Enhanced Mechanism to Detect and Mitigate Economic Denial of Sustainability (EDoS) Attack in Cloud Computing Environments

Parminder Singh Bawa, Shafiq Ul Rehman, Selvakumar Manickam
National Advanced IPv6 Centre (NAv6)
University of Science Malaysia
Penang, Malaysia

***Abstract*—Cloud computing (CC) is the next revolution in the Information and Communication Technology arena. CC is often provided as a service comparable to utility services such as electricity, water, and telecommunications. Cloud service providers (CSP) offers tailored CC services which are delivered as subscription-based services, in which customers pay based on the usage. Many organizations and service providers have started shifting from traditional server-cluster infrastructure to cloud-based infrastructure. Nevertheless, security is one of the main factors that inhibit the proliferation of cloud computing. The threat of Distributed Denial of Service (DDoS) attack continues to wreak havoc in these cloud infrastructures. In addition to DDoS attacks, a new form of attack known as Economic Denial of Sustainability (EDoS) attack has emerged in recent years. DDoS attack in conventional computing setup usually disrupts the service, which affects the client reputation, and results in financial loss. In CC environment, service disruption is very rare due to the auto-scalability (Elasticity), capability, and availability of service level agreements (SLA). However, auto scalability utilize more computing resources in event of a DDoS attack, exceeding the economic bounds for service delivery, thereby triggering EDoS for the organization targeted. Although EDoS attacks are small at the moment, it is expected to grow in the near future in tandem with the growth in cloud usage. There are few EDoS detection and mitigation techniques available but they have weaknesses and are not efficient in mitigating EDoS. Hence, an enhanced EDoS mitigation mechanism (EDoS-EMM) has been proposed. The aim of this mechanism is to provide a real-time detection and effective mitigation of EDoS attack.**

Keywords—Cloud computing; Economic Denial of Sustainability (EDoS) attack; security; Distributed Denial of Service (DDoS) attack; mitigation mechanism; anomaly detection technique

I. Introduction

Internet has become an integral part of our everyday routine. Technology has evolved rapidly especially around the field of Information and Communication Technology (ICT) whereby new platforms are being continuously introduced; leading to newer opportunities and challenges [1], [2]. Cloud computing (CC) is one of the latest revolution in ICT [3]. It is a model in which computing is delivered as any other commoditized service like electricity, water, and telecommunication. CC solutions are usually offered by Cloud Service providers (CSP) by providing customizable cloud service models such as *Infrastructure-as-a-Service, Platform-as-a-Service*, and *Software-as-a-Service* [4]. In fact, cloud spending was forecasted to touch $37 billion in 2016 alone [5].

There are abundant of security concerns for CC as it incorporates numerous distinct technologies including networks, systems, virtualization, scheduling, DBMS 6 management, load balancing, etc. [6]. Hence, security concerns for many of these systems and technologies are also applicable to CC. Security in the cloud is accomplished, in part, through third party utilities and assertion much like in old-fashioned outsourcing engagements [7]. However, as there is no collective CC security standard, there are additional challenges related with this. Cloud service providers tend to implement their own copyrighted standards and security technologies, and deploy divergent security models. Consequently, such tendencies call for qualities of each technology and system to be assessed individually. More of this discussion is presented in Section II.

One of the most common security threats to most devices and services connected on the Internet is the Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks [8]. With the advent of CC, a mechanism known as sPOW was proposed by Khor and Nakao in 2009, so that the impact of DDoS can be alleviated by continuously scaling up the amount of required resources, i.e., elasticity (bandwidth), of devices and/or services [9]. Although it is true that service availability can be assured for legitimate users, a newer issue arises because of the usage of elasticity for withstanding DDoS attacks. The new issue is about the high cost that needs to be paid by the client/user of the CC platform due to extra resources allocated due to resource saturation caused by the DDoS attack [9], [10]. Consequently, a new term has been coined to characterize this particular issue that happens in a CC environment: an Economic Denial of Sustainability (EDoS) attack [11].

With the introduction of infrastructure as a service (IaaS) models for cloud computing, commodity servers have become a necessity for the computing resource needed by such IaaS models [4]. Organizations can now save on Capex for infrastructure, licensing as well as on Opex cost for maintenance and support service of infrastructure. Instead of handling all the costs by themselves, they just need to pay for the bandwidth, storage, and computing power similar to the utility charge for water and electricity, i.e., pay-per-use. Since cloud infrastructures have an important auto-scaling feature, i.e., elasticity, compared to traditional computing infrastructures, they are less susceptible to flash flood and DDoS attack. However, the elastic nature of cloud computing can be used against the clients in a different form of attack, an EDoS attack. The intention of EDoS attacks is not to overkill and crash a server as that of traditional DDoS attack; instead, the objective of an EDoS attack is to consume cloud resources in such a manner as to affect the cloud hosting expenses to incur high cost on the victim's bills [11].

In the remainder part of this article, DDoS is asserted as a major cause of EDoS attack in CC environments. In addition, this article also investigates the existing techniques proposed to detect and mitigate DDoS (EDoS) attacks and their limitations in CC environments. Afterwards, the details of the proposed mechanism are described in details to mitigate it effectively.

II. Related Works

Most of the current literature available addresses mainly on DDoS protection emphasizing on techniques for preventing of apparently malicious traffic at the network or application layer. There is very limited number of literature that are available to provide deployable solutions specifically for mitigating EDoS attacks in Cloud Computing environment. Most of the researchers in the field of CC and network monitoring are relying on the predefined threshold and on entropy techniques to detect anomalies in network traffic. Some of the well-known EDoS defence techniques are as under.

A. *Self-verifying Proof of Work (sPOW)*

Khor and Nakao proposed a self-verifying proof of work (sPoW) [9]. This method employs an application layer mitigation mechanism. The main function of this mechanism is to filter the attack traffic before it starts overcommitting resources. The concept of self-verifying Proof of Work (sPoW) is introduced to transform the network level DDoS traffic to distinguish the EDoS attack, On-demand network filter and prioritize legitimate traffic. sPoW consists of two main activities: 1) converting network-level DDoS into traffic that can be distinguished and filtered by simple packet pattern matching, and 2) allowing the remaining legitimate traffic stream to pass through. The combination of both legitimate and application-level DDoS traffic then competes for server resources by solving self-verifying proof of work (sPoW). The first action discards network-level DDoS traffic before it activates the billing mechanism. The second action uses puzzle solving technique to allow genuine traffic to contend and reduce the aggregate of expensive cloud resources consumed on application-level DDoS.

sPoW is the solitary steadfast method to prevent EDoS in CC. Conversely, it also inherits a number of limitations. Firstly, asymmetric computational power consumption for the clients. Solving computational puzzles require more CPU power and suitable only to faster CPUs. Therefore, mobile devices with less processing power will not be able to resolve the puzzles, thus unable to access the cloud resources. Green et al. iterates the problem of computational disproportion when Graphics Processing Unit (GPU) is used by attacker to resolve the puzzles [12]. Secondly, the Server must create separate channels to address each request. In case of a large number of incoming requests, server will generate number of puzzles which leads to puzzle accumulation attack if puzzles do not resolve in time.

B. *Cloud Trace Back (CTB)*

Ashley Chonka and co-researchers proposed the Cloud Trace Back (CTB) and Cloud protector model [13]. CTB is built upon Deterministic Packet Marking (DPM) algorithm [14]. CTB is implemented on the edge routers in directive to be close to the source of the cloud network. In directive to use Cloud Trace Back Mark (CTM) tag in the CTB header, it is positioned in front of the web-server. Consequently, all service requests are initially forwarded to the CTB for marking, thus efficiently confiscating the service provider's address and averting a direct attack. If an attack is effective to bring the web-service down, the target server will recover and rebuild the CTM tag to disclose the identity of the target source. CTB requires Cloud Protector (CP) to eliminate a DDoS attack. CP acts as a filter engine. The CP is a self-learning back propagation Neural Network (NN), to support detection and filtration of DDoS [15]. A neural network is a set of connected units made up of input, hidden and output layers. In a neural network, the emphasis is on the Threshold Logic Unit (TLU). The TLU injects input objects into an array of prejudiced quantities and calculate to check and compare with the defined threshold values [16].

In 2012 VivinSandar and Shenai came up with the framework to address EDoS by confronting HTTP and XML based DDoS attack [17]. This framework is a combination of a firewall and challenge server. The challenge server directs the Graphic Turning Test (GTT) to the user and if the user solves the offered GTT, then user host is added to the whitelist of the firewall to allow future access of the user. On the contrary, if the user fails to resolve the test in case of automated tools or bot, then host will be added to firewall's blacklist and user access will be blocked in the future. This framework limits traffic from automated tools or bots, but it provides no protection in case attack is initiated from already whitelisted hosts (or spoofed). Furthermore, this method is very elementary in providing a firm protection against EDoS. Also, this method faces the same challenges with sPow in puzzle resolution and computing power requirements besides being prone to puzzle accumulation attack.

C. *EDoS-Shield*

Sqalli and co-researchers proposed a mitigation technique called EDoS-Shield [18]. The scheme differentiates between legitimate and malicious requests through verification of human presence at the end-user machine. The proposed architecture of the EDoS-Shield mitigation mechanism comprises of Virtual Firewall (VF) and Verifier Nodes that operates in tandem to perform the EDoS mitigation tasks. The firewall filters incoming requests based on two lists, namely: whitelist and blacklist. Whenever the client makes an initial access request, the verifier node verifies it through a Turing test. If the client passes the Turing test, its IP address will be included in the white list and subsequent requests from the same client are forwarded directly to the cloud scheduler, approving resource allocations. On the contrary, if a user fails the Turing test, its IP address will be held in the black list and subsequent requests from this user will be dropped by the front-end firewall itself. However, the proposed approach has a few shortcomings. Firstly, its vulnerability to IP address spoofing. An EDoS attack perpetrated by an attacker using a spoofed IP address belonging to the white list of the verifier node, would remain undetected. A second shortcoming is the high number of false positives identified through blocking of many IP addresses belonging to legitimate users, as the two lists are not updated in a timely and accurate manner.

An enhanced version of the EDoS-Shield was proposed in 2012 by Al-Haidari and co-researchers [19] wherein, a Time-To-Live (TTL) field is appended alongside the IP address of end-users requesting for cloud services. In this approach, the authors attempt to thwart the threat of spoofed IP addresses, as the distinctness in IP addresses when accompanied with a TTL field; will help differentiate malicious clients using spoofed addresses from legitimate ones. A similar scheme proposed by Chapade and co-researchers allows for classification of network traffic into legitimate and anomalous based on mean absolute variances of TTL values [20].

D. Scrubber Service

Naresh Kumar and co-researchers proposed In-cloud scrubber service for EDoS mitigation [21]. This method consists of on-demand EDoS mitigation web service (Scrubber Service). In-Cloud Scrubber spawns a service and validate the client side submission of a crypto puzzle [18]. The service provider can select between two modes: normal mode or suspected mode. When the service provider perceives that the web server is under normal situation, then it runs in normal mode. In suspected mode, the consumer/user resolves the spawned crypto puzzle through brute force method to attest its legitimacy for service access. Once the service provider observes the web server resource exhaustion beyond an acceptable limit and high bandwidth utilization, this could be considered as high level DDoS attacks. Thus, service provider enables its suspected mode and an On-demand call is directed to the Scrubber service to generate and verifies hard puzzle. If the service provider observes low-level DDoS attacks, i.e., the web server resource exhaustion level is within an acceptable limit with normal bandwidth utilization, the Scrubber service generates and verifies moderate puzzle.

E. EDoS Armor

In the year 2013, Masood proposed an EDoS mitigation framework for E-Commerce applications [22]. It is a two-fold solution with an admission control and a congestion control. This is a multi-dimensional protection system; firstly, when user initiates a session, the server sends a challenge to the user, it may be either a GTT or a cryptographic puzzle form. Once the user resolves the challenge, the request will get forwarded to admission control. If the user could not resolve, the session of the user will be dropped and the number of connections to the server will be limited for the user. This mechanism uses port hiding method to limit the users, as attack cannot be initiated in the absence of valid port number. In the next phase, user browsing behavior is monitored for continuous learning. If an anomalous behavior is observed, service priority for such users is reduced resulting in slow service response thereby mitigating application DDoS.

Although, there are some existing mechanisms to mitigate EDoS attacks as aforementioned. Nevertheless, these mechanisms possesses some constraints which limits their implementation in CC environments. Therefore, an enhanced mechanism is needed to counter EDoS attacks. Hence, we designed an effective mitigation mechanism to address the EDoS attacks in CC environments.

III. Proposed Mechanism

This section discusses the proposed EDoS Mitigation Mechanism (EMM) in Cloud Computing (CC) environments. This mechanism aims to deliver an enhancement over existing mitigation techniques that were discussed in Section II to minimize the damages caused by an EDoS attack. The design of the proposed mechanism, i.e., EDoS-EMM and its components are discussed in relevant subsection.

EDoS-EMM involves the amalgamation of three main but interconnected modules which are called: 1) *Data preparation*, 2) *Detection*, and 3) *Mitigation*. The first module, i.e., Data Preparation, is accountable for flow-based monitoring and data collection. The collected flow information is processed and segregated based on the type of protocol before the flows being summarized and passed to the next module. The second or the Detection module analyses the collected datagram packets and process them in real-time to extract information like source and destination IP, port number, and number of packets per second. This module is also responsible to allow dynamic threshold settings as well as anomaly detection. Finally, the Mitigation module is responsible for generation of alerts and mitigation of attacks. This module initiates the process of updating of rules on the network devices to take appropriate action like blocking network traffic originating from an IP address for specific period. The decisions are made through a decision engine that analyses the incoming traffic against a set of rules. Fig. 1 depicts the architecture of the proposed EDoS-EMM.

In the following subsections, all components within each of the mentioned modules are discussed in depth in terms of their functionalities.

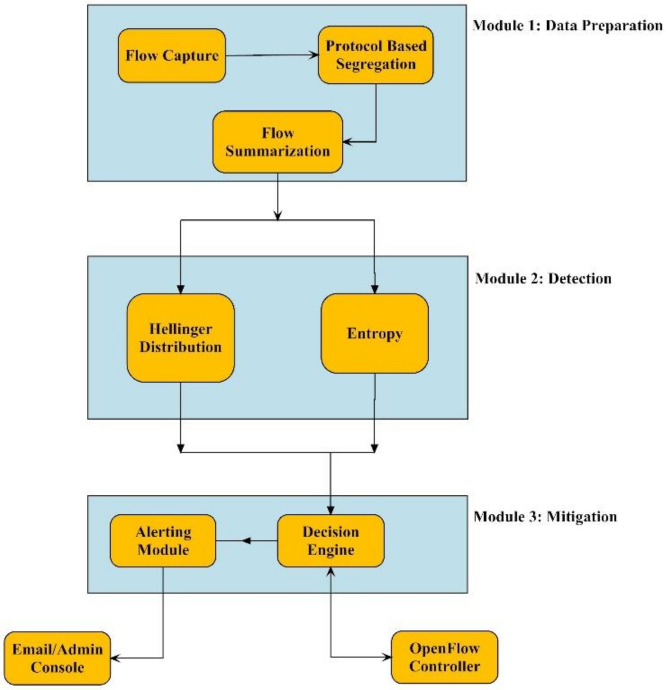


Fig. 1. Architecture design of EDoS-EMM.

A. Data Preparation Module

The data preparation module is responsible for data gathering, segregation, and normalization of flow information as shown in Fig. 2, which is essential to perform flow-based flooding detection using OpenFlow (OF) controllers [23]. This module collects flow information and periodically exports them to the protocol-based segregation component as mentioned in the following subsections.

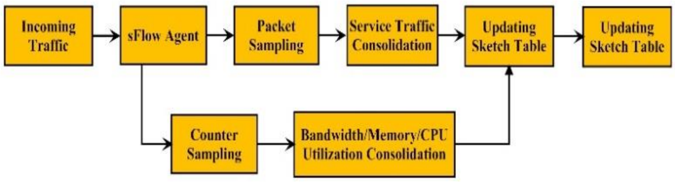


Fig. 2. Components within data preparation module.

1) Network Flow Collection

Network flows from the network switch will be collected using sFlow agent with a design as depicted in Fig. 3. The collected flows are then sent to sFlow collector for information extraction. To overcome the limitation with native approach as discussed in Section II, EDoS-EMM leverages on packet sampling technique provided by sFlow to monitor traffic in real-time, Packet sampling decouple the flow collection process form the forwarding plane and provide all flow-related statistical information. It collects the packet samples creating flow and update counters for every flow entry as controller application. This method provided efficient and aggregated packet forwarding, eliminating the specific flow entries requirement of native OF approach and overcome flow table size limitations by reducing the number of flow entries in OF switches.

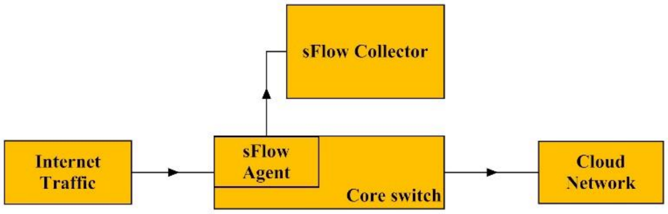


Fig. 3. Network flow collection process.

sFlow collector collects the updates of respective counters in monitoring module on a periodic basis, i.e., packet sampling, and hence eliminating the need to maintain and compare detailed flow information for each flow entry. Therefore, EDoS-EMM uses a simplified flow collection algorithm as shown in Fig. 4, to minimize system resource requirements and provides adequate information for a reliable attack detection process.

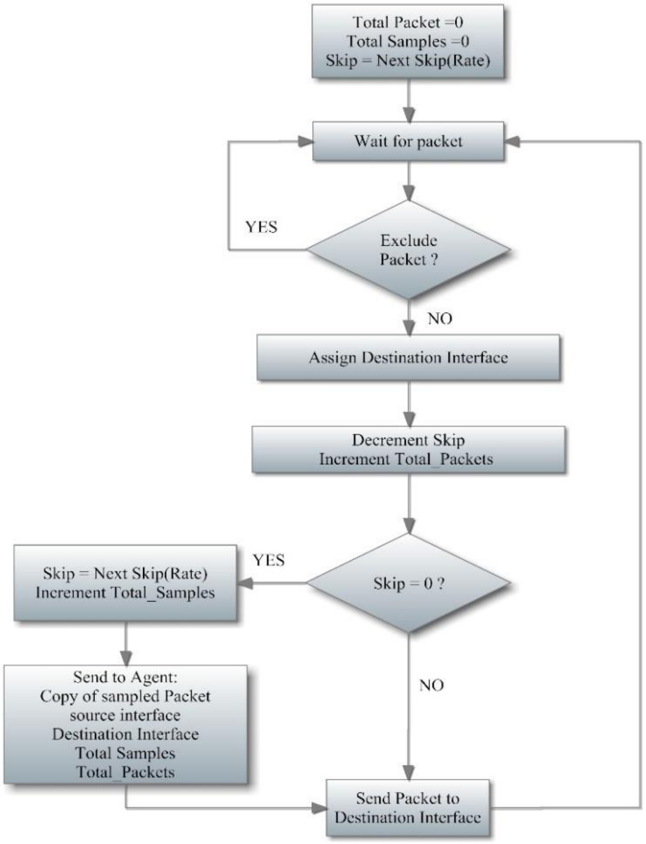


Fig. 4. sFlow agent algorithm.

Flow Sampling and *Counter Sampling* are two types of sampling techniques available for a sFlow agent [24]. Both sampling is independent of one another and recommended to use in conjunction. *Flow Sampling* collects statistics about a specific service whereas C*ounter Sampling* collects information about traffic on interface. *Flow Sampling* is based on sampling ratio, sFlow agent can parse sample packet information for incoming and outgoing packets of an interface. *Flow sampling* technique monitors the traffic details and parse behavior of network traffic. In *Counter Sampling* technique, sFlow agent gets periodic statistics on a monitored network interface. This technique focus only on the traffic statistics rather than traffic details on network interface.

2) Protocol Based Flow Segregation and Summarization

This component filters and segregates the flow information using 6-tuple information collected from 12 tuple information from sFlow datagrams. The extracted information is filtered based on the protocol or on service like *TCP*, *UDP*, and *ICMP* protocols with flow going towards a destination host IP. This component extracts the 6-tuple information like *switch ID, source IP, destination IP, source port, destination port,* and *counter* from the datagram as shown in Fig. 5. Extracted information is then further processed using sketch data structure [25], a probability data summary procedure, to randomly cumulative high dimensional data stream into small dimensions. The sketch data structure is a probabilistic data summary technique. It randomly aggregates high dimensional data streams into smaller dimensions.

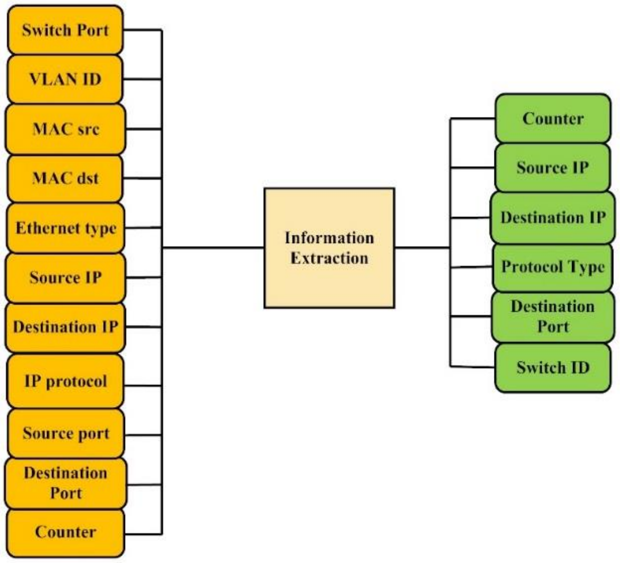


Fig. 5. Extraction of essential flow information by sFlow agent.

In this data structure, every data element $ai = (ki,\ vi)$ consist of key ki and its associated value vi in sketch data modelling. Whenever new packets arrived in network, its corresponding value gets added with the same key. The proposed EDoS-EMM utilizes *Source IP* (srcIP) as the key and the *number of packets per service protocol* (ICMP, UDP, HTTP) as the corresponding value as shown in Fig. 6.

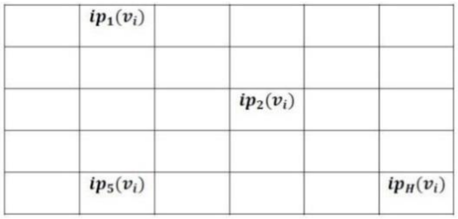


Fig. 6. Data feed within the sketch data structure.

This information will later be utilized by Module 2 as described in Subsection B to set a dynamic threshold using Hellinger Distance and Entropy based flood detection for alerting and mitigation of attack.

B. Detection Module

The Detection module has two independent components namely: *Threshold detection* and *Anomaly Detection*. Threshold calculation component relies on Hellinger Distance (HD) probability distribution whereas anomaly in traffic is detected using an entropy method. Output of the both module is correlated to confirm the attack in network and provided as input to Module 3 for generating alerts and perform the mitigation of attacks. A natural idea for flooding detection is to identify changes in traffic volume or rate. In such methods, alarms are raised if the traffic volume during a time interval is larger than a threshold predicted according to past normal conditions. A main issue of volume/rate monitoring is that the detection accuracy can be severely degraded if the normal rate is dynamic in the observation window due to the random nature and the flooding attack rate is not very high [26]. The Hellinger distance (HD) [27] which describes the deviation between two probability distributions, has been proposed as a detection method. The Hellinger distance is defined between vectors having only positive or zero elements. The HD mechanism has shown its strong capability to detect flooding attack because the low-rate flooding is likely to have different probability distributions from the normal traffic [27].

Sample flows collected by the sFlow agent will be processed using sFlow-RT and REST API's to control the packet flow in the network using an OpenFlow controller. The segregated traffic from Module 1 is fed as input to Module 2 for training data as depicted in Fig. 7.

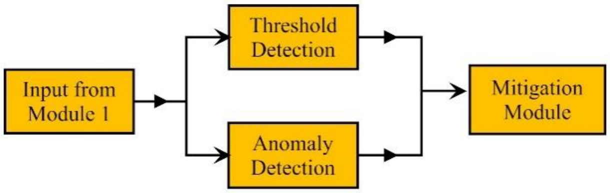


Fig. 7. Segregated traffic and data training processes.

1) Threshold Detection

To indicate the anomaly in the network, a detection threshold is required. To obtain a dynamic threshold that allows the proposed mechanism to be used in any kind of network environment, EDoS-EMM relies on the Hellinger Distance (HD) probability distribution method. HD is used to measure the distance between two probability distribution [28]. To compute HD, assume two distributions on same sample space are present, namely P:($p1$,$p2$,………,pn) and Q:($q1$,$q2$,………,qn). HD between two distributions can then be defined as:

$$H^2\ (P,Q\ ) = \frac{1}{2}\sum_i^n = 1\left(\sqrt{Pi} - \sqrt{Qi}\right)^2 \tag{1}$$

As it may be obvious, any benign changes observed in a monitored traffic pattern may indicate high HD values, resulting to higher false positives. To address this anomaly, EDoS-EMM adopts Exponential Weighted Moving Average (EWMA) method. A dynamic threshold will then be calculated as:

$$H_{n+1}= (1-\ \alpha).H_n + \alpha.h_n \quad (2)$$

$$\sigma_n = \ |H_n -\ \ h_n| \quad (3)$$

$$S_{n+1} \ = \ (1-\ \beta)\,.S_n \ + \ \beta\,.\sigma_n \quad (4)$$

$$H_{n+1}^{Threshold} \ = \ \lambda\,.H_{n+1} \ +\mu\,.\,S_{n+1} \quad (5)$$

Where h_n represent the current counter value in row one for the source IP in HD. H_n and H_{n+1} are the estimated average of current and upcoming HD. σ_n gauge the deviation of H_{n+1} from h_n. S_n and S_{n+1} denote current and subsequent mean deviation. EWMA by G.J. Ross is utilized to forecast the upcoming values based on current values [29]. On the basis of H_{n+1} and S_{n+1} the estimated threshold $H_{n+1}^{Threshold}$ is calculated where the recommended value of α and β is defined as 0.125 and 0.25, respectively [29]. Threshold should be defined as higher value than the HD in normal condition to prevent any false alarms. Therefore, the variables λ and μ help in defining a safe margin for each threshold value [30].

In case of a potential EDoS attack, the threshold will shift the probability distribution acquired from current sketched dataset. Thus the $HD1$ become larger than the threshold calculated and anomaly detection is enumerated. To safeguard the threshold in case of an attack, "estimation freezing" is also performed. In this procedure, the current training set is first frozen and the upcoming dataset is proceeded to be tested in next time interval. Thus, HD will be calculated between *frozen* dataset and the *upcoming* dataset. This "*one freezing one proceeding*" action will continue till the $HD1$ value drop below the current threshold value. The main motive is to keep the $HD1$ value high during attack. Secondly, the threshold is *frozen* to avoid being impacted by the attack, by not updating it until the $HD1$ drops below the defined threshold using the above-mentioned equations.

2) *Anomaly Detection*

Consolidated data from previous component are provided as input to attack detection module at regular interval of time. In the case of EDoS-EMM 20s time window is chosen, to achieve near real-time attack detection coherent with similar studies [31], [32].

For each time-window (20s), this component inspects all flow entries, revealing any anomaly in network flow and classifying a likely attacker or the victim of the attack. This architecture can integrate various algorithms especially statistical anomaly detection [33], machine learning-based anomaly detection [34] and data mining based anomaly detection [35] as presented in [36], [37]. In the proposed EDoS-EMM, an entropy-based algorithm [38] is adopted as the anomaly detection algorithm. This chosen algorithm not only effectively classifies attack patterns, but also distinguishes the attackers and the victims. Once network anomaly is detected, the algorithm examines and correlates definite network metrics identifying the attack and revealing all related information to the Attack Mitigation module.

Entropy-based detection method can be applied to monitor network abnormalities in any type of network topologies with diverse traffic characteristics for classification and detection of anomalies. Entropy measures the randomness of a unique data set. Higher and lower values of entropy signify dispersed and/or concentrated probability distributions, respectively. To ensure a metric neutral of the number of unique values of the data set, the entropy is normalized by dividing it with the highest entropy value of the data set, so that its values range in (0, 1). Note that the source IP address *(srcIP)*, the source port *(srcPort)*, the destination IP address *(dstIP)* and the destination port *(dstPort)* are the required feature for the traffic flow distributions. In case of an attack, the attack source generates a large number of flows, causing the source IP address to dominate in the flow distribution. Based on fluctuations in entropy, the algorithm can distinguish the anomaly in network using dynamic thresholds.

Shannon introduced entropy to measure the ambiguity of random variable in operational data [39], [40]. When applied to an information source, the entropy measures the information enclosed in a message and is inversely related to its probability of occurrence [41]. Due to this, the word "entropy" is also referred to as information entropy which is defined as the average amount of the information in certain event [42].

Suppose that there are a set of n events $\{a1,2,\dots\dots.,an\}$ whose probabilities of occurrence are $\{p1,p2,\dots\dots.,pn\}$ respectively. In a selection of the event, the information gain from an event ai on that particular selection is log $(1pi)$. For N selections, the occurrence of event ai is $(N*\ pi)$. Thus, the total information I obtained from N selection is:

$$I = \textstyle\sum_{i=1}^{n}(N * pi) * \log(\frac{1}{pi}) \quad (6)$$

Then entropy which is average information of an event is

$$Entropy = \frac{1}{N} = \left(\frac{1}{N}\right)\textstyle\sum_{i=1}^{n}(N * pi) * \log(\frac{1}{pi}) \quad (7)$$

$$Entropy = -K \ \ \textstyle\sum_{i=1}^{n}(pi) * \log(pi) \quad (8)$$

Where, K is a positive constant which is the choice of a unit of measurement [39]. From the equation of entropy, it has been shown that the more uniform a probability distribution is, the larger is its information entropy [43]. The entropy is said to be at its maximum when all the observed events have an equal probability *pi*, which signals the most uncertain situation [39]. In other words, an event which has higher entropy is less predictable based on the interpretation of entropy as an information measure [43].

C. *Mitigation Module*

This module is responsible for identification & mitigation of attack in the network. Input from Module 2 (threshold and anomaly detections module) is provided to decision engine where a dynamic threshold is used to detect the EDoS attack and entropy analysis is used to verify the existence of attack. Both feedbacks from Module 2 is correlated with the traffic statistics from the OpenFlow network Switch. Based on the correlation, the decision engine in tandem with the mitigation engine make decision to either drop the packet on network perimeter or report the anomaly to the network/client administrator. Fig. 8 depicts the components and process of Mitigation module.

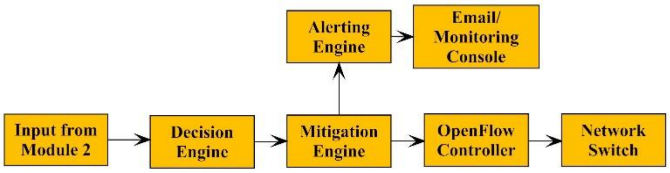


Fig. 8. Flow diagram of mitigation module.

1) *Decision Engine*

Decisions engine correlate the input from Module 2 as well as the statistics from the traffic flow through the OF Switch to compare and classify the anomaly in the network. For instance, Decision engine compares the defined threshold in Module 2 with the traffic flowing in the network along with its corresponding entropy value. If the network flow from an IP address to the client network, with the defined threshold exceeded and entropy value is 1, then it is classified as attack. Subsequently a request is forwarded to mitigation module to drop the network traffic originating from that IP address. Whereas, if the flow is less than the defined threshold and the entropy is lesser than 1, a network anomaly will get registered and an alert is raised as presented in Fig. 9.

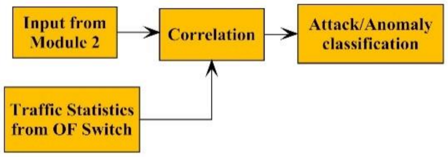
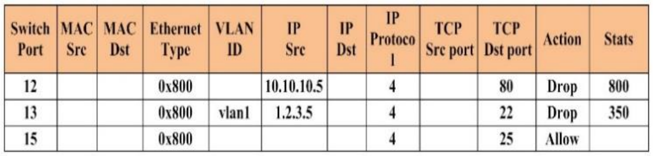

Input from Module 2
Correlation
Attack/Anomaly classification
Traffic Statistics from OF Switch

Fig. 9. Flow diagram of decision engine.

2) *Mitigation Engine*

Mitigation engine generates rule updates from the information gathered by Module 2. Attacker/anomaly-generating IP addresses gets identified along with the corresponding switch address. This engine creates rule to drop or block an IP address at the cloud's network switch. Once a rule is formulated as shown in Fig. 10, it is sent to OpenFlow controller to push it to the network switch to mitigate the ongoing attack. This engine also generates an additional message to send to user/client and Security operation Centre of the respective Cloud Service Provider (CSP) via E-mail/SMS or other methods.



Switch Port	MAC Src	MAC Dst	Ethernet Type	VLAN ID	IP Src	IP Dst	IP Protocol	TCP Src port	TCP Dst port	Action	Stats
12			0x800		10.10.10.5		4		80	Drop	800
13			0x800	vlan1	1.2.3.5		4		22	Drop	350
15			0x800				4		25	Allow	

Fig. 10. Flow updated rule sample.

3) *Alerting Engine*

Alerting engine allows EDoS-EMM to generate alert and update the client's cloud administrators via periodic email updates. In case of an anomaly detected in the network, this engine sends network updates using its periodic update cycle as defined by client. Whereas, in case of an attack, it immediately sends a Short Message Service (SMS) along with an E-mail alert to the client as well as CSP's Security Operation Centre (SOC) for notifying the ongoing attack.

IV. Conclusion and Future Work

In this paper, an enhanced EDoS mitigation mechanism has been presented. The proposed mechanism, i.e., EDoS-EMM is expected to address the limitations of existing EDoS techniques by providing real-time detection and mitigation of EDoS attack in cloud computing environments. The design of an EDoS-EMM mechanism was built on three modules approach, i.e., data preparation, detection, and mitigation modules. The purpose of this modular approach was to perform network flow processing, anomaly detection, and mitigation of an EDoS attack respectively. To refine the incoming network traffic sFlow agent algorithm has been proposed. Moreover, to achieve the high accuracy of anomaly detection Hellinger distance and entropy methods were incorporated. The future work will be to verify the effectiveness of EDoS-EMM based on its capability of handling the various scenarios whereby different types of attack traffic will be generated from various tools with random packet size and throughput. This includes HTTP and UDP attack traffic besides a flow of legitimate traffic (normal traffic).

Acknowledgment

This research was supported by EZY Infotech Inc., Philippines. In association with National Advanced IPv6 Centre (NAV6), Universiti Sains Malaysia (USM), Malaysia.

References

[1] T. Velte, A. Velte, and R. Elsenpeter, "Cloud Computing, A Practical Approach", McGraw-Hill, Inc., 2010.

[2] F. Gens, "New IDC IT cloud services survey: Top benefits and challenges", 2009.

[3] Adamov, and M. Erguvan, "The truth about cloud computing as new paradigm in IT", Paper presented at the International Conference on Application of Information and Communication Technologies, 2009.

[4] S. Bhardwaj, L. Jain, and S. Jain, "Cloud computing: A study of infrastructure as a service (IAAS)", International Journal of engineering and information Technology, vol. 2, pp. 60-63, 2010.

[5] Babcock, "Cloud Spending Will Top $37 Billion In 2016", Retrieved from http://www.informationweek.com/cloud/infrastructure-as-a-service/cloud-spending-will-top-$37-billion-in-2016-idc-reports/d/d-id/1326193, 2016.

[6] Zissis, and D. Lekkas, "Addressing cloud computing security issues", Future Generation Computer System, 28: 583-592. DOI: 10.1016/j.future.2010.12.006, 2012.

[7] K. Popovic, and Z. Hocenski, "Cloud computing security issues and challenges", Proceedings of the 33rd International Convention MIPRO, May 24-28, IEEE Xplore Press, Opatija, pp: 344-349, 2010.

[8] J. Nazario, "DDoS attack evolution. Network Security", vol. 7, pp. 7-10, 2008.

[9] S. H. Khor, and A. Nakao, "sPoW: On-demand cloud-based EDDoS mitigation mechanism", Paper presented at the HotDep (Fifth Workshop on Hot Topics in System Dependability, 2009.

[10] Hoff, "Cloud computing security: From DDoS (distributed denial of service) to EDoS (economic denial of sustainability)", Blog, Retrieved November, 27, 2008.

[11] P. Singh, S. Manickam, and S.U. Rehman, "A survey of mitigation techniques against Economic Denial of Sustainability (EDoS) attack on cloud computing architecture", in 3rd IEEE International Conference on Reliability, Infocom Technologies, and Optimization (ICRITO)(Trends and Future Directions), 2014.

[12] J. Green, J. Juen, O. Fatemieh, R. Shankesi, D. Jin, and C. A. Gunter, "Reconstructing Hash Reversal based Proof of Work Schemes", Paper presented at the LEET, 2011.

[13] Chonka, Y. Xiang, W. Zhou, and A. Bonti, “Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks”, Journal of Network and Computer Applications, vol. 34, pp. 1097-1107, 2011.

[14] Belenky, and N. Ansari, “On deterministic packet marking”, Computer Networks, vol. 51, pp. 2677-2700, 2007.

[15] Joshi, A. S. Vijayan, and B. K. Joshi, “Securing cloud computing environment against DDoS attacks”, In IEEE International Conference on Computer Communication and Informatics (ICCCI), January, 2012, pp. 1-5, 2012.

[16] S. I. Horikawa, T. Furuhashi, and Y. Uchikawa, “On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm”, IEEE transactions on Neural Networks, vol. 3, pp. 801-806, 1992.

[17] S. VivinSandar, and S. Shenai, “Economic denial of sustainability (EDoS) in cloud services using http and xml based DDoS attacks”, International Journal of Computer Applications, vol. 41, pp. 11-16, 2012.

[18] M. H. Sqalli, F. Al-Haidari, and K. Salah, “EDoS-shield-a two-steps mitigation technique against EDoS attacks in cloud computing”, Paper presented in Fourth IEEE International Conference on Utility and Cloud Computing (UCC), 2011.

[19] Al-Haidari, M. H. Sqalli, and K. Salah, “Enhanced EDoS -shield for mitigating EDoS attacks originating from spoofed IP addresses”. Paper presented at the IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2012.

[20] S. Chapade, K. Pandey, and D. Bhade, “Securing cloud servers against flooding based DDoS attacks”, Paper presented at the International Conference on Communication Systems and Network Technologies (CSNT), 2013.

[21] M. Naresh Kumar, P. Sujatha, V. Kalva, R. Nagori, A. K. Katukojwala, and M. Kumar, “Mitigating economic denial of sustainability (EDoS) in cloud computing using in-cloud scrubber service”. Paper presented at the Fourth International Conference on Computational Intelligence and Communication Networks (CICN), 2012.

[22] M. Masood, “A Cost Effective Economic Denial of Sustainability (EDoS) Attack Mitigation Framework for E-Commerce Applications in Cloud Environments”, 2013.

[23] Shalimov, D. Zuikov, D. Zimarina, V. Pashkov, and R. Smeliansky, “Advanced study of SDN/OpenFlow controllers”, In Proceedings of the 9th central & eastern European software engineering conference in Russia, October, 2013, ACM, p. 1, 2013.

[24] M. M. Hulboj, and R. E. Jurga, “Packet Sampling and Network Monitoring”, 2007.

[25] M. Thorup, and Y. Zhang, “Tabulation based 4-universal hashing with applications to second moment estimation”, Paper presented at the SODA, 2004.

[26] Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, “Sketch-based change detection: methods, evaluation, and applications”, In Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement, October, 2003, ACM, pp 234-247, 2003.

[27] H. Sengar, H. Wang, D. Wijesekera, and S. Jajodia, “Detecting VoIP floods using the Hellinger distance”, IEEE Transactions on Parallel and Distributed systems, vol. 19, pp. 794-805, 2008.

[28] L. Le Cam, and G. L. Yang, “Asymptotics in statistics: some basic concepts”, Springer Science & Business Media, 2012.

[29] J. Ross, “Parametric and nonparametric sequential change detection in R: The cpm package”, Journal of Statistical Software, vol. 78, 2013.

[30] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, “Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments”, Computer Networks, vol. 62, pp.122-136, 2014.

[31] Siaterlis, and V. Maglaris, “One step ahead to multisensor data fusion for DDoS detection”, Journal of Computer Security, vol. 13, pp. 779-806, 2005.

[32] M. Zhanikeev, and Y. Tanaka, “Anomaly identification based on flow analysis”, In IEEE Region 10 Conference Tencon, November 2006, pp. 1-4, 2006.

[33] S. Staniford, J. A. Hoagland, and J. M. McAlerney, “Practical automated detection of stealthy portscans”, Journal of Computer Security, vol. 10, pp. 105-136, 2002.

[34] T. Ahmed, B. Oreshkin, and M. Coates, “Machine learning approaches to network anomaly detection”, In Proceedings of the 2nd USENIX workshop on Tackling computer systems problems with machine learning techniques, USENIX Association, April 2007, pp. 1-6, 2007.

[35] S. Y. Wu, and E. Yen, “Data mining-based intrusion detectors”, Expert Systems with Applications, vol. 36, pp. 5605-5612, 2009.

[36] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, “Anomaly-based network intrusion detection: Techniques, systems and challenges”, Computers & Security, vol. 28, pp. 18-28, 2009.

[37] Patcha, and J. M. Park, “An overview of anomaly detection techniques: Existing solutions and latest technological trends”, Computer Networks, vol. 51, pp. 3448-3470, 2007.

[38] Lakhina, M. Crovella, and C. Diot, “Mining anomalies using traffic feature distributions”. In ACM SIGCOMM Computer Communication Review, August 2005, vol. 35, pp. 217-228, 2005.

[39] Shannon, “A note on the concept of entropy”, Bell System Tech. J, vol. 27, pp. 379-423, 1948.

[40] J. Harte, and E. A. Newman, “Maximum information entropy: a foundation for ecological theory”, Trends in ecology & evolution, vol. 29, pp. 384-389, 2014.

[41] N. R. Pal, and S. K. Pal, “Entropy: A new definition and its applications”. IEEE Transactions on Systems, Man, and cybernetics, vol. 21, pp. 1260-1270, 1991.

[42] S. Sterlacchini, C. Ballabio, J. Blahut, M. Masetti, and A. Sorichetta, “Spatial agreement of predicted patterns in landslide susceptibility maps”, Geomorphology, vol. 125, pp.51-61, 2011.

[43] T. Jaynes, “Information theory and statistical mechanics”, Physical review, vol. 106, p. 620, 1957.