itself a target of DDoS attacks. Back-pressure approaches using overlay nodes, such as [16], cannot always move overlay nodes to the direction of attackers because of the lack of universal support by the major carriers in the Internet. Because of these problems, a new solution is needed, which efficiently eliminates attacking traffic without monitoring each user's traffic rate or without depending on a back-pressure approach, especially when legitimate users' host computers can be hijacked by DDoS coordinators.

III. The New Security Architecture

The proposed new security architecture is designed to achieve the following two goals: a solution that efficiently eliminates attacking traffic (1) without monitoring each user's traffic rate (2) when legitimate users convert to attackers or when legitimate users' host computers are hijacked by DDoS coordinators to launch attacks to remote servers. This section describes the proposed architecture designs.

A. Architecture Overview

Fig. 1 shows the organization of the proposed security architecture, called Layered Migrating Overlay (LMO). In designing LMO, we assumed a cloud server, although it can be any network server that serves network applications with user authentication. LMO consists of a client-side user process, two different levels of proxy processes (user proxy and gateway proxy), gateway routers, the cloud server, the proxy/user manager, and the user information distributers. Brief descriptions of each component are provided below.

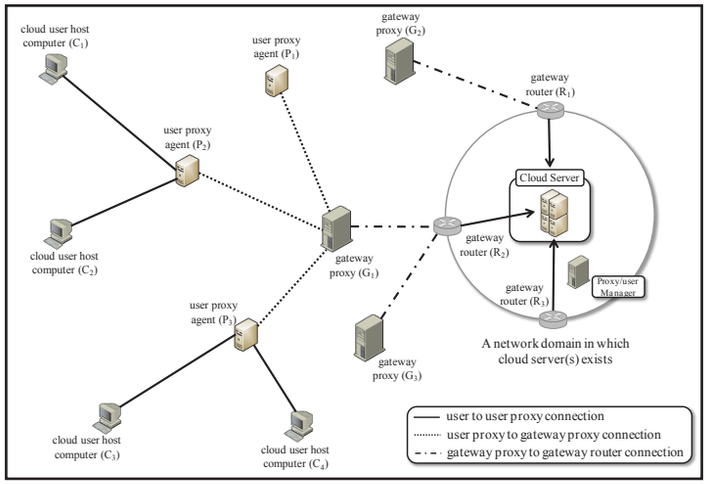

Fig. 1: Organization of the Proposed Security Architecture

Cloud Server: It is a server that provides SaaS, PaaS, or IaaS cloud service to the users who contract with the cloud provider in advance. Each cloud server can be a cluster of server host computers, or a single host computer that runs VM (virtual machine) stack.

Gateway Routers: It is assumed that the network domain of a cloud provider is connected to the Internet through more than one gateway router. Any network traffic to the cloud server must go through one of the gateway routers. Each gateway router works as a proxy for the cloud server and the proxy/user manager. Gateway routers hide the network address of the cloud server and the proxy/user manager by performing network address translation (NAT). The network address of the cloud server or the proxy/user manager is never published to users.

Gateway Proxies: Each Gateway Proxy (GP) is a software process running at a host computer in the public Internet (not in the domain of cloud providers). It is only the GPs that know the network address of the gateway routers. None of the GPs knows the network address of the cloud server or the proxy/user manager. GPs are "gateways" to a destination cloud server from multiple user proxies (the term, "user proxy" is described in the next paragraph). Each GP monitors aggregated traffic rate from user proxies to the cloud server (instead of per-flow rate from each user proxy). GPs issue a warning message to the proxy/user manager through a gateway router if excess traffic to the cloud server is detected.

User Proxy Agents: Each User Proxy (UP) is a software process running at a host computer and it works in a similar way as GP. UPs connect multiple end users to a GP. Each UP knows only the network address of GPs but none of the proxy/user manager, the cloud server, or the gateway routers. To hide the network address of GPs from users, each UP performs NAT between users and a GP. It monitors aggregate traffic rate to the server. When the traffic rate exceeds a threshold, it issues a warning message to the proxy/user manager through a GP.

Cloud Users: They are the host computers used by human cloud users. It is assumed that each user is registered to a cloud service in advance. Upon the user registration, each user is notified the network address of a UP, through which each user can reach the cloud server. It is the proxy/user manager that assigns and relocates each user to a UP.

Proxy/User Manager: The proxy/user manager manages users, UPs, GPs, and gateway routers. It is a process executed at one of the host computers within the network domain of a cloud provider. The proxy/user manager performs the following tasks:

(a) Assign users to UPs and relocate users to different UPs when a DDoS attack is detected at a UP.

(b) Assign UPs to GPs and relocate UPs to different GPs when a DDoS attack is detected at a GP.

(c) Assign GPs to gateway routers and relocate GPs to different gateway routers when a DDoS attack is detected at a gateway router.

(d) Maintain the Bloom filters at UPs, GPs, and gateway routers, and update the Bloom filters when the proxy/user manager assigns and relocates users, UPs, and GPs.

Bloom filter is a data structure that can be used to efficiently test the presence of a certain value in a data set [22]. Bloom filter at each proxy is used for two purposes: (1) to drop incoming network traffic from external attackers (those who do not have a legitimate account in a target server) without consuming much resources, and (2) to prohibit legitimate users or internal attackers (defined in section 4.1) from using their previous proxy after they are migrated to another proxy.

B. Procedure of the Migrating Overlay

The proxy/user manager maintains User Management Table (UMT) that holds the information for each registered user. UMT consists of the following four fields for each user:

Permanent User ID: A unique permanent identification number for each registered user

User Public Key: The public encryption key created by a registered user

Current Secret Number: A unique secret number created by the proxy/user manager. It is a number each user must show to the Bloom filter at a UP. This secret number is periodically changed by the proxy/user manager to prevent attackers from masquerading a legitimate user.

Current UP: The network (IP) address of the UP currently assigned to each user

In LMO, each user must register its public key to the proxy/user manager of the cloud provider. User registrations are performed by a server isolated from the cloud server. The user registration server assigns each new user a unique permanent user ID, , the current secret number, and f an initial UP. The three pieces of the information are forwarded to the proxy/user manager upon a successful user registration.

The proxy/user manager encrypts the current secret number of the new user and the network address of the initial UP using the public key of the new user. The proxy/user manager forwards the user's permanent ID and the encrypted information to the user information distributor (UID). Each user obtains his initial UP and secret number from the UID. UID is mirrored to multiple hosts and a different set of mirrored UID hosts are assigned to each user.

The proxy/user manager updates the Bloom filter of the UP assigned to the new user by inserting the current secret number of the new user to the Bloom filter. The proxy/user manager sends the updated Bloom filter to the UP through a GP. When the UP receives the updated Bloom filter, it replaces its local Bloom filter by the one updated by the proxy/user manager. The proxy/user manager does the same for GPs and gateway routers.

When a registered user needs access to the server, the user first contacts one of the UIDs assigned to the user to obtain the user's current secret number and the network address of the initial UP. A user presents the user's permanent user ID to a UID. If a matching user ID is found, the UID sends the encrypted information (which contains the user's secret number and the address of the initial UP) to the user.

Step 1: Extract the UDP payload size in bytes and updates the aggregated traffic load to the cloud server through this UP.

Step 2: Issue a DDoS warning message to the proxy/user manager through a gateway proxy if the aggregated traffic load exceeds a certain threshold.

Step 3: Perform network address translation (i.e., replace the source and destination network addresses in the UDP packet by the network address of this UP and the one for the GP this UP is assigned to).

Step 4: Transmit the UDP packet to the cloud server through the GP.

Fig. 2: Procedure Performed by Each UP for Handling Each Incoming User UDP Packet

When a legitimate user receives a response from a UID, the current secret number and the network address of the initial UP are decrypted using the user's private key. For every message the user sends to the cloud server, each user includes the current secret number at the top of the payload field in each UDP packet. When the UP receives a UDP packet from a user, it first extracts the user's current secret number and tests it using its local Bloom filter. If the user's current secret number is not recognized by the Bloom filter, the UDP packet is dropped. For network applications that require reliable connections, the flow and error controls on the UDP packets should be implemented in the application layer. Fig. 2 summarizes the procedure.

(i) The first user (the first one to register to a cloud service) is assigned to one of the initial UPs (say, $UP_{[i]}$).

(ii) Each time a new user registers after the first user, the user is assigned to $UP_{[i]}$ up to the point where $N_{[i]} = N_{MC}$.

(iii) For the first new user after $N_{[i]} = N_{MC}$ is reached, the user is assigned to another initial UP.

(iv) If a user leaves $UP_{[i]}$, $N_{[i]}$ is decreased by one.

(v) If $N_{[i]}$ of $UP_{[i]}$ reaches zero, the UP becomes inactive.

Fig. 3: Procedure of Allocating and Removing Legitimate Users to and from an Initial UP

Step 1: If $N_{[i]} = 1$ at $UP_{[i]}$ while an attack is in progress at the UP, the active user must be an attacker. The UP is shutdown (forced to be inactive). The permanent identification of the user is blacklisted and it is advertised to all other active UPs so that their Bloom filters drop any message from the user. Then, the procedure is terminated. Otherwise, proceed to Step 2.

Step 2: Activate two extra UPs ($UP_{[m]}$ and $UP_{[n]}$). Split the active users at $UP_{[i]}$ to two groups and assign the groups to $UP_{[m]}$ and $UP_{[n]}$.

Step 3: After all attacking hosts are eliminated, the legitimate users are reassigned to $UP_{[i]}$. Otherwise, go back to Step 1.

Fig. 4: Procedure of Handling DDoS Warning Message by Proxy/User Manager

The proxy/user manager performs the initial assignment of a UP to new users, relocation of users, and identifying attackers. Its procedures are shown in Fig. 3. When a UP detects excess traffic to the cloud server, it issues a DDoS warning message to the proxy/user manager through a GP. The proxy/user manager then performs the procedure defined in Fig. 4. In Fig. 4, extra UPs are recursively split to half until all attackers in the user group at an initial UP are identified. After all attackers are successfully identified, the legitimate users in the group are re-assigned to their initial UP. Each UP is assigned a unique UP identification number as denoted "$UP_{[i]}$" in the figures. The two other parameters are:

NMC: the maximum number of users that can be assigned to a UP

$N_{[i]}$: the total number of users currently assigned to $UP_{[i]}$.

Each UDP packet transmitted by a user to a UP contains the user's current secret number assigned by the proxy/user manager in plain text. If attackers have access to the secret number, they can spoof UDP packets from legitimate users. This lets such spoofed UDP packets reach the cloud server to successfully launch DDoS attacks.

We propose a solution that periodically changes the secret number for each user so that attackers can not use the captured secret numbers for launching DDoS attacks to effectively disable the cloud server. The proxy/user manager periodically generates a secret number using a random number generator function and a random seed, while the generator function and a random seed are agreed on between the proxy/user manager and a user. The random seed is periodically updated by the proxy/user manager. The proxy/user manager periodically posts a new random seed for each user to UIDs, where users obtain a new seed. The time interval for switching to a new seed should be agreed on between the proxy/user manager and a user.

The time interval of generating new secret numbers should be short enough to prevent attackers' UDP packets from penetrating to an overlay through UPs as predicted by formula (1):

$$t_2 + t_3 > t_1 \quad (1)$$

where:

t_1: the time interval for the proxy/user manager to generate a new secret number for each user

t_2: the delay a DDoS coordinator needs to capture a secret number

t_3: the delay a DDoS coordinator needs to coordinate a group of DDoS bots that is large enough to disable a target server host computer

Gateway proxies (GPs) are used to hide the network address of the gateway routers from UPs. We hypothesized that users' computers can be hijacked by DDoS attackers and that some legitimate users will not conform to the required security procedures to keep the locations of UPs secret. Based on the hypothesis, we assumed that the likelihood of UPs to be hijacked by attackers is not negligible because of their direct exposure to users. LMO makes sure that none of the users can easily discover the location of the GPs behind UPs. Thus, the GPs provide another defense line for reducing the risk of an overlay network being penetrated by DDoS attackers.

The GPs play the similar role the UPs do for users. One GP serves multiple UPs. Whenever a GP detects excess network traffic to the cloud server, it issues a DDoS warning message to the proxy/user manager through a gateway router. The proxy/user manager splits the UPs currently assigned to a GP, just like it splits users assigned to a UP under an attack. This mechanism is introduced also for coping with sophisticated DDoS attacks, where a DDoS coordinator distributes high network traffic to the cloud server through a large number of hijacked user host computers in such ways that none of the individual UPs can detect excess network traffic to the cloud server. The proxy/user manager also applies the same splitting method to the gateway routers.

IV. Performance Evaluation

This section describes our performance evaluations of LMO using discrete event driven simulation. The two primary metrics of our interest were the convergence delay and the number of extra proxies needed for identifying the attacking host computers. This section describes modeling of user and attacker behavior (Section 4.1), definition of each experiment (Section 4.2), and descriptions of the outcomes from our experiments, as well as our analyses on the outputs from our experiments (Section 4.3). The performance study did not include the communication overhead for notifying the network address of a new UP to each user when users at a UP are split to two extra UPs. We are currently making progresses on the performance evaluations of LMO including the overhead.

A. Modeling Behavior of Legitimate and Attacking Users

We hypothesized three different classes of "users" in our performance evaluation: legitimate users, external DoS/DDoS attackers, and internal DoS/DDoS attackers. The legitimate users are those who have own legitimate account in a cloud service. The legitimate users have no intention to deplete resources in a cloud server and they are assumed never to do so. The external DDoS attackers are those who do not have a legitimate account in a cloud service. Since the external DDoS attackers do not have a legitimate account, the form of their DDoS attacks is mainly flooding attack, depleting the bandwidth of the uplink to a cloud provider and/or exploiting bugs in authentication or login process, which are prevented by the Bloom filter and migrations of UPs and GPs.

The internal DDoS attackers are those who have a legitimate account in a cloud service. The majority of the internal attackers are presumably the legitimate users whose host computers are hijacked by a DDoS coordinator by infection of viruses, or they can be legitimate users who converted to attackers by some reasons (such as personal revenges or mischiefs to a cloud provider). The primary difference between the internal and the external attackers is that the former has accesses to the resources in a cloud server by passing the user authentication.

Because of the following three reasons, we focused only on the internal attackers (the term "attackers" means "internal attackers" hereafter):

1. Since external attackers are not able to penetrate through user authentication process at UPs, they will not deplete resources at a cloud server.
2. Although external attackers can effectively deplete link bandwidth and computing resource at UPs, once it happens, such UPs under an attack can be migrated to

other UPs. This isolates the legitimate users from the external attackers.

3. External attackers will be efficiently dropped by Bloom filter without depleting significant resources at UPs.

The behavior of each user is modeled by a sequence of "online time" and "offline time". The online time is the time interval a user connects to a service in a cloud server and performs some operations, while the offline time is the time interval between two "online time" intervals. We applied Bounded Pareto and Exponential distributions to randomly generate the online and offline time intervals for each user. We also assumed that each attacking user will stay online without going back to offline state once it becomes online, while each legitimate user will repeat online and offline statuses. The parameters used for modeling the user behaviors are listed in Table I. For Bounded Pareto distribution, Pareto alpha = 1.161 was used to simulate "80/20 rule" and we assumed that its upper bound was ten times of the mean (i.e., 1800 seconds).

B. Experiment Designs

We developed seven experiments using the number of all users, the ratio of the attacking users to all users, the number of the initial UPs, and attack detection latency at each proxy. The number of all users included both legitimate and attacking users. The ratio of the attacking users was used to calculate the number of attacking users. For example, if the ratio is 1/8 while 2048 total users participate in an experiment run, 256 (= 2048/8) users are attackers and the rest (i.e., 1792 (= 2048 – 256) users) are legitimate users. The number of the initial UPs is the total number of the UPs deployed at the beginning of an experiment when no user-split has been performed yet. Users are evenly distributed to each of the initial UPs. For example, if there are 2048 users and if there are 16 initial UPs, each initial UP is assigned 128 users, of which 16 are attackers and 112 are legitimate users if the attacker ratio is 1/8. The attack detection latency is the time interval (in seconds) before a UP recognizes an attack after an attacker becomes "online" or after an active attacker is migrated to another UP.

Table II lists our experiments except Experiment #7. Table III shows the parameter values used for the experiments (except #1 and #7). We repeated the same experiment for ten times (by randomly generating user behaviors with a different random seed) and the outputs from the ten runs of the same experiment were averaged for the convergence delay and the number of the extra UPs. The convergence delay is defined as the elapsed time since the first attacker became active until the last attacker was identified in an experiment run. Extra UPs are the UPs (other than the initial UPs) that are activated only for splitting a user group at an initial UP to identify the attackers in the user group. As soon as all the attackers were successfully identified, those extra UPs became inactive.

Experiment #7 observed the convergence delay to identify all the 128 attackers in Experiment #1 when 2, 4, and 8 GPs were used on top of the 16 UPs in Experiment #1. In the experiment, 8, 4, and 2 UPs were connected to one of the 2, 4, and 8 GPs respectively. Each GP was assigned the maximum traffic rate (R_i bps) it could forward to the cloud server through a gateway router. When a GP detected ongoing DDoS attack by detecting the excess aggregate traffic from the UPs connected to this GP. Then, the group of UPs at this GP was split to two extra GPs. Each of the two extra GPs was assigned the maximum traffic rate R_j, which was calculated by $R_j = R_i/2$. This procedure was recursively repeated until hijacked UPs (or the UPs that contributed to the excess traffic to the cloud server) were identified. The identified mal-conducting UPs were blacklisted and shut down by the proxy/user manager. They were eliminated from the list of the available UPs. When a hijacked UP was shut down, the users allocated to the UP were assigned to another initial UP, which was activated to replace the hijacked UP. In Experiment #7, the overhead of the communications between GPs and the proxy/user manager, as well as those for the users to relocate them to a new initial UP, was not implemented because of the "millisecond time order" of their overhead.

C. Outcomes from Our Experiments and Analyses

Fig. 5(a) shows the outcomes from Experiment #1. The figure contains graphs for the number of the extra UPs and the cumulative number of the identified attacking users at each second since the beginning of an experiment run (average of ten experiment runs). Fig. 5(b) shows the outcomes when the number of the initial UPs was reduced from 16 to 4 with all the other parameters unchanged. The figures visualize the working of LMO. When a session of DDoS attack was initiated by a DDoS coordinator (at around 50 and 100 seconds since the beginning of an experiment in (a) and (b)), LMO started launching extra UPs to split users at an initial UP under attack. With approximately 30 to 50 seconds after the deployment of the first extra UP, LMO started nailing down the attacking users. When approximately one-third to half of the attacking users were nailed down, the number of the extra UPs reached the peak (at around 250 and 320 seconds after the beginning of an experiment). After "take off" of the cumulative number of the identified attacking users, the number constantly grew, while the number of the deployed extra UPs gradually decreased after the peak.

Fig. 6 shows the outcomes from Experiment #2. The graphs show the convergence delay for the number of the identified attacking users for 512, 1024, 2048, 4096, and 8192 total users (with 64, 128, 256, 512, and 1024 attacking users respectively), while the number of the initial UPs remained constant (16 initial UPs). The dashed line indicates a projection of the expected convergence delay. The projected trend shows a parabola curve with its vertex at (0, 0) origin point in the graph. This outcome suggests that LMO will efficiently cope with a large number of coordinated DDoS attacking users in terms of the time necessary to nail them down.

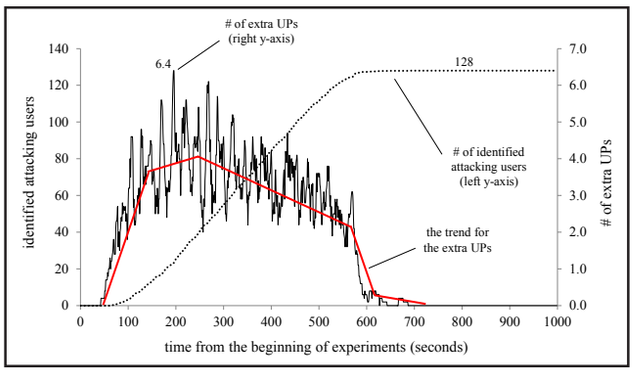

Fig. 5(a): Convergence Delay and Number of the Used UPs

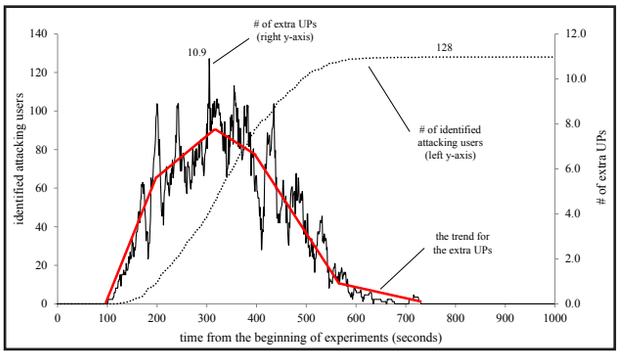

Fig. 5(b): Results of Experiment #1 When Only Four Initial Ups Were Deployed

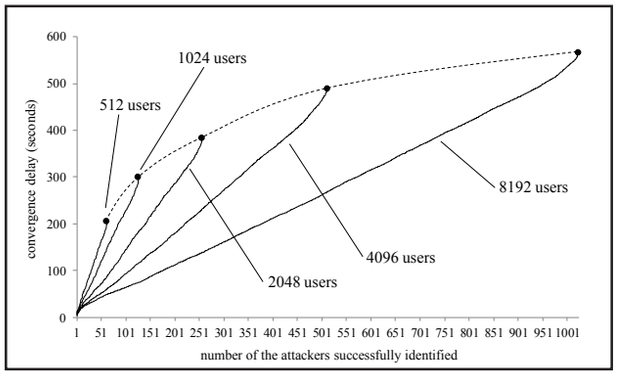

Fig. 6: Convergence Delay for Different Number of Users

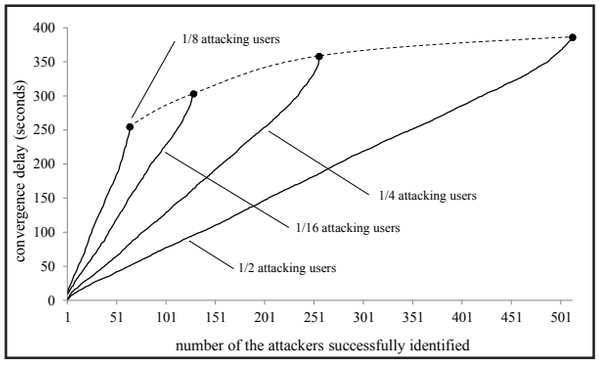

Fig. 7: Convergence Delay for Different Attacker Ratio

Fig. 7 shows the outcomes from Experiment #3. The graphs show the convergence delay for the number of the identified attacking users when the ratio of attacking users to all users was 1/2, 1/4, 1/8 and 1/16, while the total number of users and the number of the initial UPs remained constant (1024 total users and 16 initial UPs). In the outcomes, we observed a similar result from Experiment #2. When a total of 512 attacking users existed for 1/2 attacker ratio (1024 × 1/2 = 512), it took 386.2 seconds to identify all of them, while it took 252.9 seconds when a total of only 64 attacking users existed for 1/16 attacker ratio (1024 × 1/16 = 64). These outcomes suggest that LMO will adapt well to increases in the scale (in the number of the participating attacking users) of coordinated DDoS attacks.

Fig. 8 shows the outcomes from Experiment #4. The graphs show the convergence delay and simple linear regression on the number of identified attacking users for 4, 8, 16, and 32 initial UPs, while the total number of users and the attacker ratio remained constant (1024 total users and 1/8 attacker ratio). The observed linear coefficients were 0.742, 1.317, 2.215, and 3.676 for 4, 8, 16, and 32 initial UPs respectively. Contrary to our prior expectation, the experiment suggests the fewer the number of the initial UPs is, the shorter the convergence delay will be to identify all attacking users. This result implies that LMO will be a scalable solution in the number of UPs needed to cope with large-scale DDoS attacks. The major downside of having fewer initial UPs is the slowdown to a larger number of legitimate users. Since fewer initial UPs means a larger number of users, especially legitimate users, at each initial UP, a larger number of legitimate users will be involved in a sequence of user splits once an initial UP launches user splits.

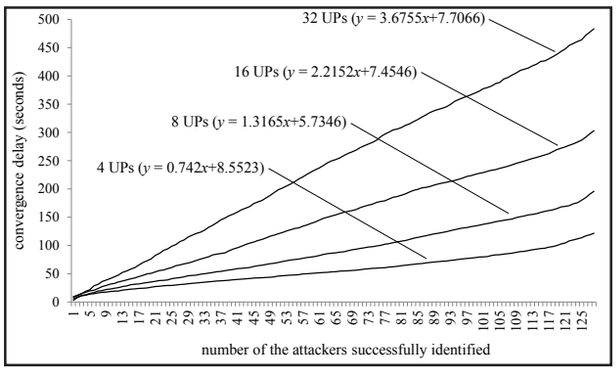

Fig. 8: Convergence Delay and Its Linear Regression for Different Number of Initial UPs

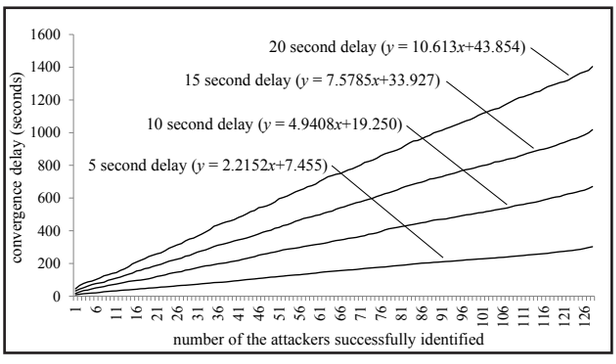

Fig. 9: Convergence Delay and Its Linear Regression for Different Attack Detection Latency

Fig. 9 shows the outcomes from Experiment #5. The graphs show the convergence delay and the simple linear regression on the number of the identified attacking users when the expected latency for detecting a DDoS attack at each UP was 5, 10, 15, and 20 seconds while all the other parameters remained unchanged from Experiment #1. The graphs showed that the attack detection latency had the most significant impact in our experiments to the convergence delay. When the latency was increased by four times to 20 seconds, the linear coefficient exceeded 10.0. This implies that the attack detection latency, which indicates how quickly each UP can detect excess traffic through it, is a key for the convergence delay in LMO especially for large-scale DDoS attacks.

Fig. 10 shows the outcomes from Experiment #6. The figure shows the average and peak number of the extra UPs. The linear regressions of the graphs demonstrated the scalability of LMO. Although the peak number of the extra UPs increased as the user group size increased (from 256 to 8192 total users), its impact to the peak was minor (a linear coefficient of 1.269), while the average usage of extra UPs was almost constant no matter how large the user group was (a linear coefficient of 0.203). These observations suggest that LMO will efficiently sieve DDoS attacking hosts in many different situations without requiring a large number of extra UPs, such as when a small number of attacking hosts hide behind a large legitimate user group, or when a stampede of DDoS attacking hosts occupy the majority of incoming traffic.

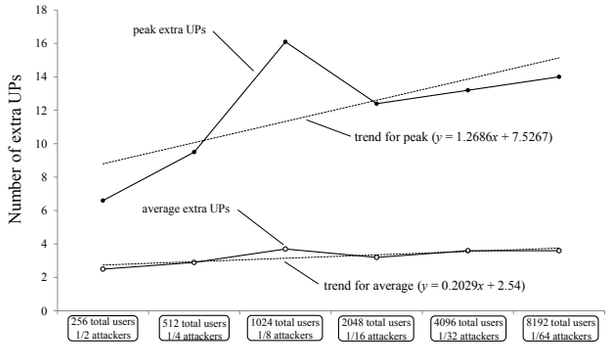

Fig. 10: Total and Peak Number of Extra UPs Used in an Experiment Run

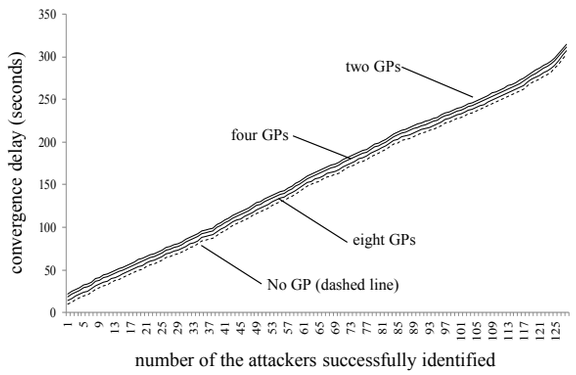

Fig. 11: Convergence Delay when GPs Triggered User-Splits at UPs

Fig. 11 shows the outcomes from Experiment #7. It was observed that all the 128 attacking hosts in Experiment #1 were nailed down while the convergence delay was extended by 4.2, 8.9, and 12.2 seconds when 2, 4, and 8 GPs were used as the second line of defense. The extension was only 1.37, 2.85, and 3.87% to their convergence delay. Because of the same nature of the procedure, we expected that even including the gateway routers as the third defense line, the increase in the convergence delay would be approximately double of the numbers (i.e., 8.4, 17.8, and 24.4 seconds). These observations suggest that the convergence delay for sieving attacking traffic by the multiple defense lines of migrating proxies will be feasible.

V. Future Research Direction

This paper presents an approach to effectively protect production servers by preventing DDoS attackers from reaching servers using overlay networks. Having an overlay network for protecting a server is not commercially cost-effective. Especially if a group of DDoS attackers employ attacks to a specific server at a time (which is usually the case, otherwise DDoS attackers will not be effectively employ DDoS attacks to target servers), sharing an overlay network for multiple servers can be a next step. One way to realize this would be by using "overlay networks as a cloud". Farther study is necessary to assess the effectiveness of the cloud-base overlay approach, as well as to develop a protocol to coordinate routers in overlays.

VI. Conclusions

The two primary contributions by this work are designing a new protection mechanism for network servers from DDoS attacks using migrations of proxies and dynamic binary user splits, called "Layered Migrating Overlay (LMO)", and assessing its potential. In protecting network servers from DDoS attacks, this work primarily focuses on the situations where legitimate users' host computers are hijacked by DDoS coordinators. Although we designed LMO for protecting cloud servers, LMO can be applied to most of the network servers that require user authentication (this requirement is for the Bloom filter at each UP). We evaluated the potential of LMO, which identifies and drops attacking traffic without monitoring each user's traffic load. We assessed the potential of LMO using event-driven discrete simulation using multiple control parameters (number of total users, attacker ratio, number of initial UPs, and attack detection latency at UPs).

Based on the observations from our experiments, we draw the following conclusions. Migrating proxies in an overlay can efficiently identify and drop attacking traffic without monitoring each user's traffic load (Experiment 1). The dynamic binary user split in LMO is scalable in the number of users who access a server (Experiment 2) and in attacker ratio (Experiment 3). The results from Experiment 6 also suggest that LMO will efficiently sieve DDoS attacking users in many different situations without increase in the required extra UPs, such as when a small number

of attacking hosts hide behind a large legitimate user group, or when a stampede of DDoS attacking hosts occupy the majority of incoming traffic. *f* Contrary to our prior expectation, fewer initial UPs will result in shorter convergence delay (Experiment 4). This implies that a large number of initial UPs are not necessary for coping with large-scale DDoS attacks, making LMO a scalable, and therefore, an efficient solution. „ The attack detection latency, which indicates how quickly each UP can detect excess traffic through it, is the factor that resulted in the largest linear coefficient (Experiment 5). … The second defense line (i.e., the layer of "gateway proxies") added a minor delay on top of the convergence delay incurred by UPs. Since the third defense line (i.e., adding the gateway routers) should work logically in the same way, adding extra defense lines can be achieved without significantly increasing the convergence delay. This "multi-layer defense lines" of migrating proxies is expected to enhance the strength of the protection.

The future work in our high priority includes the following tasks: performance studies using prototypes, performance study on network latency, and *f* performance study on the overhead of switching proxies. The first ( ) is an essential testing that will let us study the impact of network system software (various network protocols, such as those for data link, network, and transport layers, as well as those for routing in both intra and inter-domain routing) when LMO is executed on top of them. The second (,) will let us study the impact of dynamically changing end-to-end delay as well as the impact of other ongoing network traffic to the working of LMO. This study will let us evaluate the situations when users connect a server from different places in a large-scale network, such as the Internet. The third (*f*) is the highest priority task to assess the overhead for notifying new extra UPs to users during user-splits. We are currently making progresses on these three areas of the future work.

References

[1] B. Prabadevi, and N. Jeyanthi, "Distributed denial of service attacks and its effects on cloud environment- A survey," *Proceedings of the International Symposium on Networks*, *Computers and Communications*, pp. 1-5, 2014.

[2] M. Durairaj, and A. Persia, "ANM to perceive and thwart denial of service attack in WLAN," *International Journal of Communication Networks and Information Security*, vol. 7, no. 6, pp. 59-66, 2015.

[3] G. Booth, A. Soknacki, and A. Somayaji, "Cloud Security: Attacks and Current Defenses," *Proceedings of the Annual Symposium on Information Assurance*, pp. 56-62, 2013.

[4] P. Ferguson, and D. Senie, "Network ingress filtering: Defeating denial of service attacks which employ ip source address spoofing," *RFC-2827*, May 2000.

[5] J. Ioannidis, and S. M. Bellovin, "Implementing push-back: Router-based defense against DDoS attacks," *Proceedings of Network and Distributed System Security Symposium*, pp. 100-108, 2002.

[6] M. Darwish, A. Ouda, and L. F. Capretz, "Cloud-based DDoS attacks and defenses," *Proceedings of the International Conference on Information Society*, pp. 67-71, 2013.

[7] E. Cooke, F. Jahanian, and D. McPherson, "The zombie roundup: Understanding, detecting, and disrupting botnets," *Proceedings of the Steps to Reducing Unwanted Traffic on the Internet on Steps to Reducing Unwanted Traffic on the Internet Workshop*, p. 6, 2005.

[8] E. Shi, I. Stoica, and D. A. Perrig, "OverDoSe: A generic DDoS protection service using an overlay network," Carnegie Mellon University Computer Science Department Technical Report CMU-CS-06-114, 2006.

[9] S. Khattab, C. Sangpachatanaruk, R. Melhem, D. Moss´e, and T. Znati, "Proactive server roaming for mitigating denial-of-service attacks," *Proceedings of International Conference on Information Technology Research and Education*, pp. 286-290, 2003.

[10] N. Jeyanthi, N. Ch. S. N Iyengar, P. C. M. Kumar, and A. Kannammal, "An enhanced entropy approach to detect and prevent DDoS in cloud environment," *International Journal of Communication Networks and Information Security*, vol. 5, no. 2, pp. 163-173, 2013.

[11] T. Okumura, D. Mosse, M. Minami, and O. Nakamura, "Operating system support for network control: A virtual network interface approach for end-host OSS," *Proceedings of IEEE International Workshop on Quality of Service*, pp. 170-179, 2002.

[12] C. Sangpachatanaruk, S. M. Khattab, T. Znati, R. Melhem, and D. Moss´e, "A simulation study of the proactive server roaming for mitigating denial of service attacks," *Proceedings of the Annual Symposium on Simulation*, pp. 7-14, 2003.

[13] V. K. Pingali, and J. D. Touch, "Protecting public servers from DDoS attacks using drifting overlays," *Proceedings of the IEEE Computer and Information Technology Workshops*, pp. 270-272, 2008.

[14] H. Wang, Q. Jia, D. Fleck, W. Powell, F. Li, and A. Stavrou, "A moving target DDoS defense mechanism," *Computer Communications*, vol. 46, no. 15, pp. 10-21, June 2014.

[15] J. Kurian, and K. Sarac, "Provider provisioned overlay networks and their utility in DoS defense," *Proceedings of IEEE Global Telecommunications Conference*, pp. 474-479, 2007.

[16] S. Khattab, R. Melhem, D. Moss´e, and T. Znati, "Honeypot back-propagation for mitigating spoof-

ing distributed denial-of-service attacks," *Proceedings of the IEEE International Parallel and Distributed Processing Symposium*, pp. 1152-1164, 2006.

[17] S. Khattab, C. Sangpachatanarukz, D. Moss´e, R. Melhemx, and T. Znatixz, "Roaming honeypots for mitigating service-level denial-of-service attacks," *Proceedings of International Conference on Distributed Computing Systems*, pp. 328-337, 2004.

[18] D. L. Cook, W. G. Morein, A. D. Keromytis, V. Misra, and D. Rubenstein, "Web SOS: Protecting web servers from DDoS attacks," *Proceedings of the IEEE International Conference on Networks*, pp. 455-460, 2003.

[19] D. G. Anderson, "Mayday: Distributed filtering for internet services," *Proceedings of USENIX Symposium on Internet Technologies and Systems*, pp. 3-15, 2003.

[20] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 149-160, 2010.

[21] M. Canini, D. Fay, D. J. Miller, A. W. Moore, and R. Bolla, "Per flow packet sampling for high-speed network monitoring," *Proceedings of the Communication Systems and Networks and Workshops*, pp. 1-10, 2009.

[22] A. Broder, and M. Mitzenmacher, "Network applications of bloom filters: A survey," *Internet Mathematics*, vol. 1, no. 4, pp. 485-509, November, 2003.