Title: A cybersecurity framework to identify malicious edge device in fog computing and cloud-of-things environments

Author: Amandeep Singh Sohal, Rajinder Sandhu, Sandeep K. Sood, Victor Chang

PII: S0167-4048(17)30182-7
DOI: http://dx.doi.org/doi: 10.1016/j.cose.2017.08.016
Reference: COSE 1196

To appear in: *Computers & Security*

A Cybersecurity Framework to Identify Malicious Edge Device in Fog Computing and Cloud-of-Things Environments

Amandeep Singh Sohal[1], Rajinder Sandhu[1,3], Sandeep K. Sood[1], Victor Chang[2]
1. Guru Nanak Dev University, Regional Campus, Gurdaspur
Punjab, India.
2. IBSS, Xi'an Jiaotong-Liverpool University, Suzhou, China
3. Department of Computer Science and Engineering, Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, India.

jaissisohal@gmail.com, rajsandhu1989@gmail.com, san1198@gmail.com and victorchang.research@gmail.com

Abstract
Device security is one of the major challenges for successful implementation of Internet of Things and fog computing environment in current IT space. Researchers and Information Technology (IT) organizations have explored many solutions to protect systems from unauthenticated device attacks (known as outside device attacks). Fog computing uses network devices (e.g. router, switch and hub) for latency-aware processing of collected data using IoT. Then, identification of malicious edge device is one of the critical activity in data security of fog computing environment. Preventing attacks from malicious edge devices in fog environment is more difficult because they have certain granted privileges to use and process the data. In this paper, proposed cybersecurity framework uses three technologies which are Markov model, Intrusion Detection System (IDS) and Virtual Honeypot Device (VHD) to identify malicious edge device in fog computing environment. A two-stage hidden Markov model is used to effectively categorize edge devices in four different levels. VHD is designed to store and maintain log repository of all identified malicious devices which assists the system to defend itself from any unknown attacks in the future. Proposed cybersecurity framework is tested with real attacks in virtual environment created using Openstack and Microsoft Azure. Results indicated that proposed cybersecurity framework is successful in identifying the malicious device as well as reducing the false IDS alarm rate.

Keywords: Edge device, Two-Stage Markov Model, Fog Computing, Intrusion Detection System, Virtual Honeypot Device, Internet of Things.

1. Introduction

With the increase in smart applications and requirement to reduce the latency between the data generation and data processing stage, Cisco coined a new term called the fog computing. Fog computing enables smart applications to perform their processing on network devices which can be routers, gateways or switches rather than sending data to cloud datacenters [1]. It is not desirable to run smart application which are latency sensitive (i.e. their performance is affected by the latency of the network) on cloud computing datacenters for real-time and fast results [2]. An alternative for latency sensitive applications is to execute their processing on devices which are in closer proximity to the data generation node. Devices which have computational power and are in closer proximity to the data generation nodes are network devices such as routers, gateways and switches. Fog computing paradigm envisioned to use these network devices and create a datacenter for processing of latency aware smart applications. As compared with cloud computing, fog computing devices will provide less latency for the smart application however it is not possible to get computational power as compared to cloud computing.

Fig. 1 provides a layered architecture of fog computing environment with different devices involved in it. The lowest layer of this architecture contains the data generation devices which can be sensors, RFID tags, cameras, Internet of Things (IoT) devices and actuators. These devices will generate real-time data which should be processed by a smart application to make real-time decisions. Middle layer of the architecture consists of the network devices used to send the data from lower layer devices to the cloud computing infrastructure. This layer can act as a fog layer part or whole computation can be processed so that the application performance is not affected by the latency of the network. Devices in the fog layer are called edge device thereafter in the paper. The upper layer is the cloud layer which consists of virtual machines which can be provisioned from any of the cloud service provider. If fog layer does not have free computational resources then it can send the requests directly to the cloud infrastructure. Even if the processing is done at the fog layer the raw data, intermediate data and results are eventually stored on the cloud infrastructure [3].

In fog computing, for any instance of time same edge device can be used by multiple smart applications with different set of users which raises the issue of security of edge device. If edge device is hacked, it can get false input and provide false output data, formulate bad results which can affect the performance of whole application. Hacked device can also be used to share data and results with competitive companies or radical groups involved in unwanted activities. Security protocol which are used now days authenticate each edge device with the application before providing data or computation to perform. However, if authenticated device is hacked then situation became even more worse because that edge device has certain privileges inside the network. So, attack by edge devices on the fog computing environment can be broadly characterized into two main categories which are (a) Unauthenticated attacks and (b) Unauthorized attacks. If the edge device is unauthenticated and tries to attack the device layer or cloud layer then this attack is called unauthenticated attack or outside attack. But, if the edge device which attacks the fog layer or cloud layer is authenticated and it is inside the trusted network of application then it is called unauthorized or inside attack. Authenticated edge devices can easily cover their tracks because they have certain privilege to be in the network [4]. According to a survey conducted by the US state of cybercrime in 2013, 34% attacks performed on any system were insider attacks, 31% attacks were outside attacks and they were not able to classify rest 35% attacks as inside or outside attacks [5]. A different survey conducted by Furnell [4] stated that system security administrators are more aware and concerned about the outside attacks that most of the insider attacks goes undetected. As fog computing environment is a multitenant architecture and resources are shared among different applications, it is very difficult to identify the inside attacker. Severity of insider attack in fog computing environment is also very high because applications using real-time data are of high importance. For example, if an application process real-time meteorological data and predict flood or other natural calamity then result of false data feed by attacked edge device can be catastrophic. Therefore, there is immediate requirement of a proactive and predictive cybersecurity framework for protecting the fog services from malicious edge devices. Proactive prediction of malicious edge device attacks on the system will result in better and secure deployment of fog layer in Internet of Things environment.

Markov models are the stochastic model which satisfy the markov property which states that the probability of occurrence of any event in the future depends on the present state rather than the past states. Whereas, hidden markov models are the type of markov models where the intermediate states are not observed or hidden [6]. Markov model are used by many researcher for network security and assessment based on the activities of the current users [7]–[9]. Hidden markov models can be used to predict the probability of occurrence of malicious edge device in fog computing environment based on the activity of the device [10]. In proposed cybersecurity framework, two stage hidden markov model has been used to find the malicious device and shift it to Virtual Honeypot Cloud (VHD) based on its recent activities. Honeypot is a sweet trap set by the system administrator which is a decoy of real system to lure the attackers on the system [11]. Honeypot can be a computer, application, or data which can simulate the behavior of real system and log the attack path of attacker. Honeypot is different from Intrusion Detection System (IDS), Intrusion Detection and Prevention System (IDPS), and firewall in the sense that it allows user to attack a different version of real system [12]–[18]. Correct installation of honeypot can lead to more efficient and secure system but if honeypot is static and attacker knows its location then its values diminish to an extent. To make proposed cybersecurity framework more adaptive, a virtual honeypot cloud has been proposed in which location of honeypot is changed dynamically so that malicious edge device never know the exact location of it. VHD will prevent the identity of the honeypot as well as it will be easy to deploy as compared with traditional honeypots. VHD will be like a virtual machine image which can be easily deployed on any available hypervisor. Based on the level and severity of attacks, VHD can auto-scale its resource capacity which makes it more adaptive to any level and number of attacks by malicious edge device [19].

Malicious edge device attacks can be the major bottleneck in any fog computing environment and they can be more severe based on the importance of the smart application. The main objective of the study conducted in this paper is to design an effective framework for identifying malicious edge devices in fog computing environment, to shift working of malicious edge device to the VHD and to make fog layer more secure and adaptive in nature.

To achieve the objectives as stated above, a cybersecurity framework on top of fog computing is proposed which consists of three phases for mitigating the malicious edge device. In the first phase, all the edge devices running their services with IoT are under the surveillance of Secure Load Balancer (SLB)'s IDS. SLB consists of two modules which are IDS and two-stage Markov model. In the second phase, SLB categorized the edge device using Two-Stage Markov model and provides a probability of shifting the edge device on to the VHD. The two-stage Markov model module of the SLB also helps to reduce the false alarm rate of IDS. In the third phase, VHD monitors malicious edge devices to generate the log files of the attacker activities and save them in the attack database repository. Attack database repository is used by the cybersecurity framework to prevent unknown attacks in the future and make system more adaptive. The proposed cybersecurity framework is also capable of reverting back the legitimate edge device from the VHD based on probability generated by Markov model.

This paper is written around the different Sections. Section 2 summarizes the related work. In the Section 3, the two-stage hidden Markov model is explained. Section 4 proposes the cybersecurity framework for identifying malicious edge devices and mitigating insider user attacks. Additionally, VHD and its working is explained. Section 5 describes the experimental setup and result discussion of the proposed cybersecurity framework. Section 6 sums up the paper with future work.

2. Related Work

System analysts are aware of threat of insider users on the system for a while now. Many studies surveyed the definitions, cause and effects of a malicious insider user. But, to the best of our knowledge malicious edge device detection in fog computing, it is still in its very initial stages. Hence, in this section after studying some relevant literature of fog computing, more comprehensive studies of insider user attacks on systems will be undertaken.

In 2012, Bonomi et al. [2] coined the term fog computing in the paper published by Cisco. They explained the basic architecture of fog computing and how it extends the cloud computing paradigm. They also explained its relationship with Internet of Things. In 2014, Stojmenovic and Wen [3] studied the major scenarios and security issues that will be faced by fog computing paradigm. They also analyzed the real-time application of fog in smart grid and traffic situations. In 2015, Luan et al. [20] carried out an extensive study about the relationship of mobile cloud computing and fog computing. He also argued that fog computing is the perfect solution for running latency sensitive applications from mobile to cloud computing architecture.

In 2005, Ning and Sun [21] categorized the insider attack into two categories which are atomic misuse and compound misuse. They also discussed the effect of different insider user attacks in detail. In 2005, Chinchani et al. [22] described the definition of the insider attacks and intruders. They also argued that there are no sufficient tools which can protect the system from insider attacks. In 2005, Magklaras and Furnell [23] stated that out of many properties of malicious users, the level of sophistication property is notable because user sophistication and the potential to harm the system are more interrelated. In 2008, Salem et al. [24] surveyed on the solution of the insider attacks. They distinguished masquerades and traitors as well as discussed the insider user. They also highlighted some of the notable solutions for the insider user attacks. Similarly in 2008, Predd et al. [25] also conducted a survey on computer crimes & security and found that 59% attacks were launched by the insider's users. Further in 2009, Warkentin and Willison [26] surveyed the security risks of any information system and categorized broadly them as external and internal. They also concluded that there is more protection for external security risks than the internal security risks. In 2008, Martinez-Moyano et al. [27] described the behavioral theory of the insider threat. They presented a case study to explain the dynamics observation in data for studying the insider user attacks. In 2009, Colwilll [28] discussed that how malicious insiders effect an organization. He stated that insider misuse their privileges because they know how to achieve valuable assets while leaving little evidence. In 2014, Mundie et al. [29] conducted a study by interviewing thirty industry experts about the use case of insider user attack. Study stated that insider attack severity depends on sensitivity of data and rules of an organization.

The initial work in preventing information systems from insider user attacks were effective profiling of entire insider users. In 1998, Magklaras and Furnell [30] argued that there are no effective solutions for the problem of internal IT misuse. They estimated the level of threat of insider by using threat evaluation system based on particular profile of the insider user. In 2002, Schultz [31] proposed to detect an insider attack using behavior and specific activity indicators of any insider user. Specific indicators include deliberate marking, meaningful error generation, personal traits and correlated usage patterns. This system can find insider attacker by combining time stamp value against these indicators. In 2005, Theoharidou et al. [32] studied recent cyber criminology theories to understand various insider threats. They concluded that multi-paradigm and multi-disciplinary approach can mitigate the insider threat. In 2009, Bowen et al. [33] designed monitors which can record activities of all insider users and detect the malicious insider using a profile matcher. In 2010, Pfleeeger et al. [34] proposed a framework which categorized the insider users on the basis of the organization, system, individual and environment. They gave taxonomy for understanding different types of insider users. In 2014, Costa et al. [35] tried to bridge the gap between natural language description of any malicious insider user and his/her technical activities. They use semantic ontology approach to detect the malicious insider user from large pool of users. In 2014, Zhang et al. [36] proposed a framework which uses user's current behavior, his/her past behavior and his/her community behavior to detect anomaly in any user activity. This system removes false anomaly detection and helps to improve performance of the system. In 2015, Bondada and Bhanu [37] analyzed pattern key strokes of all the users for effective profiling of insider user. Based on the profile output, effective security mechanism can be imposed hence preventing insider user attack.

Many methods were also proposed by researchers for mitigating insider user attacks. In 2005, Ray and Poolsapassit [38] proposed a framework which used the attack tree to find the intruders from the insiders. In 2005, Dent [39] proposed a theory for constructing signcryption schemes with insider security based on hybrid cryptography. In 2007, Liu et al. [40] proposed a light, effective and flexible algorithm for the detection of insider attack in wireless sensor networks. In 2008, Liu et al. [41] proposed an "insider game" which is a game-theoretic model for predicting malicious insider user activities. Insider game was built on the basis of stochastic processes and game theoretic model. In 2008, Greitzer et al. [42] studied the research conducted by researchers at Pacific Northwest nation laboratory. They had developed a complex, cognitive-based instruction set to train the organization against the insider threat. In 2010, Probst et al. [43] proposed an access based model for defining high value insider in the system and then imposing high security on those particular users. In 2015, Sriram et al. [44] proposed a hybrid approach for preventing insider user attack in cloud by using multiple technologies. They used selective encryption for data encryption, neural network to create profile of users and decoy technology for concealing sensitive information.

Work related to intrusion detection system in the cloud computing is also briefly discussed. Characteristics of ideal intrusion detection system and cloud based intrusion detection system are also reviewed. In 2004, Furnel [4] discussed the insider user attack in an IT organization. He categorized the insider attackers into three types: masqueraders, clandestine users and misfeasors. He argued that the installation of Misuse Based Intrusion Detection System and Anomaly Based Intrusion Detection System can mitigate the insider user attacks. Though, he did not propose any model to protect the cloud from insider attacks. In 2007, Artail et al. [45] proposed a model to improve the functionality of IDS with the help of honeypot. The main concept of the model is to deploy a honeypot on IDS which interacts with malicious activities and save all log files of the malicious activities in the data base repository. In 2009, Sardana and Joshi [46] proposed an auto-responsive framework which uses an auto-responsive honeypot to defend from Distributed Denial of Service (DDoS) attacks. They used three line of defense from DDoS attack. In 2011, Chonka et al. [47] discussed that Denial of Service (DoS) attack is most dangerous attack on any cloud services. XML-DoS and HTTP-DoS attacks were the simplest types of DoS attacks which can be easily launched and are difficult to detect by security systems. Their framework used a neural network called cloud protector which can detect and prevent the attacks. Though, their solution is not capable of preventing the HTTP-DoS attack. In 2015, Zhang et al. [19] discussed about VHP. VHP captures data, control data, and analyze data. The cost of virtual honeypot was less and it was easy to maintain. It enhances the existing features of the traditional honeypot. In 2013, Modi et al. [48] argued that firewall is not sufficient to provide security to the data stored in the cloud system. They said that the IDS and Intrusion Prevention System (IPS) can be able to mitigate the insider attacks. They discussed various types of IDS and IPS in their paper and also concluded that IDS is the best option against the insider attack. In 2013, Patel et al. [49] mentioned that simple IDS is not capable to prevent the insider attacks because it produces large number of false alarms. To reduce the false alarm rate of any IDS, they used fuzzy logic based risk manager. In their work, they did not propose any model to defend the insider attack and no mathematical implementation is given for the concepts that they have proposed. Based on the concept, we proposed a model which uses Markov model based adaptive system for detection of attacks. Table 1 provides comparison of different IDS mentioned in the literature. Proposed framework combined the concept of multiple papers and created a framework which contains IDS and honeypot concepts combined together to form a seamless single framework. Proposed framework also introduces the virtual honeypot which further enhances the work of [46] and [49] whereas work in [19] is only centered on VHD and further enhanced by using IDS. Proposed framework targets to create a system in which different IDS and honeypots can be deployed easily as per the requirement of the application or the user.

Insiders' attacks are becoming more and more threatening than the outsider attacks because insider always enjoy certain privileges of the system. Finding the intruders in the distributed cloud environment is the biggest challenge for the cloud service providers. In the current scenario, there is no efficient security mechanism in the cloud computing with respect to insider attacks. IDSs are used for the insider attacks but it produces colossal number of false alarms those decreases overall efficiency of cloud. In the distributed cloud environment, the insider attacks and

IDS's false alarms rate are bottleneck in the successful implementation of cloud security. To defend insider attacks and handles IDS's false alarms, a cybersecurity framework is proposed which is adaptive in nature and even capable of reverting back the legitimate user from the VHC to real cloud if it is shifted accidently due to false alarm.

3. Markov Models

Real world formulates significant outputs which can be used to predict future. Markov model predicts the probability of the future outcomes on the basis of present outcomes of system rather than past activities. Hidden Markov model is used to enhance the capability of the Markov model [6]. This model is capable to predict the next state of the system with hidden state which was not possible with Markov model. The proposed cybersecurity framework uses Hidden Markov model to detect the future behavior of the edge devices. It also decides whether to shift edge device on VHD or not at any instant of time as there could be a probability that a legitimate edge device can be hacked at any time with or without help of other edge device. The Two Stage Markov Model of the proposed cybersecurity framework analyzes every edge device when an IDS generates an attack alarm towards the working of edge device. Terminologies of Markov model are discussed in Table 2.

3.1 Two Stage Markov Model

Applying traditional IDS will produces lots of false alarm because multiple IoT devices will send different types of data to the edge device for computation. Hence, our cybersecurity framework should also reduce the false alarm rate of traditional IDS in order to make proposed framework more effective. When the IDS will detect a malicious activity on the edge device, it will generate the attack alarm and sends the edge device identification and type of attack to Two Stage Markov model. Two Stage Markov model works in two stages to detect the malicious edge device. In first stage (Markov1), the edge device is categorized with help of Markov model and its Shifting Probability (SP) is calculated. These outputs are sent to the second stage (Markov2). In second stage, there is a Hidden Markov model that predicts whether the edge device has to be shifted to VHD or not on the bases of the SP and type of attack of the edge device. In a same way, legitimate edge devices are reverted back from VHD, who are shifted on it due to any continuous false alarms of the IDS.

Shifting of an edge device depends upon its SP or Legitimate Device Probability (LDP) which can be calculated by Markov1 or Markov3 respectively. These values are predicted with the help of viterbi algorithm of the proposed cybersecurity framework [50]. The general transition graph is used by the Markov1 or Markov3 and Markov2 or Markov4 is showed in the Fig. 2(a) and Fig. 2(b) respectively. Due to different behavior of edge devices, the transition graph of each edge device will be different and these transition graphs are updated after each activity of the edge device by viterbi algorithm. Viterbi algorithm in hidden Markov model computes the probability of all hidden states based on sequence of activities or emissions performed by the user. Algorithm 1 explains the step of viterbi algorithm for identification of malicious edge devices based on the sequence of requests generated by them. viterbi algorithm is an example of dynamic programing which uses the same schema as the forwardBackward algorithm. Viterbi algorithm is different from forwardBackward algorithm in two major cases which are Viterbi uses the maximization function in place of summation as done in forwardBackward algorithm. Viterbi algorithm stores value of maximization function at each instant of time in a matrix which is read in backtracking so that best sequence is selected.

Let $\delta_t(i)$ provides the maximum probability of edge device to end in state i with request sequence of length t which will result in first t observations of the hidden Markov model. We can define $\delta_t(i)$ as follows

$$\delta_t(i) = \max\{P(q(1), q(2), \dots\dots, q(t-1); O(1), O(2), \dots\dots, O(t) | q(t) = q_i)\}$$

Notions used for the implementation of the viterbi algorithms are discussed below.

$\delta_i(t)$	probability of edge device to end in state i based on input requests of length t.
p_i	is the initial state probabilities of state i.
$b_i(O(t))$	is the probability that output is *O(t)* if the initial state is i.
a_{ij}	is the transition probability from state i to state j.

Algorithm 1: Calculating SP or LUP of edge device

```
Viterbi
Begin of procedure:
Step I:  Initialization of probability variable and matrix for storing arguments
        $\delta_1(i) = p_i b_i(O(1))$
        $\psi_1(i) = 0$
Step II: Recursion is done for calculating likelihood for all the arguments of requests submitted by the edge
device and output of likelihood is updated in the $\psi$.
        $\delta_t(j) = max_i[\delta_{t-1}(i)a_{ij}]b_j(O(t))$
        $\psi_t(j) = \arg max_i[\delta_{t-1}(i)a_{ij}]$
Step III: Termination of the recursion step based on following conditions
        $p^* = max_i[\delta_T(i)]$
        $q_T^* = \arg max_i[\delta_T(i)]$
Step IV: Backtracking the results to find the best state
        $q_t^* = \psi_{t+1}(q_{t+1}^*)$
End of procedure.
```

Algorithm 1 provides all the steps involved in calculation of output state based on input requests of the edge device. Step I provide the initialization of all the variables of the algorithm which are likelihood probability and stored matrix. Step II is an recursion step which calculates likelihood probability for each input provided by the edge device. It also stores the calculated likelihood into a matrix for all arguments. This recursion step is terminated based on the termination condition stated in Step III. After the termination condition is achieved, a backtracking step as shown in Step IV is conducted which finds the best suited output state of the edge device based on all the input requests provided by it.

4. The Proposed Cybersecurity Framework

In this section, the working of proposed cybersecurity framework and its various components is discussed in detail. In proposed framework, the edge devices will be categorized into four categories: Legitimate Device (LD), Sensitive Device (SD), Under-attack Device (UD) and Hacked Device (HD). LDs are those edge devices, which use their privileges in right manner and never tried to breach the security of the system. SDs are those edge devices, which sends very few false data points to the system. SDs may send false data unintentionally or may be under attack from some amateur hacker but these devices should be monitored carefully. Amateur hackers are not well trained but launches attacks which may have impact on the overall system such as bad password, forgot request too often or sending captcha request again. UDs are the devices which are under attack from some professional hacker and must be prevented immediately. Professional hackers are fully trained hackers and they launch high impact attacks on the system which can be of system level and can stop the overall application. HDs are the edge devices which IDS predicts are being hacked and are sending continuous false data to the system. These devices are kept alive but shifted to VHD so that path of attacker can be successfully predicted. Fig. 3 shows the main architecture of proposed cybersecurity framework which identifies the malicious edge device and makes the fog computing security system more adaptive in nature. Fog layer becomes adaptive in the sense that once a particular type of device hack path is detected by the system, it is stored in the repository so that it can be prevented in the future. In our cybersecurity framework, all edge devices receive their services through Secure Load Balancer (SLB) located in the distributed fog computing environment for balancing the load of available fog layer resources and IoT devices. When requests for services are made on the fog layer, the SLB grant services to IoT application on the LD under continuous surveillance of IDS.

Whenever IDS detects any attack, it generates the attack alarm and triggers the detection phase for the edge device. IDS send edge device identity and category of attack to the Markov1. Markov1 is the first stage Markov model which calculates the attack probability of the edge device and its category. Markov2 is the second stage Markov model which predicts whether the edge device should be shifted to VHD or not on the basis of information send by Markov1. SLB continuously monitors each device's activities and updates their respective probability transition matrix's values using Markov1. Shifted devices are continuously monitored by the VHD. If VHD detects a LD on the VHD which had been shifted by mistake, it calculates LDP of the edge device using Markov3 and sends the value to Markov4. Markov3 is the VHD's first stage Markov model and Markov4 is VHD's second stage Markov model. Markov4 predicts whether to shift edge device back to LD status or not. Concurrently, it monitors all activities of the HDs and at the end it generates the log files. All the log files are saved into the attack database repository. The attack database repository helps the SLB to prevent particular type of attack in the future. The flow of the cybersecurity framework is shown in Fig. 4. The cybersecurity framework works in three important phases which are explained in detail ahead.

4.1 Intrusion Detection Phase

In the first phase, SLB initially direct every IoT device request to the legitimate edge device. SLB monitors each edge device's activities using IDS deployed within SLB. SLB's IDS can be the combination of multiple types of IDS such as Misuse based Intrusion Detection System (MIDS), Anomaly based Intrusion Detection System (AIDS) and Network based Intrusion Detection System (NIDS) [46]. MIDS analyzes all edge devices activities and if any edge device misuses its rights then it generates an attack alarm. AIDS analyzes all edge device's activities when some anomaly behavior of edge device is detected by it, it generates an attack alarm. NIDS analyzes the traffic of the fog computing network when some edge device tries to slow down the speed of the network for example by sending data at slower rate than its capacity then it generates an attack alarm. Proposed cybersecurity framework is capable of installing any type of IDS or IPS system but for experimental evaluation Snort IDS has been used which is also a hybrid IDS system.

4.2 Activity Analysis Phase

In the second phase, a two-stage Markov model module which is embedded in SLB is used. Two-stage Markov model helps the SLB to identify malicious edge device from the request of edge device and data generated by the IDS system. When an attack is launch on an edge device, IDS generates the attack alarm and sends the detected edge device's information to the two-stage Markov model module of the SLB.

On the basis of the IDS's information, the two-stage Markov model predicts whether the edge device should be shifted to VHD or not. Fig. 5 shows the workflow diagram of detection phase. First stage Markov model (Markov1) predicts the edge device category and shifting probability of the edge device from the output generated by IDS. Second stage Markov model (Markov2) decides whether to shift the edge device on the VHD or not on the basis of the edge device category and shifting probability of the edge device.

4.3 Adaptive Phase

In the third phase, the VHD is responsible for the generation of log files of the malicious edge device activities. VHD is a cloud based virtual application device which has dummy data which looks like an original data and provides same environment as the real smart application. Unlike firewall or IDS, VHD does not give protection from attacks but detects the attack path and generates log files of attacks. The log files are logged in the Attack Database Repository (ADR) and these are used by SLB for predicting the same type of attacks in the future. VHD is also responsible for finding the LD on the VHD which had been shifted on it by mistake due to false alarm. To detect the LD on the VHD, it uses the honeypot two-stage Markov model.

4.4 Virtual Honeypot Device (VHD)

The proposed cybersecurity framework's VHD monitors the malicious edge devices as well as identify the LD which had been shifted on to the VHD by mistake. The monitor of VHD measures each activity of the malicious edge device and sends its result to the log file generator. Here, the log file generator generates the log files of the malicious edge device activities and save them into the ADR. This ADR helps the framework to prevent the unknown attacks in future and makes the framework adaptive in nature. While monitoring if any malicious edge device is detected as a LD by VHD's monitor, immediately its LDP is calculated by Markov3. After that, Markov4 takes the decision whether to shift edge device back on the real application on the basis of LDP.

5. Experimental Evaluation

In this section, complete evaluation of proposed framework is done using real-time setup using virtual machines from private and public cloud. Fig. 6 shows the overall setup deployed for the evaluation and also depicts different tools used in the process.

To test the proposed framework, an Apache based web service has been used because it is easy to create vast number of attacks to be performed on any web service. Also, using different URL and port number the size of total attacks can also be increased drastically so that overall system can be tested exhaustively. Multiple versions of pytbull 2.1 [51] attack generator have been emulated on 4 virtual machines with 4 vCPU, 40 GB of disk and 4GB of RAM on OpenStack based private cloud available in our university. Each virtual machine has been running 20 emulated versions of pytbull which is equal to 80 parallel attack sources. Pytbull is a python based attack generator which can generate more than 300 distinct attacks which are divided into multiple categories such as client side attacks, bad traffic attacks, fragmented packet attack, multiple failed login attack, shell code attack, denial of service attack etc. These requests generated by pytbull can be easily mixed with genuine requests so to test the system with distinction of attacks from the genuine requests.

All the requests generated by the pytbull is forwarded to the secure load balancer which contains markov model based proposed framework. Secure load balancer is running on DS3_V2 machine of Microsoft Azure which has 4 cores, 14GB of memory, 8 data disks, 12800 max IOPS and 28GB of SSD drive. Secure load balancer has 80 markov models related to 80 attack sources created by the emulation of pytbull. This machine is running Microsoft Server 2012 R2 with Matlab services and Java JDK 1.7. This machine is also running a live version of Snort [52] intrusion detection system for web applications. Snort performs multiple checks on the web requests and also classifies them into severity of the attack performed. Snort provides input to the proposed framework about the attack and severity of the attack so thet proposed framework can make decision for sending fog device to virtual honeypot cloud.

Glastopf [53] has been used as virtual honeypot for the apache web service and it is deployed on Microsoft Azure DS2_V2 machine with 2 cores, 7GB of memory, 4 data disks, 6400 max IOPS and 14GB of SSD drive. It gives same type of response to the web request as given by original apache web server so that device activity can be logged. Whole system is run for multiple duration of time and various metrics are recorded which are discussed in next sections.

5.3 Input Parameters

To test the proposed framework exhaustively, appropriate design of input parameter is very important. Experimental setup created 80 attack sources using pytbull which can be easily trained to use specific type of attacks. All the attacks generated by pytbull are divided into different severity levels which ranges from 0 to 5 where 1 is the least severity level and 0 is the genuine request. Table 2 provides the setup of input parameters used for test of proposed framework. All device category has same number of attack sources but with different percentage and severity level of attacks. Each attack source on an average generates 10 requests per second and total experiment was run for 60 minutes. Fig. 7 shows total number and different types of attacks generated by the pytbull emulated version running on 4 virtual machines of Open stack. All the requests are generate based on the percentage explained in Table 3 so most of the requests are generated in linear fashion. Graphs in Fig. 7 shows the accumulated number of requests from time 0 to 60 minutes.

5.3 Two-stage Markov Model Training

Mendel HMM toolbox of Matlab is used for the training of two-stage Markov model [50]. The attacks generated by the pytbull running on multiple virtual machines and their severity level is feed into the Mendel HMM toolbox [54] to generate probabilities for the state transition matrix. Based on the state transition matrix Mendel HMM toolbox predicts the future state of the fog device. Matlab code for predicting the future state of the fog device is converted into the jar file and it has been used as parameter sweep application for 80 attack sources used in the experimental setup. So, separate versions of Matlab for all the attack source is generated and updated which provides system more capabilities for personalized attack prevention based on activities of each fog device. Fig. 8 shows Matlab plot for the state transitions of 1st device for the total length of experiment which is 60 minutes. This device state changes from LD to SD and again back to LD based on the values provided by Mendel HMM toolbox.

5.4 Overall System Evaluation

As the experimental setup is complete, this section provides results based on the overall system evaluation. Our cybersecurity framework is compared with two systems which are “with_IDS” and “without IDS” (just firewall and inbuilt securities). In with_IDS system we have simple snort IDS filtering the web requests generated by attack sources running on the virtual machines. All the requests filtered by IDS as attacks are prevented to reach to the web application and are blocked at IDS virtual machine. In without_IDS we are not using snort IDS and system has only its firewall and traditional security policies. All the web requests are going directly to the apache virtual machine hosting the web application. Duration of experiment and number of input requests are already discussed in Section

5.3. Various metrics are recorded for all the three systems in evaluation and Fig. 9 shows the output generated for them.

Fig. 9(a) provides the average resource utilization of virtual machine which is hosting the web server for processing of web requests generated by the attack sources. Resource utilization of this virtual machine can be increased because of two reasons which are large processing of web requests and system is slow due to attacks. Proposed framework provided an average of 85% resource utilization which is ideal for proper working of the virtual machine. In with_IDS has low utilization due to less number of queries processed by the web server virtual machine and without_IDS has high utilization because web server was attacked by many attacks generated by pytbull. This is validated by the response time graph shown in Fig. 9(b). As depicted in Fig. 9(b) response time of with_IDS is less due to less number of queries processed by the web server and response time for without_IDS is considerably high due to attacks performed by pytbull. However, proposed framework provided better resource utilization and response time due to effective filtering done by HMM. Due to better resource utilization and response time of proposed framework, it could process more number of web requests as compared to both other systems as shown in Fig. 9(c).

Proposed HMM evaluation is also done during the overall evaluation of proposed framework and results are presented in Fig. 9(d), Fig. 9(e) and Fig. 9(f). Fig. 9(d) shows the number of web requests which are processed by honeypot virtual machine because they were identified as attacks by HMM. As the total number of attacks increased with time, shown in Fig. 7(b), total requests submitted to honeypot cloud also increased. Device categories are changed based on the input attacks conducted by them. Due to brevity and limitation of space it is not possible to show the changes in categories of all devices. However, Fig. 9(e) shows the average change in categories of all the devices due the course of experimental evaluation. As we already know the severity of the attacks generated by pytbull, we compared the HMM prediction with the dataset and Fig. 9(f) shows the resultant accuracy of HMM. Accuracy of HMM was around 65% when the experiment started and it can be increased to around 92% at the end of 60 minutes. It can also be seen that system achieve the accuracy of around 80% after the 20^{th} minute of the experiment. It can be observed from Fig. 9(e) that from time 20 to 25 there is least number of changes in the categories of the devices which is also advocated by Fig. 9(b) showing better response time at time interval of 20 to 25 minutes.

6.5 Functionality Comparison

The proposed cybersecurity framework should able to handle all issues of edge device attacks and also capable to revert back the LD from the VHD. Table 4 shows the functionality comparison of the proposed framework with other insider security frameworks. Our cybersecurity framework has contained all the functions necessary to provide a better robust defense, including IDS, reverting back legitimate device, adaptive in nature and false alarm controller.

6. Conclusion and future work

Edge device attacks will become the bottle neck in the successful implementation of the fog computing environment [55]–[57]. In this paper, our proposed cybersecurity framework has been fully illustrated to identify the malicious edge devices in the distributed fog computing environment. The proposed cybersecurity framework uses the two-stage Markov model for early prediction of the malicious edge devices as well as legitimate edge devices. Results of the experiments show the effectiveness and effectiveness of our framework with supporting test outcomes. A key point of the proposed framework is to revert back the legitimate edge device from the VHD which could happen mistakenly. Additionally, functions to support IDS, adaptive in nature and false alarm controller have been added and fully tested. Our future work will include integration with large scale ethical hacking and penetration services from Cloud Computing Adoption Framework [41] to ensure our services are robust and resilient enough against attacks and hacking and designing a framework which will effectively deal with hacked devices shifted to VHD.

Reference

[1] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K. K. R. Choo, and M. Dlodlo, “From Cloud to Fog Computing: A Review and a Conceptual Live VM Migration Framework,” *IEEE Access*, vol. 5. pp. 8284–8300, 2017.

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog Computing and Its Role in the Internet of Things,” *Proc. first Ed. MCC Work. Mob. cloud Comput.*, pp. 13–16, 2012.

[3] I. Stojmenovic and S. Wen, “The Fog Computing Paradigm: Scenarios and Security Issues,” *Proc. 2014 Fed. Conf. Comput. Sci. Inf. Syst.*, vol. 2, pp. 1–8, 2014.

[4] S. Furnell, “Enemies within: The problem of insider attacks,” *Computer Fraud and Security*, vol. 2004, no. 7. pp. 6–11, 2004.

[5] PriceWaterHouse Coopers L.L.C., “Key findings from the 2013 US state of cyber-crime survey,” *Technical Report*, 2013. [Online]. Available: https://www.pwc.com/en_US/us/increasing-iteffectiveness/publications/assets/us-state-of-cybercrime. [Accessed: 13-Feb-2017].

[6] S. Fine, Y. Singer, and N. Tishby, “Hierarchical Hidden Markov Model: Analysis and applications,” *Mach. Learn.*, vol. 32, no. 1, pp. 41–62, 1998.

[7] W. Li and Z. Guo, “Hidden Markov Model Based Real Time Network Security Quantification Method,” in *2009 International Conference on Networks Security, Wireless Communications and Trusted Computing*, 2009, vol. 2, pp. 94–100.

[8] S. Abraham and S. Nair, “Cyber security analytics: A stochastic model for security quantification using absorbing markov chains,” *J. Commun.*, vol. 9, no. 12, pp. 899–907, 2014.

[9] C. J. D’Orazio, R. Lu, K. K. R. Choo, and A. V. Vasilakos, “A Markov adversary model to detect vulnerable iOS devices and vulnerabilities in iOS apps,” *Appl. Math. Comput.*, vol. 293, pp. 523–544, 2017.

[10] V. Kharchenko, M. Kolisnyk, I. Piskachova, and N. Bardis, “Reliability & security issues for IoT-based smart business center: Architecture & Markov model,” in *Proceedings - 2016 3rd International Conference on Mathematics and Computers in Sciences and in Industry, MCSI 2016*, 2017, pp. 313–318.

[11] Honeynet Project [Honeypot], “Blogs | The Honeynet Project.” [Online]. Available: http://www.honeynet.org/. [Accessed: 13-Feb-2017].

[12] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, “Intrusion detection system: A comprehensive review,” *J. Netw. Comput. Appl.*, vol. 36, pp. 16–24, 2012.

[13] J. Peng, K.-K. R. Choo, and H. Ashman, “User profiling in intrusion detection: A review,” *J. Netw. Comput. Appl.*, vol. 72, pp. 14–27, 2016.

[14] V. Prokhorenko, K. K. R. Choo, and H. Ashman, “Context-oriented web application protection model,” *Appl. Math. Comput.*, vol. 285, pp. 59–78, 2016.

[15] V. Prokhorenko, K. K. R. Choo, and H. Ashman, “Intent-Based Extensible Real-Time PHP Supervision Framework,” *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 10, pp. 2215–2226, 2016.

[16] V. Prokhorenko, K. K. R. Choo, and H. Ashman, “Web application protection techniques: A taxonomy,” *Journal of Network and Computer Applications*, vol. 60. pp. 95–112, 2016.

[17] O. Osanaiye, H. Cai, K.-K. R. Choo, A. Dehghantanha, Z. Xu, and M. Dlodlo, “Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing,” *EURASIP J. Wirel. Commun. Netw.*, vol. 2016, no. 1, p. 130, 2016.

[18] O. Osanaiye, K. K. R. Choo, and M. Dlodlo, “Distributed denial of service (DDoS) resilience in cloud: Review and conceptual cloud DDoS mitigation framework,” *Journal of Network and Computer Applications*, vol. 67. pp. 147–165, 2016.

[19] W. Zhang, H. He, and T. hoon Kim, “Xen-based virtual honeypot system for smart device,” *Multimed. Tools Appl.*, vol. 74, no. 19, pp. 8541–8558, Oct. 2015.

[20] T. H. Luan, L. Gao, Z. Li, Y. Xiang, and L. Sun, “Fog Computing: Focusing on Mobile Users at the Edge,” *eprint arXiv:1502.01815*, pp. 1–11, 2015.

[21] P. Ning and K. Sun, “How to misuse AODV: A case study of insider attacks against mobile ad-hoc routing protocols,” in *IEEE Systems, Man and Cybernetics Society Information Assurance Workshop*, 2003, pp. 60–67.

[22] R. Chinchani, a. Iyer, H. Q. Ngo, and S. Upadhyaya, “Towards a theory of insider threat assessment,” *2005 Int. Conf. Dependable Syst. Networks*, 2005.

[23] G. B. Magklaras and S. M. Furnell, “A preliminary model of end user sophistication for insider threat prediction in IT systems,” *Comput. Secur.*, vol. 24, no. 5, pp. 371–380, 2005.

[24] M. M. B. Salem, S. Hershkop, and S. J. S. Stolfo, “A Survey of Insider Attack Detection Research,” in *Advances in Information Security*, vol. 39, Boston, MA: Springer US, 2008, pp. 69–90.

[25] J. Predd, S. L. Pfleeger, J. Hunker, and C. Bulford, “Insiders behaving badly,” *IEEE Secur. Priv.*, vol. 6, no. 4, pp. 66–70, 2008.

[26] M. Warkentin and R. Willison, “Behavioral and policy issues in information systems security: the insider threat,” *Eur. J. Inf. Syst.*, vol. 18, no. 2, pp. 101–105, Apr. 2009.

[27] I. J. Martinez-Moyano, E. Rich, S. Conrad, D. F. Andersen, and T. R. Stewart, “A behavioral

theory of insider-threat risks," *ACM Trans. Model. Comput. Simul.*, vol. 18, no. 2, pp. 1–27, 2008.

[28] C. Colwill, "Human factors in information security: The insider threat - Who can you trust these days?," *Inf. Secur. Tech. Rep.*, vol. 14, no. 4, pp. 186–196, 2009.

[29] D. A. Mundie, S. J. Perl, and C. L. Huth, "Insider Threat Defined : Discovering the Prototypical Case," *J. Wirel. Mob. Networks, Ubiquitous Comput. Dependable Appl.*, vol. 5, no. 2, pp. 7–23, 2014.

[30] G. B. Magklaras and S. M. Furnell, "Insider threat prediction tool: Evaluating the probability of IT misuse," *Comput. Secur.*, vol. 21, no. 1, pp. 62–73, 1998.

[31] E. E. Schultz, "A framework for understanding and predicting insider attacks," *Comput. Secur.*, vol. 21, no. 6, pp. 526–531, 2002.

[32] M. Theoharidou, S. Kokolakis, M. Karyda, and E. Kiountouzis, "The insider threat to information systems and the effectiveness of ISO17799," *Comput. Secur.*, vol. 24, no. 6, pp. 472–484, 2005.

[33] B. M. Bowen, M. Ben Salem, S. Hershkop, A. D. Keromytis, and S. J. Stolfo, "Designing host and network sensors to mitigate the insider threat," *IEEE Secur. Priv.*, vol. 7, no. 6, pp. 22–29, Nov. 2009.

[34] S. L. Pfleeger, J. B. Predd, J. Hunker, and C. Bulford, "Insiders behaving badly: Addressing bad actors and their actions," *IEEE Trans. Inf. Forensics Secur.*, vol. 5, no. 1, pp. 169–179, 2010.

[35] D. L. Costa, M. L. Collins, S. J. Perl, M. J. Albrethsen, G. J. Silowash, and D. L. Spooner, "An ontology for insider threat indicators development and applications," in *CEUR Workshop Proceedings*, 2014, vol. 1304, pp. 48–53.

[36] R. Zhang, X. Chen, J. Shi, F. Xu, and Y. Pu, "Detecting insider threat based on document access behavior analysis," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8710 LNCS, Springer International Publishing, 2014, pp. 376–387.

[37] M. B. Bondada and S. M. S. Bhanu, "Analyzing user behavior using keystroke dynamics to protect cloud from malicious insiders," in *2014 IEEE International Conference on Cloud Computing in Emerging Markets, CCEM 2014*, 2015, pp. 1–8.

[38] I. Ray and N. Poolsapassit, "Using attack trees to identify malicious attacks from authorized insiders," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2005, vol. 3679 LNCS, pp. 231–246.

[39] A. W. Dent, "Hybrid Signcryption Schemes with Insider Security," in *Information Security and Privacy, LNCS*, 2005, vol. 3574, pp. 253–266.

[40] F. Liu, X. Cheng, and D. Chen, "Insider Attacker Detection in Wireless Sensor Networks," in *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, 2007, pp. 1937–1945.

[41] D. Liu, X. Wang, and J. Camp, "Game-theoretic modeling and analysis of insider threats," *Int. J. Crit. Infrastruct. Prot.*, vol. 1, pp. 75–80, 2008.

[42] F. L. Greitzer, A. P. Moore, D. M. Cappelli, D. H. Andrews, L. A. Carroll, and T. D. Hull, "Combating the insider cyber threat," *IEEE Secur. Priv.*, vol. 6, no. 1, pp. 61–64, 2008.

[43] C. W. Probst, J. Hunker, D. Gollmann, and M. Bishop, *Insider threats in cyber security*, vol. 49. 2010.

[44] M. Sriram, V. Patel, D. Harishma, and N. Lakshmanan, "A hybrid protocol to secure the cloud from insider threats," in *2014 IEEE International Conference on Cloud Computing in Emerging Markets, CCEM 2014*, 2015, pp. 1–5.

[45] H. Artail, H. Safa, M. Sraj, I. Kuwatly, and Z. Al-Masri, "A hybrid honeypot framework for improving intrusion detection systems in protecting organizational networks," *Comput. Secur.*, vol. 25, no. 4, pp. 274–288, 2006.

[46] A. Sardana and R. Joshi, "An auto-responsive honeypot architecture for dynamic resource allocation and QoS adaptation in DDoS attacked networks," *Comput. Commun.*, vol. 32, no. 12, pp. 1384–1399, 2009.

[47] A. Chonka, Y. Xiang, W. Zhou, and A. Bonti, "Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks," *J. Netw. Comput. Appl.*, vol. 34, no. 4, pp. 1097–1107, 2011.

[48] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in Cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1. pp. 42–57, 2013.

[49] A. Patel, M. Taghavi, K. Bakhtiyari, and J. Celestino Júnior, "An intrusion detection and prevention system in cloud computing: A systematic review," *Journal of Network and*

Computer Applications, vol. 36, no. 1. pp. 25–41, 2013.

[50] V. De Fonzo, F. Aluffi-Pentini, and V. Parisi, "Hidden Markov Models in Bioinformatics," *Curr. Bioinform.*, vol. 2, no. 1, pp. 49–61, 2007.

[51] "Pytbull," 2017. [Online]. Available: http://pytbull.sourceforge.net. [Accessed: 20-Aug-2017].

[52] "Snort Network Intrusion Detection and Prevention System," 2017. [Online]. Available: https://www.snort.org. [Accessed: 20-Aug-2017].

[53] "Glastopf- A Web Application Honeypot," 2017. [Online]. Available: https://www.edgis-security.org/honeypot/glastopf. [Accessed: 19-Aug-2017].

[54] S. Thorvaldsen, "A tutorial on Markov models based on Mendel's classical experiments.," *J. Bioinform. Comput. Biol.*, vol. 3, no. 6, pp. 1441–1460, 2005.

[55] C. J. D'Orazio and K. K. R. Choo, "Circumventing iOS security mechanisms for APT forensic investigations: A security taxonomy for cloud apps," *Future Generation Computer Systems*, 2016.

[56] C. J. D'Orazio and K. K. R. Choo, "A technique to circumvent SSL/TLS validations on iOS devices," *Futur. Gener. Comput. Syst.*, vol. 74, pp. 366–374, 2017.

[57] C. J. D'Orazio, K. K. R. Choo, and L. T. Yang, "Data Exfiltration from Internet of Things Devices: IOS Devices as Case Studies," *IEEE Internet Things J.*, vol. 4, no. 2, pp. 524–535, 2017.

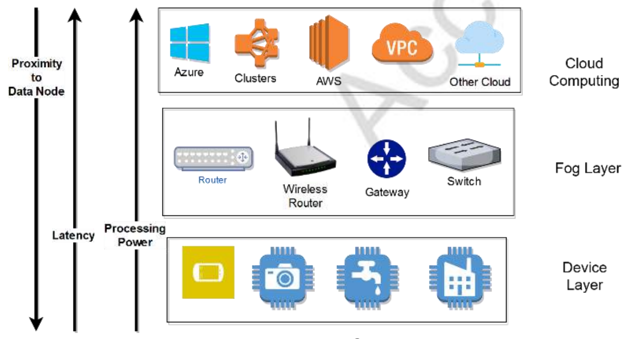
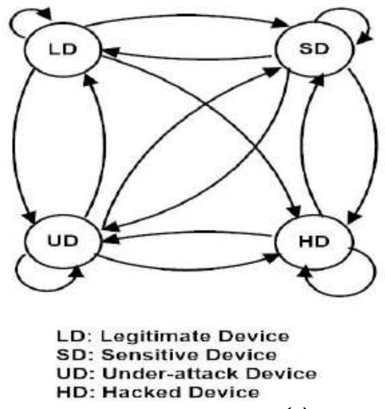
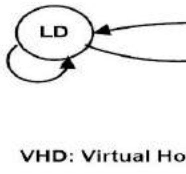


Fig. 1. Layers of Fog Computing



(a)



(b)

Fig.2. General Transition Graph used by (a) First Stage Markov Model (b) Second Stage Markov Model

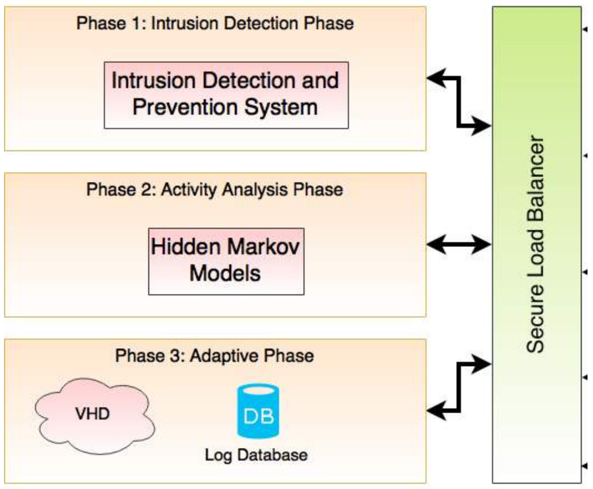

Phase 1: Intrusion Detection Phase
Intrusion Detection and Prevention System
Phase 2: Activity Analysis Phase
Hidden Markov Models
Phase 3: Adaptive Phase
VHD
DB
Log Database
Secure Load Balancer

Fig. 3. Three Phase of Proposed cybersecurity Framework

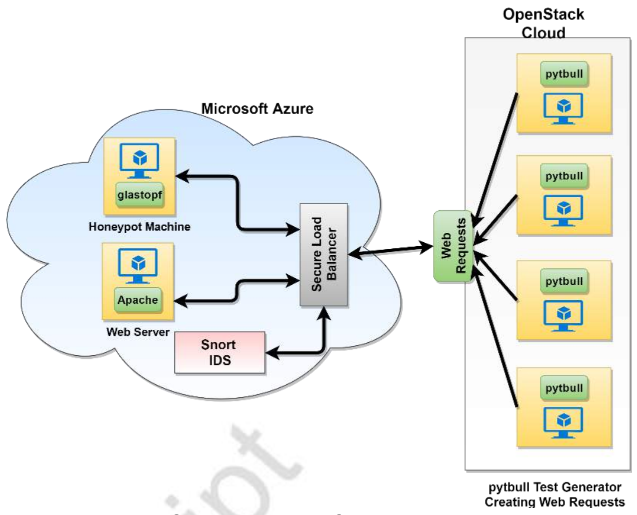


Fig. 6. Test Bed for Simulation of our Cybersecurity Framework

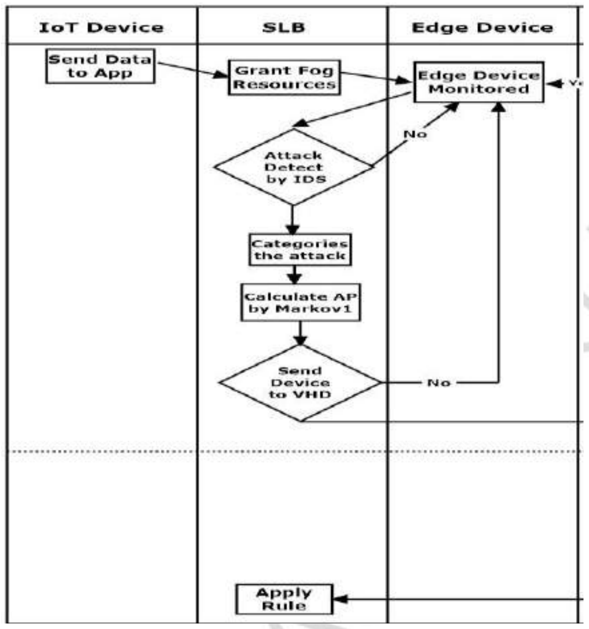
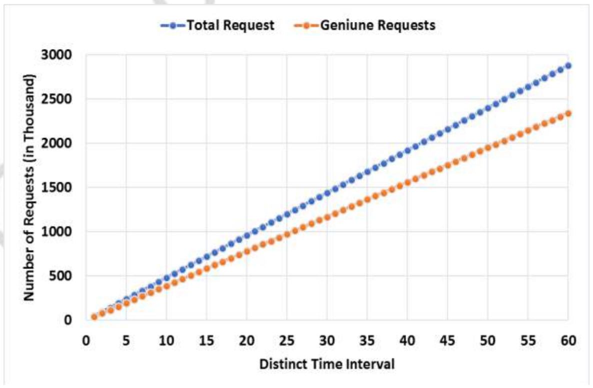
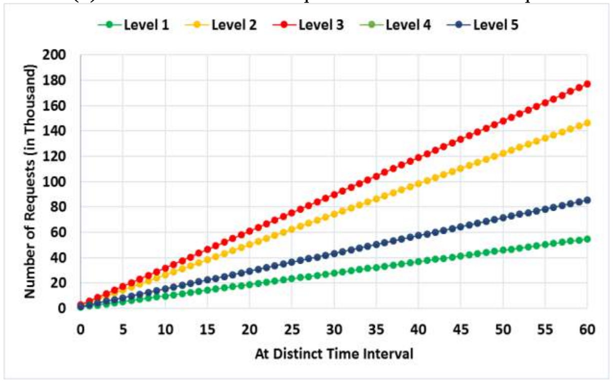


Fig. 4. System Flow



(a) Total Number of Requests and Genuine Requests



(b) Different Level of Attack Requests

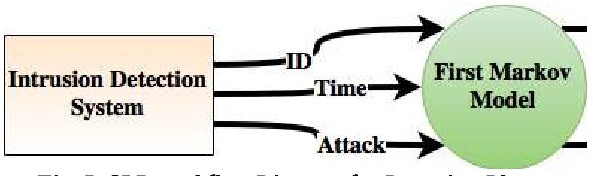


Fig. 5. SLB workflow Diagram for Detection Phase

Fig. 7. Total Requests Generated by the pytbull Emulated on 4 Virtual Machines

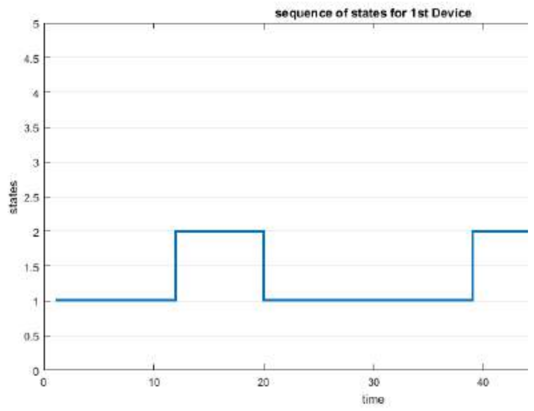
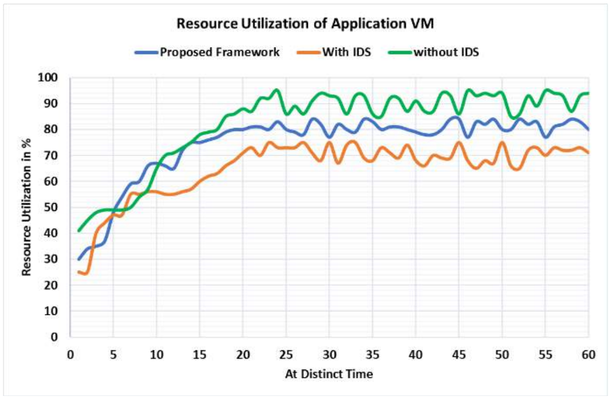
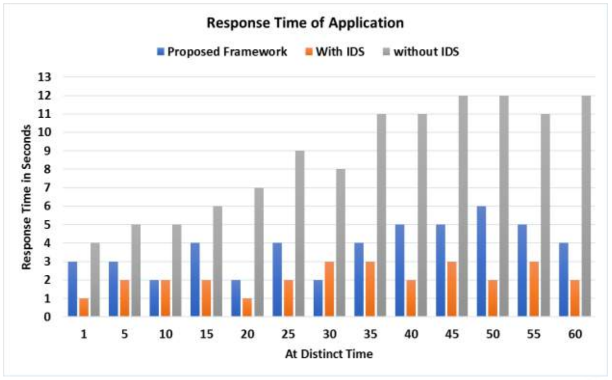


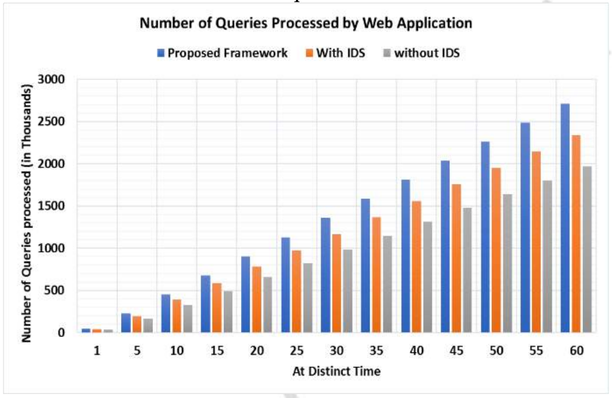
Fig. 8. Graph for State Transition of 1st Device based on type and number of attacks



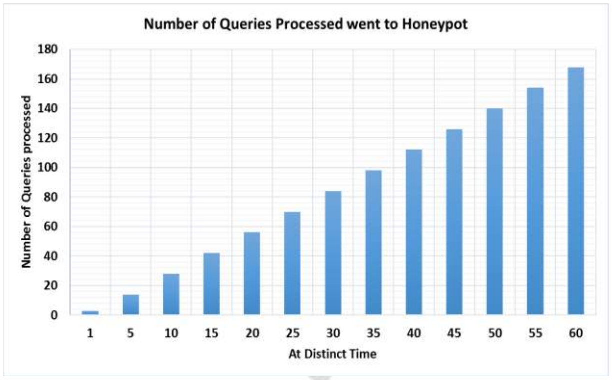
(a) Resource Utilization of Web Service Virtual Machine



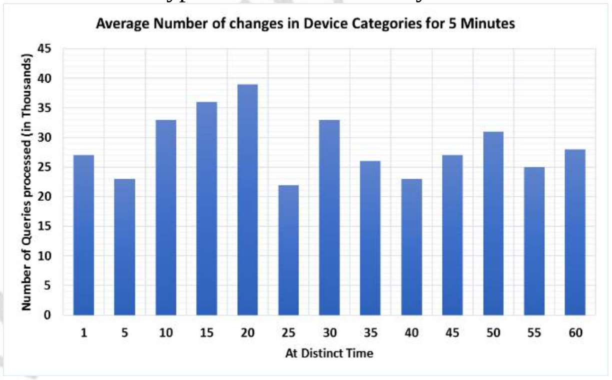
(b) Average Response Time of Web Server for all Web Requests



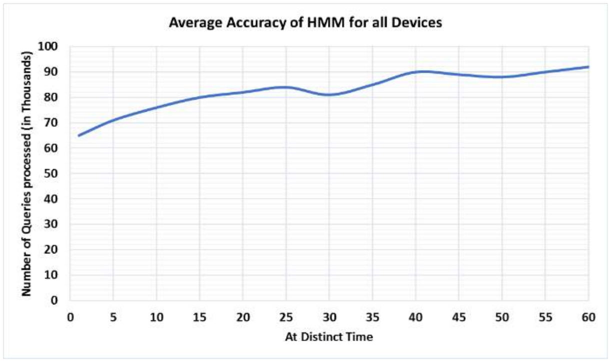
(c) Number of Web Requests Perfectly Processed by Web Server



(d) Number of Web Requests which are Directed to Honeypot Virtual Machine by HMM



(e) Average Change in Category of All Devices using HMM



(f) Accuracy of HMM for all Attack Sources

Fig. 9. Metrics Recorded from Overall System Simulation

Table 1. Comparison of Different IDS Techniques Studied in Literature

S. No.	Technique	IDS	Honeypot	Limitation
1.	[4]	Misuse Based	No	No model is proposed and evaluated.
2.	[45]	Yes	Yes	High level concept is proposed but no system is evaluated in the paper using real attacks.
3.	[46]	Yes	Auto-responsive	Only worked for DDOS attacks and honeypot is static.
4.	[47]	Yes	No	Only work with DOS attack and honeypot concept is not proposed for making system more adaptive.
5.	[19]	NO	Virtual Honeypot	Only virtual honeypot concept is explained for mitigating attackers on it. No discussion of IDS and initial attack preventer is presented.
6.	[48]	Yes	No	IDS and IPS are discussed in this study no discussion of honeypot is presented.
7.	[49]	Fuzzy logic based	No	Provides high level concept of using adaptive IDS to prevent attackers but no implementation and model is proposed.

Table 2. Terminologies of Markov models

Name	Stage	Description
Markov1	First stage Markov model	Used to compute the shifting probability and predict the future state of any edge device.
Markov2	Second stage Markov model	Used to take decision whether to shift edge device to VHD.
Markov3	Honeypot's first stage Markov model	Used to compute the legitimate edge device probability and predict the future state of the device.
Markov4	Honeypot's second stage Markov model	Used to take decision whether an identified malicious edge device should be shifted back on legitimate cloud.

Table 3: Different input parameters of attack sources

Category	Percentage of Attack requests	Severity Level of Attack Request	Number of Source
LD	0	0	20
SD	10%	1,2	10
	20%	1,2	10
UD	20%	2,3	10
	30%	2,3	10
HD	30%	3,4,5	10
	40%	3,4,5	10

Table 4. Functionality Comparison

Frameworks in comparison	IDS			Revert Back Legitimate Device	Adaptive in Nature	False Alarm Controller
	Network	Anomaly	Misuse			
Furnel (2004) [4]	No	Yes	Yes	No	No	No
Artail et al. (2007) [45]	Yes	Yes	Yes	No	Yes	No
Sardana and Joshi (2009) [46]	No	No	No	No	Yes	No
Modi et al. (2013) [48]	Yes	Yes	Yes	No	Yes	No
Patel et al. (2013) [49]	Yes	No	No	No	Yes	Yes
Our Cybersecurity Framework	Yes	Yes	Yes	Yes	Yes	Yes