Journal of Theoretical and Applied Information Technology
10th October 2015. Vol.80. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

LIMIT ANALYSIS OF THE IPFH

VLADIMIR KRYLOV, ELEANORA SOKOLOVA, KIRILL KRAVTSOV

Nizhny Novgorod State Technical University n.a. R.E. Alekseev, Russia

ABSTRACT

In this paper we introduce IP Fast Hopping, easily deployable and effective network-layer architecture against Distributed Denial of Service attacks and discuss its major limitations. Our approach provides an easy way for clients to hide content and destination server of theirs communication sessions. We describe a method of dynamic server IP address change and all modules necessary to implement the approach.

Keywords: *Internet security, DDoS attacks, traffic interception, switcher*

1. INTRODUCTION

1.1. DDoS attacks

In this article, we discuss security challenges in TCP/IP networks and, especially, Software Defined Networks. Such networks are vulnerable to various types of network attacks like traffic eavesdropping, IP Spoofing Attacks [6], Denial of Service etc. (see [3] for a more detailed classification).

A Denial-of-Service attack is characterizes by an explicit attempt to prevent the legitimate use of a service. A Distributed Denial-of-Service attack deploys multiple attacking entities to attain this goal [13]. In such attacks, a single bot or a group of bots are sending a large number of packets that lead to exhausting of victim's bandwidth capacity or software processing capabilities.

According to [13], methods of DDoS attacks can be divided by the two groups: semantic attacks and brute-force (flood) attacks. A semantic attack exploits a specific feature or implementation bug of some protocol or application installed at the victim in order to consume excess amounts of its resources. For example, an attacker can send a specific consequence of packets initiating CPU time consuming procedures on the server. In case of a large number of such requests, the victim is unable to handle requests from legitimate clients. Undesirable impact from such attack can be minimized by protocol or software modifying and by applying of special filter mechanisms. In our paper, we introduce a DDoS defense mechanism that aims to filter all TCP traffic issued by unauthorized clients on network level. Therefore unauthorized malefactor is unable to perform semantic attacks based on TCP protocol on the victim server.

A brute-force attack is an attack aimed to prevent legitimate service using by exhausting of bandwidth. E.g. it is a large number of short requests to the victim which initiates heavy responses sent by the victim. Together these streams overfills bandwidth of the victim server or it's ISP. In contrast to semantic attacks, brute-force attacks abuses legal services so installing of filtering mechanisms for such requests will impact traffic from legitimate client too. During brute-force DDoS attacks a number of malefactor terminals (botnet) and legitimate users are connected to the victim at the same time (see Figure 1). Each bot sends a big number of requests to the victim which creates heavy malicious traffic targeted at the server. Since the increase in the flow of requests is created here increase the number of terminals, then whichever level of server performance has not been achieved starting from a certain number of bots, they create the flow of requests exceeds the permissible level for any server.

1.2. Importance of new DDoS solutions

Size and frequency of DDoS attack is continue to grow [19] despite on the fact that a large number of defense mechanisms have been proposed. According to [18] application layer attacks rose approximately 42% in 2013 from 2012, infrastructure layer attacks increased approximately 30% at the same period. The infrastructure layer DDoS attacks are still most popular: around 77% of DDoS attacks in Q4 2013 were infrastructure layer attacks. The average size of attacks increased by 19.5% from Q1 2012 to Q1 2013 (up to 1.77 GB/s) [9]. So, developing new DDoS prevention mechanisms is still a topical issue.

Journal of Theoretical and Applied Information Technology
10th October 2015. Vol.80. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

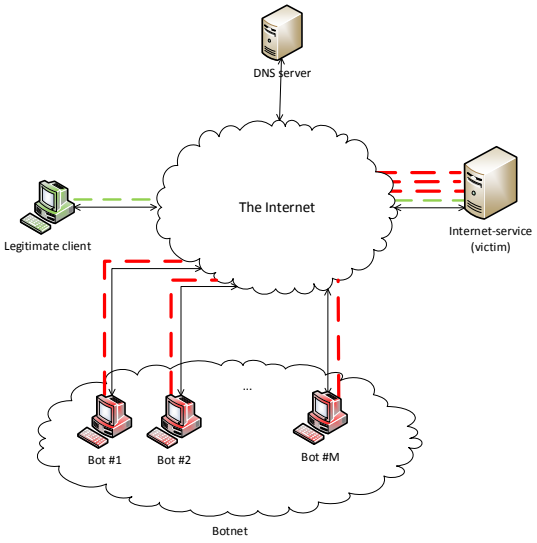

Figure 1. Schema Of Brute-Force Ddos Attack

1.3. Related work

In related literature, a certain number of different classifications of DDoS detection and mitigation solutions are presented. According to various principles, they can be divided into groups: based on the location of their deployment (source-based, network-based or destination-based) [20], based on the type of the applied algorithm (statistical, knowledge-based, soft computing, data mining and machine learning methods [15] and, according to other principles, there are a couple of more groups to be found [7]. Each new DDoS defense method should satisfy the following main principles:

Real world applicability. Now a number of different approaches have been suggested in literature which require a significant changes of existing network architecture of ISPs or entire Internet architecture. But the main problem here that mostly DDoS attacks threaten organizations which provide services to end users [17], and so this problem is not very vital for transit ISPs because they actually suffer very little from such attacks. Thus, priority will be given to such systems, which filters malicious traffic without changing of global network architecture, into server's or edge ISP's networks.

The solution must be designed to prevent misuse. So, it must be not possible to exploit the method to increase impact issues by an attack or to filter legitimate traffic.

The rise of DDoS attack frequency in recent years has resulted in many proposed defense approaches from the community. For example, patent [5] suggests solution on application level. According to the article, the server sends a special response on first connection attempt from a client. The client uses this response to identify new URL address, after that the client creates new requests and sends this new requests to this new URL. After receiving this second request, the server validates the new URL based on sent response. If the value does not exceed preset load threshold, the packet will be prioritized and processed by the server. Thus the solution is based on applying of special filter on server side; this filter controls prioritizing of client's requests depending on load on the server at the moment of receiving of the first request from the client.

Other high level approach was suggested under [2]. This method introduced special server responsible for creating and updating of cryptographically secured keys. Each client can access to the service only after successful legitimacy verification on this special server. Thus, client should be successfully authorized on the special server to get a secure key which should be used for processing of a special scenario. The aim of this scenario is identifying client's legitimacy. These approaches purpose defense methods on application layer and does not impact IP packets exchange. So, a malefactor can perform brute force attack on the server.

Also, the research community suggested a wide scope of different more low level approaches. For example, [12] purposes dividing of data stream transmitted between server and client into two consecutive segments on TCP level. This work suggested comparing of keys of two consecutive segments to detect possible segments from not legitimate source. In case of detection of such segments, data receiving will be blocked to prevent possible impact from attack.

Paper [14] introduces DDoS defense mechanism based on dynamic change of server's IP address. Server's IP address is changing according to pseudo-random law which is known only for authorized clients. At the first sight the work [14] purposes a similar DDoS prevention mechanism (dynamic changing of IP address), but this contains some significant differences. Among others, are:

IP address of the victim is changing only during active DDoS attack on the server.

Journal of Theoretical and Applied Information Technology
10th October 2015. Vol.80. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The new IP address is assigned for all client sessions simultaneously on a relatively long time (suggested period is around 5 minutes)

Accurate time synchronization is required for calculation of each next IP address since external timestamp is using.

Another paper [1] suggests a network address space randomization (NASR) technique that is intended to protect enabled networks against hitlist worms. This method presupposes that an address from a global IP address space pool is randomly assigned and this randomization suggested to be performed on protected server directly. So, a basic form of NASR can be implemented by configuring the DHCP server to expire DHCP leases at particular intervals. Therefore, the scope of NASR implementation is limited to local regions.

2. METHOD

IP Fast Hopping technique [10], [11] is intended to make the real destination of a client's communication invisible for all external terminals and, consequently, to prevent DDoS attacks and unauthorized access from illegitimate clients. IP Fast Hopping method is based on our model of convoluted multiaddress networks and is implementation of this model in TCP/IP networks. In the IP Fast Hopping approach, a server has a random IP address for each particular client at the each time moment. After this method has been applied, for an external observer close to a client, a communication session between the client and the server does no longer look like a packet stream between these two Internet terminals. Instead of this, an observer detects a packet flow between a client and a number of independent (topologically and physically) terminals in the Internet. None of the streams in this flow has a correlation between packets inside the stream.

IP Fast Hopping is similar to radio systems with frequency hopping. In such systems, a receiver and a transmitter are switching from one frequency to another frequency synchronously during an ongoing data transmission session. A malefactor's transmitter, which is going to introduce a noise into such a session, does not have an actual schedule of frequency hopping; therefore, such an attacker cannot do noticeable harm to the legitimate transmitter defended by the frequency hopping mechanism. In our case, frequency can be treated as an IP address. So, the legitimate client must know the schedule of the server's IP address changing. At the same time, the schedule should be unavailable to non-legitimate clients.

Figure 2. Abstract System Model

To illustrate the basic idea of applying of convoluted multiaddress networks in TCP/IP networks, consider a system model, with a server with address s, a set of clients $C = \{c_1, \ldots, c_I\}$, a subset of network address space $IP_V = \{ip_1, ip_2, \ldots, ip_N\}$ where $s \notin IP_V$ and a set of gateways $R = \{r_1, \ldots, r_M\}$ where $M \leq N$. Each client ci□C has a representation of server's address s: s' or "initial address". Each message a

Journal of Theoretical and Applied Information Technology
10th October 2015. Vol.80. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

from whole set of messages A from the client c_i to the server s is being transmitted via route p with input point P_{in} (gateway) with address y and output point P_{out} (gateway) with address x. In this model we can logically split the system on three independent addressing subsystems: 1) subsystem (subnetwork) W_1 from the client c_i to input point P_{in} of the route p; 2) subnetwork W_2 of the route p; 3) subnetwork W_3 from out point P_{out} of the route p to the server s. When a message a is being transmitted through the particular subnetwork W_z, a has a specific destination address IP_{dstW}. Obviously, $IP_{dstW_1} = s'$, $IP_{dstW_2} = x$, $IP_{dstW_3} = s$ To make entire system model consistent, each W_z has a function F_z that maps the destination address of a message a in the preceding W_{z-1} to a destination address in the current subnetwork W_z (Figure 2).

In common client-server architecture of Internet networks, the destination address of message a is always equal to s, which means that $P_{in} = c_i$ and $x = s$. So, our system model simplifies as shown in Figure 2. In this case, function F_z is trivial and, thus, all messages from c_i addressed to s' are being transmitted directly to server s. In this case, an external observer can easily identify the address of traffic destination and initiate a malicious data stream to address s and this traffic achieves the physical server. Therefore, an unauthorized client can acquire access to the server. In addition, in these conditions, a set of malefactor terminals (botnet) $B = \{b_1, \dots, b_k\}$ can initiate a brute-force DDoS attack on address s (

Figure *3*).

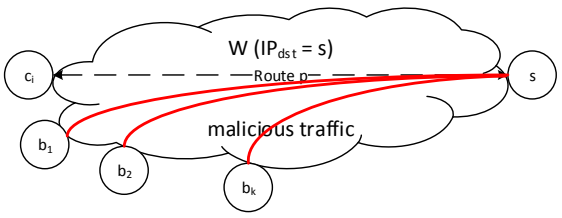

Figure 3. Model Of Common Client-Server Architecture With Brute-Force Ddos Attack

As has been reflected before, nowadays most of DDoS prevention techniques suggests installing firewalls and filtering solutions in a network between a set of clients C and victim server s. With reference to our model (Figure 2), we can treat getaway p_{out} as a firewall, which performs traffic analyzing and filtering according to one of existing defensive methods against DDoS attacks. Therefore, in these approaches, W_3 subsystem remains unprotected and is treated as trusted. Still, in terms of addressing policy the whole system remains transparent, i.e. all messages targeted to the server always have destination address equal to s. In this paper, it is suggested to conceal s in untrusted subsystems W_1 and W_2 in order to hide the location of a victim server and, as a result, prevent unauthorized access from illegitimate clients.

In IP Fast Hopping, $s' = y$ and s is unavailable outside W_3. Each client c_i is connected to shared global network W_2 via security gateway P_{in}. Due to the fact that s', c_i local representation of server's address s, is equal to the address of this gateway, all messages a targeted to the server with address s will achieve this gateway. The packet a has a pair of keys that uniquely identify this message: 1) timestamp (message creation time) of a, t_a, as a public key and 2) unique identification of the client-originator of this packet, ID_{c_i}, as a private key. The system model under consideration has a pseudo-random function $H(t_a, ID_{c_i}) = n \in [1; N]$ that determinates valid virtual address ip_n of the server for particular message a. This function is regarded as a hopper function because of it defines the hopping of the IP address of the protected server in convoluted multiaddress networks. Due to the fact that unique identification of the client is part of the domain of function H, hopping of s can be different for different clients. These virtual addresses ip_n are a representation of s in subnetwork W_2. Therefore, $F_2 = H(t_a, ID_{c_i})$. An address subspace IP_V has a number of disjoint

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

subsets $IP_V^{r_m}$, where $i \in [1; M]$ and $IP_V = \bigcup_1^N IP_V^{r_m}$. Each address from $IP_V^{r_m}$ is related to corresponding gateway r_m, thereby W_2 has M paths p_m for messages from client c_i. Gateway r_m validates all incoming packets and maps destination address of these messages IP_{dst} on address in W_3 using function F_3 (NB: zero address means here that the message should be treated as malicious):

$$F_3(t_a, ID_{c_i}) = \begin{cases} IP_r, & IP_{dst} = ip_n \\ 0, & IP_{dst} \neq ip_n \end{cases}, \quad \text{where } n = H(t_a, ID_{c_i})$$

The same rules are applied for the source address of all responses from the server to clients c_i. As a result, a stream of messages between the server and clients are separated into independent streams between independent terminals in the network.

3. IMPLEMENTATION

As was noted above, one of requirements for our work is real world deployability. Therefore, we implement IP Fast Hopping mechanism as kernel module of OS GNU/Linux. In this case, installing this module on routers based on GNU/Linux is enough to deploy the suggested system. In our work we build such routers based on Debian OS.

Linux kernel contains built-in firewall Netfilter [16], which is responsible on packet filtering and forwarding according to predefined rules by iptables utility. Netfilter architecture is scope of hooks of ordered rules. Netfilter performs a predefined action with a packet, which is passed to a hook, according to the corresponding rule.

Netfilter supports 5 hooks: PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING. When the packet comes to the system, the packet is processed by PREROUTING hook. If this packet is addressed to a local process, it is passed to INPUT hook, otherwise it is passed to FORWARD. All packets sent by local processes are processed by OUTPUT hook. The final processing of the packet outgoing from the system (forwarded under FORWARD hook or issued by a local process) is performing by POSTROUTING hook.

In our work, Netfilter contains new module which is responsible for changing of IP address into destination field of outgoing packets and into source field of ingoing packets. This module is calculating the new IP address according to IP Fast Hopping rules (by timestamp field [19] and session UID). During handshake, IP Hopper Manager adds new set of rules into POSTROUTING hook on client's terminal and into PREROUTING each edge switcher. This rule activates the kernel module which implements the following algorithm:

On the client side this module calculates hash-function using timestamps field and session UID for each outgoing packet addressed to the initial IP address. After that the module uses this result as index of correct address into IP pool which should be put into destination field of the packet. For each ingoing packet from the same communication session, the module performs the same actions for source field: checks the current value of the field (by calculation of the same hash-function) and changes it on the initial address.

On switchers side this module calculates hash-function using timestamps field and session UID for each ingoing packet addressed to IP addresses from IP pool. If the current destination address corresponds to the timestamps field and session UID, the real IP address of the server will be placed into the destination field. Otherwise, the packet will be dropped. For all ingoing packets issued by the server, the module will replace source field by one of virtual addresses according to current value of hash-function.

Implementation of IP Fast Hopping in SDN does not require any significant changes in the suggested implementation. Software Defined Infrastructure is frequently based on GNU/Linux solution, so such basic implementation is compatible and easy to adopt. Furthermore, protected SDN controller may be utilized here to balance load between switches which perform validation of packets addresses.

4. RESULTS AND DISCUSSION

4.1. Initial experimental results

The described basic implementation of IP Fast Hopping approach has been validated on small test stand. The main purpose of our experiments is to show that IP Fast Hopping successfully filters traffic from unauthorized clients. To achieve this goal, we built test stand consisting of several Virtual Machines: client, client's router and server router (both machines had implemented Netfilter module installed), authorization server, victim server and bot. During the experiment, we

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

measured the average traffic intensity at the server's network interface during active DoS attacks and without attack (see

Figure *4*). For generating DDoS attack we used third-party application, LOIQ.

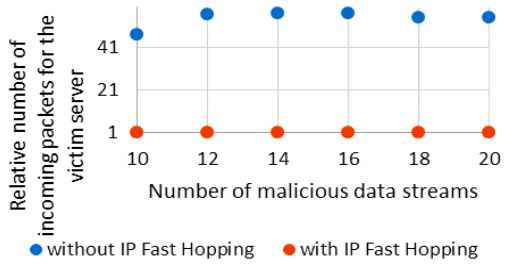

Figure 4 Results of validation of basic implementation of IP Fast Hopping technique

Easy to see, that even such minimal testing has shown that applying of IP Fast Hopping methods leads to filtering of malicious traffic stream from incoming traffic of a protected server.

Implementation of IP Fast Hopping in SDN does not require any significant changes in the suggested implementation. Software Defined Infrastructure is frequently based on GNU/Linux solution, so such basic implementation is compatible and easy to adopt. Furthermore, protected SDN controller may be utilized here to balance load between switches which perform validation of packets addresses.

4.2. Practical constraints

The model under discussion illustrates the basic idea of IP Fast Hopping mechanism and how this technique ensures hiding the destination of a client's traffic from external observers. The approach has a various possible implementations, each of them can have particular limitations, but in this section, we review basic constraints of our technique:

1. If IP pool IP_V is not large enough, a botnet can start an attack on each IP address using masquerading of malicious traffic as legitimate data stream by IP spoofing technique (Farha, 2007). In this case, gateways redirect part of hateful traffic together with legal traffic to a protected server. Therefore, the IP pool, which is used for IP Fast Hopping, should be large enough to make such an excessive attack way too consuming in terms of resources and inefficient for possible attackers. Obviously, the method will be more efficient in IPv6 systems. In this case, IP pool can contain a thousands of addresses related to a number of different routers in the Internet.
2. The method is not session-less; because of the fact that a client unique identification is part of a hopper function's domain.
3. IP Fast Hopping requires clients' authorization since a hopper function (especially its domain and codomain) should be available only to legitimate clients. Therefore, the protocol is not applicable for publicly available Internet resources. In addition, our approach does not provide any native ways for secure and DDoS-resistant client's authorization.

5. CONCLUSIONS

We presented IP Fast Hopping, a new approach that can prevent exhausting of server's resources during brute-force DDoS attacks and can be used to hide content and destination of client's communication session. This method hides the real IP address of the server behind a big number of "virtual" IP addresses. The mapping of the real IP address on one of "virtual" is unique for each communication session and changes dynamically every millisecond. The introduced approach is distributed: it divides the traffic from legitimate users and botnets into a number of sub-streams. This leads to a decrease of load on network infrastructure during active DDoS attack. The method is easily deployable and can filter even the biggest malicious streams.

6. ACKNOWLEDGMENTS

This work was supported by the Ministry of Education and Science of the Russian Federation (grant agreement 14.574.21.0034 from 06/17/2014, the unique identifier applied research RFMEFI57414X0034).

REFERENCES:

[1] Antonatos, S., Akritidis, P., Markatos, E., & Anagnostakis, K. (2005). Defending Against Hitlist Worms Using Network Address Space Randomization. *Proceedings of the 2005 ACM Workshop on Rapid Malcode* (pp. 30-40).

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Fairfax, VA, USA: ACM. doi:10.1145/1103626.1103633

[2] Arun K. Iyengar, Mudhakar Srivatsa, & Jian Yin. (2010). *Patent No. US20100235632 A1*. USA.

[3] Blessy, R., & Deepa, A. (2015). A Survey on Network Security Attacks and Prevention Mechanism. *Journal of Current Computer Science and Technology, 5*(2), 1-5.

[4] Farha, A. (2007). IP Spoofing. *The Internet Protocol Journal, 10*(4). Retrieved from Cisco Systems: http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_10-4/104_ip-spoofing.html

[5] Feng, W.-C., & Kaiser, E. (2010). *Patent No. 20100031315 A1*. USA.

[6] Ferguson, P., & Senie, D. (2000). *RFC 2827 Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing.* Retrieved from http://www.ietf.org/rfc/rfc2827.txt

[7] Gupta, B. B., Joshi, R. C., & Misra, M. (2010). Distributed Denial of Service Prevention Techniques. *International Journal of Computer and Electrical Engineering, 2*(2), 268-276.

[8] Jacobson, V., Braden, R., & Borman, D. (1992). *RFC 1323 TCP Extensions for High Performance.* Retrieved from http://www.ietf.org/rfc/rfc1323.txt

[9] Kerner, S. M. (2013, April 26). *DDoS Attacks: Growing, but How Much?* Retrieved August 8, 2013, from http://www.esecurityplanet.com/network-security/ddos-attacks-growing-but-how-much.html

[10] Krylov, V. V., & Ponomarev, D. M. (2012, May 14). *Patent No. 2496136C1*. Russia.

[11] Krylov, V., & Kravtsov, K. (2014). IP Fast Hopping Protocol Design. *Proceedings of the 10th Central and Eastern European Software Engineering Conference in Russia* (pp. 11:1--11:5). Moscow: ACM.

[12] Marquardt, D. R., Paranjape, P. A., & Patil, P. S. (2011, May 13). *Patent No. 20110283367 A1*. USA.

[13] Mirkovic, J., & Reiher, P. (2004). A taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Computer Communication Review*(Volume 34 Issue 2), 39 - 53.

[14] Mittal, P., Kim, D., Hu, Y.-C., & Caesar, M. (2012). Mirage: Towards Deployable DDoS Defense for Web Applications. arXiv.

[15] Munivara Prasad, K., Rama Mohan Reddy, A., & Venugopal Rao, K. (2014). DoS and DDoS Attacks: Defense, Detection and Traceback Mechanisms - A Survey. *Global Journal of Computer Science and Technology, 14*(7-E), 15-32.

[16] *netfilter/iptables project*. (2014). Retrieved from http://www.netfilter.org/

[17] NSFOCUS, Inc. (2013). *NSFOCUS Mid-Year DDoS Threat Report.*

[18] Prolexic Technologies, Inc. (Q4 4013). *Prolexic Quarterly Global DDoS Attack Report.*

[19] Westervelt, R. (2013, July 18). *DDoS Attack Behind Latest Network Solutions Outage.* Retrieved August 8, 2013, from http://www.crn.com/news/security/240158492/ddos-attack-behind-latest-network-solutions-outage.htm

[20] Zargar, S. T., Joshi, J., & Tipper, D. (2013). A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. Communications Surveys & Tutorials, IEEE, 15(4), 2046 - 2069.