IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
PP 01-07

A Review: Distributed Denial of Service Attack on Cloud Service Models and Modern Detection Mechanisms

Sukhada Bhingarkar
Department of Computer Engineering, MIT College of Engineering, India

Abstract: *Cloud computing is a technology that enables people to access state-of-the-art resources in pay per use approach without the need for capital investment. In spite of these advantages, the organizations opting cloud technology are worried about the security of their data. Distributed Denial of Service attack is considered as one of the most aggressive attacks performed on cloud servers. The goal of this paper is to understand the multiple ways used by attackers to target the cloud service models and the modern detection mechanisms proposed by various researchers. This paper critically analyzes the latest detection approaches and throws a light on research directions in this field.*

Keywords: *Cloud Computing, DDoS, IaaS, PaaS, SaaS, SDN*

I. Introduction

Cloud computing refers to delivery of computing resources over the internet. Cloud computing is equipped with various essential characteristics such as on-demand self-service, resource pooling, broad network access, multi-tenancy, rapid elasticity and metered service.
In spite of various advantages provided by cloud computing, this technology suffers from serious security challenges. One of the major challenges is Distributed Denial of Service attack.

Distributed Denial of Service (DDoS) attack aims at preventing, for legitimate users, authorized access to a system resource. The attacker sends large no. of requests to the target computer with the aim of overloading its resources so that eventually, the target computer starts denying the requests including legitimate one. A hacker begins by gaining the control of initially one computer and treats it as DDoS master. The attacker communicates with other systems by loading cracking tools into it and compromises them. Thus, a group of the systems under the control of an attacker is called a Zombie army. With a single command, attacker instructs the controlled machines to launch flood attacks against the target.

DDoS is commonly carried out on a website. There are two ways in which DDoS attack is carried out on the web site. In the first way, it takes place at the network layer (Layer 3 and 4) and in the second way, at the application layer (Layer 7). At the network layer, attack brings down a website by overwhelming network and server resources, causing downtime and blocking responses to legitimate traffic. UDP Flood, ICMP Flood, Ping of Death are some of the DDoS attacks that are carried out at the network layer. Application layer DDoS attacks mimic legitimate user traffic by searching for content on the site or clicking the "add to cart" button and crash the web server. Another example is, if a bank website can handle 10 people a second clicking the Login button, an attacker only has to send 10 fake requests per second to make it so that no legitimate users can login. HTTP flood is an example of application layer attack. Figure 1 illustrates DDoS attack scenario.

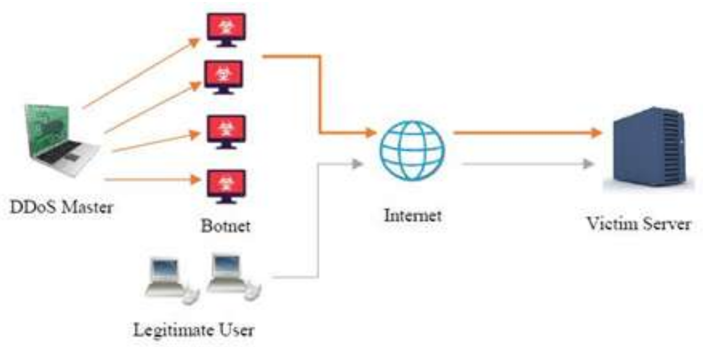


Figure 1. DDoS Attack Scenario

This paper talks about DDoS attack performed on specific cloud service models i.e. Infrastructure as a Service (IaaS), Software as a Service (SaaS) and Platform as a Service (PaaS).

Rest of the paper is organized as follows: Section II discusses all cloud service models. Section III describes DDoS attack performed on IaaS. Section IV illustrates DDoS attack performed on SaaS. Section V focuses on DDoS attack done on PaaS. Section VI thoroughly describes current DDoS detection mechanisms. Section VII critically analyzes existing approaches. Finally, Section VIII concludes the paper.

II. Cloud Service Models

A cloud-computing structure relies on three service layers: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). These cloud services are typically highly flexible, scalable owing to the ready availability of almost unlimited resources, with minimal maintenance cost to their customers, making it ideal for organizations which don't want to buy the entire infrastructure required upfront. Following is the in-detail description of service models.

- ✓ Software as-a-Service (SaaS): The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., Web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure, including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user specific application configuration settings.
- ✓ Platform as a Service (PaaS): The capability provided to the consumer is to deploy onto the cloud infrastructure, consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure, including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.
- ✓ Infrastructure as a Service (IaaS): The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., Host firewalls).

Figure 2 illustrates service models explained above.

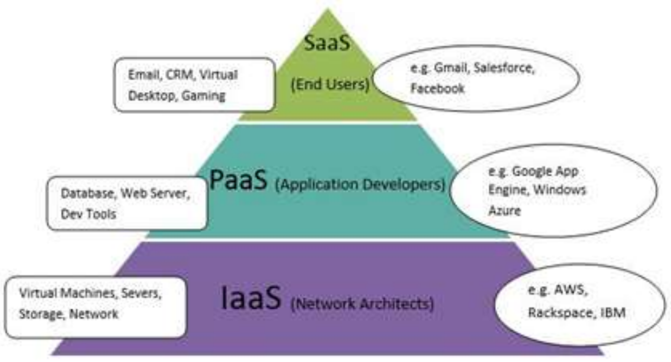

SaaS
(End Users)
Email, CRM, Virtual Desktop, Gaming
e.g. Gmail, Salesforce, Facebook
PaaS (Application Developers)
Database, Web Server, Dev Tools
e.g. Google App Engine, Windows Azure
IaaS (Network Architects)
Virtual Machines, Severs, Storage, Network
e.g. AWS, Rackspace, IBM

Figure 2. Service Models of Cloud Computing

Figure 3: Circuit diagram of the five relays and three contactors used in this design

III. DDoS Attack On Infrastructure-AS-A-Service

The attacker applies different types of DDoS attacks for each type of cloud service. Handling DDoS attacks at all layers in cloud systems is a major challenge due to the difficulty of distinguishing an attacker's request from legitimate user requests.

In case of IaaS service, TCP SYN Flood attack can be more dangerous because virtual machines (VM) are sharing their resources with the neighbor VM and host. Under the TCP SYN Flood, one virtual machine is used as a source of denial of service attack to another virtual machine present in the same infrastructure.
The attacker (external or internal) gains control of VM instantiating, and launches TCP SYN Flood attack on other VM instance running on the same physical machine.

Using the 'nmap' tool, the attacker VM performs the scan to know about the other virtual machines IP addresses present in the network. The attacker VM picks coexisting VM as a victim of TCP SYN flood attack. The attacker VM scans the victim VM to check for open TCP ports to perform the attack with 'nmap'. Then, the attacker VM initiates a TCP connection by sending SYN packets and the victim VM replies with the SYN-ACK packet. The attacker doesn't send the final acknowledgement to complete the three-way handshake. At the victim VM, high number of half-opened connections is left. The queue that is storing the half-opened connections is of finite size and it is made to overflow by intentionally creating too many half-open connections. The victim keeps on waiting for the final ACK packet and after the round-trip time expires, it resends SYN-ACK packets to the attacker. The victim VM is not able to further create new TCP sessions for the legitimate network traffic [1].

IV. DDoS ATTACK ON SOFTWARE-AS-A-SERVICE

To affect SaaS service, application layer DDoS attacks are implemented which simulate real user traffic and overwhelm the network bandwidth of SaaS applications.

XML based DDoS attack can be used to harm SaaS application in which instead of flooding the network with packets, it is flooded with XML messages. The XML message is sent to the target with a multitude of digital signatures and a parser would try to resolve it by using all CPU cycles, thus eating all the resources. One example of an X - DDoS attack is a Coercive Parsing attack that manipulates the web service request. It uses a continuous sequence of open tags so that the CPU usage on web server is exhausted.

Slowloris is also another DDoS attack, enabling one web server to take down another server, without affecting other services or ports on the target network. Slowloris does this by holding as many connections to the target web server open for as long as possible. It accomplishes this by creating connections to the target server, but sending only a partial request. Slowloris constantly sends more HTTP headers, but never completes a request. The targeted server keeps each of these false connections open. This eventually overflows the maximum concurrent connection pool, and leads to denial of additional connections from legitimate clients.

V. DDoS ATTACK ON PLATFORM-AS-A-SERVICE

The attackers make use of amplification attacks like Network Time Protocol (NTP) amplification, Domain Name Server (DNS) amplification to target server instances running software with known vulnerabilities in PaaS service.

An NTP amplification attack begins with a server controlled by an attacker on a network that allows source IP address spoofing. The attacker generates a large number of UDP packets spoofing the source IP address to make it appear the packets are coming from the intended target. These UDP packets are sent to Network Time Protocol servers (port 123) that support the MONLIST command. This command returns a list of up to the last 600 IP addresses that last accessed the NTP server. If an NTP server has its list fully populated, the response to a MONLIST request will be 206 times larger than the request. On the attack, since the source IP address is spoofed and UDP does not require a handshake, the amplified response is sent to the intended target. Thus, an attacker with a 1 Gbps connection can theoretically generate more than 200 Gbps of DDoS traffic.

A Domain Name Server (DNS) Amplification attack is a popular form of Distributed Denial of Service (DDoS), in which attackers use publicly accessible open DNS servers to flood a target system with DNS response traffic. The primary technique consists of an attacker sending a DNS name lookup request to an open DNS server with the source address spoofed to be the target's address. When the DNS server sends the DNS record response, it is sent instead to the target. Attackers will typically submit a request for as much zone information as possible to maximize the amplification effect. In most attacks of this type, the spoofed queries sent by the attacker are of the type, "ANY," which returns all known information about a DNS zone in a single request. Because the size of the response is considerably larger than the request, the attacker is able to increase the amount of traffic directed at the victim. By leveraging a botnet to produce a large number of spoofed DNS queries, an attacker can create an immense amount of traffic with little effort. Additionally, because the responses are legitimate data coming from valid servers, it is extremely difficult to prevent these types of attacks.

VI. DDoS DETECTION MECHANISMS IN CLOUD ENVIRONMENT

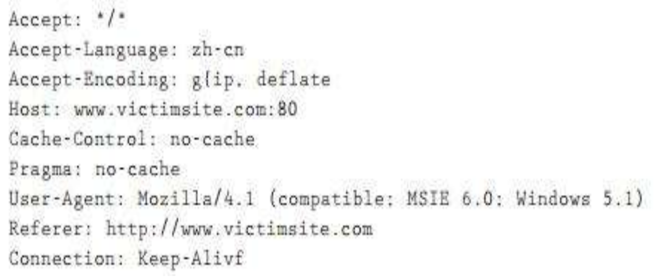
There are some traditional ways to prevent DDoS attack like using Intrusion Detection System (IDS), Web Application Firewall (WAF), ensuring there is an additional bandwidth available, updating patches etc.

✓ **IDS and Web Application Firewall:** To prevent DDoS, IDS is equipped with connection verification methods.

Web Application Firewall is another solution to detect DDoS traffic that originates from botnet. WAF detects botnet traffic by monitoring header signature of HTTP request.

According to [2] , the HTTP header of web request generated by botnet differs from normal HTTP header.

The header field in HTTP GET request looks as follows:



```
Accept: */*
Accept-Language: zh-cn
Accept-Encoding: g{ip. deflate
Host: www.victimsite.com:80
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Mozilla/4.1 (compatible: MSIE 6.0: Windows 5.1)
Referer: http://www.victimsite.com
Connection: Keep-Alivf
```

After the analysis of this header, following points can be noticed:

i. The HTTP header information generated by Botnet is often incomplete, even wrong or misspelled.

   Here, the "Connection: Keep-Alivf" header is misspelled.

ii. The header values like "User Agent", "Accept-Language" are present as hard coded strings in bot executable which suggests that they are consistently present in the HTTP GET requests generated by this family.
iii. Furthermore, the bot has been observed to use the host name of the victim Web site for the "Referer" header field.
iv. Sometimes, the bot request does not have valid user agent.
v. The botnet keeps on rotating IP's within a few seconds of each other, rotating referrers and user agents, all the while performing search requests.

Thus, by analyzing HTTP header signature of incoming HTTP request, it can be identified to be generated by botnet or human user. If the header mismatches, that request will be dropped else it will be forwarded further.

✓ **Modsecurity:** ModSecurity is another open source application layer firewall that adds intrusion-detection and prevention features to the web server. It is similar to IDS that analyses network traffic, but it works at the HTTP level. The attack prevention feature stands between the client and server; if it finds a malicious payload, it can reject the request, performing any one of a number of built-in actions [3].

The functionality of ModSecurity is divided into four main areas:

1. Parsing: Security-conscientious parsers extract bits of each request and/or response and store them for use in the rules.
2. Buffering: In a typical installation, both request and response bodies will be buffered so that the module usually sees complete requests (before they are passed to the application for processing), and complete responses (before they are sent to clients). Buffering is important, because it is the only way to provide reliable blocking.
3. Logging: Allows you to record complete HTTP traffic, logging all response/request headers and bodies.
4. Rule engine: Works on the data from the other components, to assess the transaction and act, as necessary.

Mod Security works on configuration and rules. The configuration instructs it how to process the data it sees and the rules decide what to do with the processed data. In the rule engine, gathered data is checked for any malicious or particular content.

Mod Security canbe deployed in two modes: embedded mode wherein Mod Security is added as a module into web server. However, in this mode, It is not able to inspect the content of server headers.

Another mode is network gateway in which ModSecurity is installed as a reverse proxy. This gives a single point to monitor, higher speed, high anonymity of the internal network, and Mod Security can inspect the server header of the backend database [4][5] .

Some of other features of ModSecurity are audit logging, access to any part of the request and the response, a flexible regular expression-based rule engine, file-upload interception, real-time validation and also buffer-overflow protection.

Thus, ModSecurity provides protection from a range of attacks against web applications and allows for HTTP traffic monitoring and real-time analysis with little or no changes to existing infrastructure.
Apart from the conventional ways of preventing and detecting DDoS attack as discussed above, following is the research work currently done in this area by the researchers:

- **Network Function Virtualization (NFV):** NFV is a new networking service model in which network functions are deployed on virtual machines as software instances instead of dedicated physical appliance. In a two-stage DDoS mitigation framework is proposed that leverages NFV. The first stage in the framework is called traffic screening stage that monitors and analyzes traffic flow. Depending on the result of analysis, the traffic is forwarded to the next stage of processing. The security is applied at network layer or application layer according to the screening of the traffic.

- **Completely Automated DDoS Attack Mitigation Platform (CAAMP):** It is software defined DDoS mitigation mechanism that makes use of public and private cloud [7]. When suspicious traffic is detected, it is redirected to a copy of original application called as shark tank. Shark tank is created on private cloud and works in an isolated environment with limited amount of resources. A virtual network is created that hosts the shark tank. The virtual switches are programmed by Software Defined Networking (SDN) controller that redirects the suspicious traffic. The shark tank is equipped with DDoS sensors that detect the presence of DDoS threat. If the traffic is proven to be non-suspicious, then it is redirected back to original application with the help of virtual switches.

- **Use of Data Mining:** Konstantin Borisenko et.al [8] have proposed DDoS attack detection model on Openstack using data mining techniques like k-Nearest Neighbors (k-NN), Decision Tree (DT), Support Vector Machine (SVM) and Naïve Bayes (NB). The proposed framework identifies three types of DDoS attack i.e. SYN flood, HTTP flood, and NTP flood. The architecture has components as collector, analyzer and counteraction. The collector gathers all incoming data and forwards it to analyzer. The analyzer component consists of two parts: data preprocessing and prediction. The prediction part employs various data mining methods to distinguish the traffic as malicious or benign. The features used by data mining techniques are amount of bytes, amount of packets, amount of unique pairs of source IP, address with port and destination IP address with port. If the traffic contains huge number of packets with small length, it is considered as SYN traffic. If the traffic contains large number of bytes and packets, it is identified as NTP flood. If the web server receives the requests from multiple IP addresses with different ports, then it is characterized as HTTP flood. The counteraction module uses the results generated by prediction module and feeds the respective commands to firewall to take counter action. It is observed from experimental results that decision tree performs better than other data mining techniques with False Positive Rate (FPR) of 0.05%, False Negative Rate (FNR) of 0.246%, and precision, recall, F-measure of 99.8%.

- **Information Theoretic Entropy and Random Forest:** Mohamed Idhammad et. al. has proposed slow rate HTTP DDoS detection mechanism based on information theoretic entropy and random forest (RF) algorithm

The authors have focused on three kind of slow rate attacks as Slow Read, Slow Message Body, and Slow Header. In Slow Read attack, the attacker establishes normal connection with server and sends HTTP GET or POST requests with receiver window size of zero bytes, telling the server that it is not yet prepared to receive the response. As a result, the server is forced to hold the connection and has to keep on waiting. In

Slow Message Body attack, the attacker builds a legitimate HTTP POST header with fake 'Content-Length' value. This value indicates the amount of data to be sent. The server receives the message header and waits to receive the message body according to the 'Content-Length' value mentioned in the header. However, the attacker breaks the message into small parts and sends it very slowly to the server. It causes the server to enter in waiting state. In Slow Header attack, the attacker sends incomplete HTTP GET requests to the server by not transmitting the two Carriage Return Line Feed (CRLF) that indicate the end of HTTP header. The victim server opens multiple connections and utilizes maximum resources to serve these requests. It results in denying legitimate requests. The authors have proposed to use source/destination IP addresses and ports as the features for DDoS detection. The proposed method estimates the entropy of these features using

Shannon's entropy formula and then normalize it to fit into given time window. The average value of entropy in current time window is evaluated. If it lies in the range of lower and upper threshold, the traffic is considered as legitimate else malicious. The classification of malicious traffic is done using machine learning algorithms like Random Forest (RF), DT, k-NN, NB, Multi-Layer Perceptron (MLP). It is found that RF performs better than other algorithms with 97% accuracy.

- **FlowTrApp:** SDN based framework, named FlowTrApp is proposed in [10] to protect data centers from DDoS attack. The proposed framework has utilized sFlow and OpenFlow for implementation of FlowTrApp algorithm. sFlow [11] is SDN tool used to gather flow statistics from routers and switches. OpenFlow [12] is SDN communication standard used to facilitate network engineering and security solutions. The proposed framework has considered two features; flow rate and flow duration to distinguish the traffic as legitimate or malicious. The Flow Traffic Tuple (FTT) is defined that represents the threshold values for flow rate and flow duration. This tuple is decided by doing manual survey of legitimate users. The incoming traffic is matched with FTT. Depending on the characteristics of incoming traffic, it is categorized as High Rate Spike, Short Lived High Rate, Long Lived High Rate, Idle User, or Long-Lived Low Rate attack flow. If the traffic is found to fall under any above-mentioned category, a mitigation mechanism is launched which blocks the user if it is found attempting to send the traffic flow more than the legitimate value else the flow rate is limited for that user to ensure congestion free traffic.

- **Meta-data Analysis:** Charles Tang et.al [13] has proposed to detect HTTP GET flooding attack by analyzing meta-data of HTTP connection that considers majorly two parameters; IP address and Uniform Resource Locator (URL). An intelligent probe (iProbe) is used to parse the HTTP traffic that extracts source/destination IP address, port and URL of every HTTP connection. Then, NFDUMP API is used that filters meta-data depending on threshold number of HTTP GET requests, does aggregation, sorts results, and computes statistics in real-time for big data analysis. In order to mitigate DDoS attack, the IP addresses of attackers are sent to firewall for rate limiting. The experimental results show that the proposed technique is able to detect HTTP GET flood attack with 100% accuracy if HTTP GET requests threshold is set to 15 requests/second.

VII. Analysis Of Existing Approaches

A review carried out about existing approaches of DDoS detection mechanism shows that there are still opportunities to carry out further research work.

Table 1 represents the analysis of existing techniques along with the limitations which can lead to further research.

Table 1. Analysis of Existing Approaches

Sr. No.	Approach	Advantages	Limitations
1.	Network Function Virtualization	The approach is able to detect all types of DDoS attack	The approach is limited to one screening mechanism
2.	Completely Automated DDoS Attack Mitigation Platform	Lowers False Positive Rate	The network statistics are not used by SDN controller
3.	Use of Data Mining techniques	High accuracy and small false positive and false negative rate with high F-measure	The results need to be studies on more complicated data
4.	Information Theoretic Entropy and Random Forest	Higher accuracy	The results are limited to one HTTP DDoS tool
5.	FlowTrApp	The approach utilizes sFlow and OpenFlow effectively and performs better than existing Quality of Service (QoS) approaches	The approach is limited to the detection of UDP flood and ICMP flood attack
6.	Meta-data Analysis	100% accuracy with recognition rate of HTTP GET flood attack at 15 requests/second	The approach is limited to detection of HTTP GET flood attack

VIII. Conclusion

Distributed Denial-of-service (DDoS) attack is considered one of the largest threats to the availability of cloud computing services which is used to deny access for legitimate users of an online service. This paper has presented various ways of performing DDoS attack on cloud service models; IaaS, SaaS, and PaaS. Further, the paper has discussed state-of-art approaches used for detecting DDoS attack. It is observed that most of the researchers have utilized SDN components for detection and mitigation. Also, many approaches have adopted machine learning algorithms like SVM, NB, DT, MLP, RF etc. to make detection process more intelligent and accurate. However, there is still a scope of research in this direction to design a more sophisticated system that can handle all types of DDoS attack.

References

[1]. Kanika and N. Sidhu, "Impact of Denial of Service Attack on the Virtualization in Cloud Computing," *Int. Conf. Commun. Comput. {&} Syst.*, pp. 56–62, 2014.

[2]. S. Khattak, N. R. Ramay, K. R. Khan, A. A. Syed, and S. A. Khayam, "A Taxonomy of botnet behavior, detection, and defense," *IEEE Commun. Surv. Tutorials*, vol. 16, no. 2, pp. 898–924, 2014.

[3]. I. Ristic, *ModSecurity Handbook*, 2nd ed. Feisty Duck, 2010.

[4]. grsk krish, "How to Secure an Apache Web Server - DZone Security," 2018. [Online]. Available: https://dzone.com/articles/how-to-secure-apache-web-server. [Accessed: 10-Aug-2018].

[5]. Falko Timme, "Secure Your Apache With mod_security." [Online]. Available: ttps://www.howtoforge.com/apache_mod_security. [Accessed: 10-Aug-2018].

[6]. T. Alharbi, A. Aljuhani, and H. Liu, "Holistic DDoS mitigation using NFV," *2017 IEEE 7th Annu. Comput. Commun. Work. Conf. CCWC 2017*, 2017.

[7]. N. Beigi-Mohammadi, C. Barna, M. Shtern, H. Khazaei, and M. Litoiu, "CAAMP: Completely automated DDoS attack mitigation platform in hybrid clouds," *2016 12th Int. Conf. Netw. Serv. Manag. CNSM 2016 Work. 3rd Int. Work. Manag. SDN NFV, ManSDN/NFV 2016, Int. Work. Green ICT Smart Networking, GISN 2016*, pp. 136–143, 2017.

[8]. K. Borisenko, A. Smirnov, and E. Novikova, "DDoS Attacks Detection in Cloud Computing Using Data Mining Techniques," vol. 9728, pp. 197–211, 2016.

[9]. M. Idhammad, K. Afdel, and M. Belouch, "Detection System of HTTP DDoS Attacks in a Cloud Environment Based on Information Theoretic Entropy and Random Forest," *Secur. Commun. Networks*, vol. 2018, 2018.

[10]. C. Buragohain and N. Medhi, "FlowTrApp: An SDN based architecture for DDoS attack detection and mitigation in data centers," *2016 3rd Int. Conf. Signal Process. Integr. Networks*, pp. 519–524, 2016.

[11]. J. Jerrim, "Benefits of flow analysis using sFlow[R]: network visibility, security and integrity: Lancope Inc. - Free Online Library," *Database Netw. J.*, 2013.

[12]. "Software-Defined Networking (SDN) Definition - Open Networking Foundation." [Online]. Available: https://www.opennetworking.org/sdn-definition/. [Accessed: 09-Aug-2018].

[13]. C. Tang, A. Tang, E. Lee, and L. Tao, "Mitigating HTTP flooding attacks with meta-data analysis," *Proc. - 2015 IEEE 17th Int. Conf. High Perform. Comput. Commun. 2015 IEEE 7th Int. Symp. Cybersp. Saf. Secur. 2015 IEEE 12th Int. Conf. Embed. Softw. Syst. H*, pp. 1406–1411, 2015.