Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2023.028029
Article

Tech Science Press

A Graph Theory Based Self-Learning Honeypot to Detect Persistent Threats

R. T. Pavendan[1,*], K. Sankar[1] and K. A. Varun Kumar[2]

[1]Department of Mathematics, College of Engineering, Anna University, Chennai, 600 025, India
[2]Department of Networking and Communications, School of Computing, SRM Institute of Science and Technology, Kattankulathur, 602 203, India
*Corresponding Author: R. T. Pavendan. Email: paavendanraja@gmail.com
Received: 31 January 2022; Accepted: 15 March 2022

Abstract: Attacks on the cyber space is getting exponential in recent times. Illegal penetrations and breaches are real threats to the individuals and organizations. Conventional security systems are good enough to detect the known threats but when it comes to Advanced Persistent Threats (APTs) they fails. These APTs are targeted, more sophisticated and very persistent and incorporates lot of evasive techniques to bypass the existing defenses. Hence, there is a need for an effective defense system that can achieve a complete reliance of security. To address the above-mentioned issues, this paper proposes a novel honeypot system that tracks the anonymous behavior of the APT threats. The key idea of honeypot leverages the concepts of graph theory to detect such targeted attacks. The proposed honeypot is self-realizing, strategic assisted which withholds the APTs actionable techniques and observes the behavior for analysis and modelling. The proposed graph theory based self learning honeypot using the results $\gamma(C(n,1))$,γ_c $(C(n,1))$, γ_{sc} $(C(n,1))$ outperforms traditional techniques by detecting APTs behavioral with detection rate of 96%.

Keywords: Graph theory; Domination; Connected Domination; Secure Connected Domination; honeypot; self learning; ransomware

1 Introduction

Let G=<V, E> be a simple graph. A subset S of V is a dominating set of G if every vertex in V\S is adjacent to at least a vertex in S. A dominating set S of a graph G is called a secure dominating set if for each $v \in V\S$, there exists $u \in S$ such that v is adjacent to u and $S_1=(S\{u\}) \cup \{v\}$ is a dominating set. Further, if the induced subgraph of such dominating set (secure dominating set) S of G is connected, then S is said to be a connected dominating set (secure connected dominating set). The minimum cardinality of such dominating set (connected dominating set, secure connected dominating set) of G is called the domination number (connected domination number, secure connected domination number) of G and is denoted by $\gamma(G)$(γ_c (G),γ_{sc} (G)). Let C(n, 1) be a Cycle graph with one chord. By a graph G=<V, E> we mean a finite, undirected graph without loops (or) multiple edges. For graph terminology [1,2]. The domination in graph has been introduced by O. Ore [3] in 1962, the protection of graph was introduced by E. J. Cockayne et al. [4], the connected domination in graph was introduced by Sampathkumar et al. [5] in 1979 and the secure connected domination in graph

was introduced by Amerkhan et al. [6] in 2014. Dominating sets in a graph provides applications both in the position (or location) and the protection strategies.

Let C_n:{v_1, v_2,…, v_n} be any cycle of order n and v_1 be apex, by adding a new edge (chord) v_1 $v_i \notin$ E (C_n) for $3 \leq i \leq n-1$ is called a cycle with a chord and is denoted by C(n, 1). Observe that, there are n–3 such possibilities of chords. Note that, C(n, 1) with a chord v_1 v_i is isomorphic to C(n, 1) with a chord v_1 $v_{(n-1)}$. Hence, we have $\lceil (n-3)/2 \rceil$ non-isomorphic graph possibilities in C(n, 1). Therefore, finding the above domination parameters for a cycle with one chord is dynamic in nature [7].

In this research, a graph theory based approach is used to centre the security preserving context of events generated by the honeypots [8]. These events are considered to be the breaking down components of total observables in the centralised utilities like Security Information and Event monitoring. This helps in accounting the whole security events from the past and present data. Further, it also helps to preserve and visualize the relationship across all of the data elements. Finally a classification model is applied to surface the informative relationships [9].

Ransomware is a kind of malware which is designed to compromise the victim's system and blackmailing the victim for ransom. The most common known types of ransomwares are screen lockers and encryptors. Screen lockers, as the name implies block access to the system data with lock screen. Encryptors are type of ransomware which can encrypt the data and replicating themselves. Encryptors are used to threaten victim's data. The content cannot view without decryption key. A ransomware is noxious program, which encodes the casualty report and limit casualties from getting to their own framework with the exception of if a measure of ransomware is paid. Amounts of ransomwares which are available working at a benefit publicize are not exactly equivalent to one another. Regardless, these varieties utilize an external relationship for getting the best approach to begin encryption measure. Thus recognizing and separating the ransomware correspondence can prevent the presentation of encryption measure [10]. To resolve this issue, this work proposes an original traffic assessment model. This proposed scheme uses an intelligent parcel examination plot using gathering of classifiers for ransomware traffic course of action. The introduction of the proposed system is penniless down using a safe testbed. In this investigation, we show how ransomware multiplies and defiles the devices. Through this, live traffic request of ransomware has been critically inspected. Further, the clever methodology has acquainted with bunch the ransomware traffic by using significant learning procedures. Considering request, acknowledgment of ransomware is moved nearer with the traits of the framework traffic and its exchanges. A honeypot makes distinguishing pernicious traffic dead basic. That is on the grounds that any traffic to a honeypot, after some underlying speedy blocking to preclude bogus up-sides, is dubious. A honeypot is a phony PC resource that exists possibly to alarm its proprietor in case it is contacted. No one ought to be contacting it or endeavoring to sign on. Since all movement is ill-conceived, no investigation is expected to tell great traffic from awful. Honeypots are a kind of trickiness innovation that permits you to comprehend assailant personal conduct standards. Security groups can utilize honeypots to explore network safety breaks to gather intel on how cybercriminals work. They likewise diminish the danger of bogus up-sides, when contrasted with conventional network safety measures, since they are probably not going to draw in real action.

An interruption avoidance framework (IPS), involving a firewall additionally an IDS, can assess the traffic and square pernicious information. It goes about as a safeguard against assaults, however it can't recognize if an application-layer demand is ordinary. This drawback might actually bring about assaults pervading the safeguard without being distinguished, e.g., a social designing aggressor may acquire touchy data by utilizing a compromised real username and secret phrase. In the event that an IPS coordinates with a honeypot, the entire framework would then be able to catch all assaulting exercises regardless of whether they are performed by inside or outside adversaries. Additionally, the information caught by honeypots can be utilized to make countermeasures, e.g., the mechanized interruption reaction

frameworks frequently utilizes honeypots as the information catch framework. Honeypots are regularly used to research as of now obscure assaults. The Black hat people group is adequately canny to make new-obscure dangers. A decent method to examine new dangers is to catch the malevolent action bit by bit as it compromises a framework. Honeypots in this way can increase the value of research by giving a conciliatory framework to be assaulted.

2 Literature Survey

Zaki et al. 2016. proposed the model to diagnosis the head and neck cancer by cloud service image processing. Authors use the Graphics processing Unit (GPU) acceleration technic with the combination of computed tomography with cone-beam (CBCT) to extracting the significant information from the image and registered that information in cloud environment. From that information doctors diagnosis patients by radiation therapy. This model reduces the complexity of the early diagnosis. Baliga et al. 2011 proposed the model to improve the performance of the cloud computing power consumption. The author describes the energy consumption of the network based cloud infrastructure, how the conventional cloud services to overcome the existing cloud drawbacks [11]. This model balances the energy consumption especially for personal computer cloud services. Taleb et al. 2017 [12] analyze the implementation of the 5G technology in the cloud computing environment. The authors examine the multi-access computing with edge 5G network infrastructure to provide the reliable service to mobile user to store the resource in the cloud and explain the various challenges in the implementation.

Mershad et al. 2017 analyse the performance of cloud data-center services. The authors describe the previous system model and examine cloud structure based on the usage of memory, network interface and CPU performance [13]. Yan et al. 2016 proposed the cost efficient method to archive the scalable cloud environment. Authors create a model to ensure the utilization of the cloud deployment and improve the migration of the cloud resource to another vendor at low cost [14]. They use aliyun's cloud computing service for testing and implementation purpose. Manikandan et al. 2021 [15] proposed the model to improve the cloud gaming service performance. The authors create a model with Ad-hoc network to provide the quality of services to the gamers. They use cloud environment for large storage and high performance gaming environments. A. T. Lo'ai et al. [16] proposed the model for improve the healthcare application. The authors use the mobile computing technology with big data analysis model improves the performance of the healthcare field. They use cloud based environment to store the resource and access them is efficient time to diagnosis the patients and improve immunity of the patients.

Chiang et al. 2015 [17] proposed the algorithm to improve the cost efficiency in cloud services. The authors use the green cloud algorithm to demonstrate proposed model and improve the N-policy of the cloud environment. The results ensure the performance and power saving polices in the cloud environment. Neelakandan proposed the model to optimise the cloud environment by customer requirements. Authors use the multi-server environment cloud environment for testing and implementation of the model [18]. This model algorithm uses the customer feedbacks to improve the cloud infrastructure and also improves the energy consumption, service agreement and maximizes the profit. Subramani et al. proposed the scheduling approach for hybrid clouds. The authors use the model for bag of tasks on clouds to process the resources in the scheduling manner [19]. This model improves the performance, integrity of the cloud and also reduces the complexity in the algorithm implementation.

B. Li et al. [20] proposed the model to deceptive attacks in the honeypot based IoT attacks. Authors use the Bayesian algorithm to analysis the threshold accuracy in deceptive action in the system. It reduce the attacker's success rate. Jain et al. presented survey on the anatomic view of investigate honeypot system. They analysis both novel decoy and captor to validate the taxonomy and predict the honeypot trends [21]. Neelakandan et al. proposed the ransomware detection model using the machine learning

algorithms. The authors describe the various ransomware detection technics and study the machine learning algorithm to reduce the probability of network intrusion detection [22]. They use bitflow method to reduce the data packet size and transfer ratio. Moore proposed the ransomware detection model using the honeypot techniques. They authors analysis the honeypot folder if its monitors the changes, it helps the accuracy of detection changes in the network [23]. Almashhadani et al. proposed the ransomware model to detect the network based crypto ransomware. The authors describe the network based activities in the crypto ransomware and build a test bed model to detect the ransom malware in the network [24]. The proposed model analysis the network packet flow rates and evaluate the system detection rate.

Petri et al. proposed the classification method for data mining technics. The authors use the data mining clustering algorithms to classify the remotely sensed imaged data. This model clustering the sensed image into 6 part [25]. Kamalraj et al. presented the Artificial Neural Networks (ANN) model to predict the student performance from the student profile. They trained the algorithm to cross validate the classified data set of the student profile [26]. Bahrami et al. presented the medical data stream distribution pattern for the rule mining algorithm. The authors use the traditional data mining methods to classify the medical data sets in the association rule manner. They also use density estimation method to predict the data set accuracy [27].

Bansal et al. proposed the model to predict the decision of the student enrolment in subject wise. They use the advance data mining method to predict the student growth in academic success radio. This model also uses the decision tree and support vector system to reduce the dropout rate of the data [28]. Ravichandran. proposed the knowledge system to mining the student data in educational institution. They use the knowledge graph education method to automate the knowledge training for data [29]. These models improve the precision average of the data set.

Zhang et al. Proposed the model to improve the cloud environment cost efficient and more reliable to computing the resource. They handle the service agreement issues in the cloud and improve the cost based parameters in terms of reliability and security [30]. The solution to manage the large amount of resource in the cloud services. They introduce the cloud based system called credit unit cloud model (cucloud) to avoid resource center mechanism and improve the store approach effectively. This model configuration supports for both organization/group and individual purpose. It consists of various components handle the various operations in the cloud environment. Bahrami et al. analysis the reliability and utilization of the cloud based system. The authors describe the cloud processing and issues faced to setup cloud environment. They handle basic issues in the cloud setup and give solution to avoid or overcome those issues in the cloud computing [31–33].

3 Mathematical Proof

$$\text{Theorem 1}: \gamma[C(n,\ 1)] = \begin{cases} \left\lceil \frac{n}{3} \right\rceil, & i \equiv 1 (\text{mod}\ \ 3) \\ \left\lceil \frac{n-1}{3} \right\rceil, & \text{otherwise} \end{cases} \tag{1}$$

Proof: C(n, 1) is a cycle graph with one chord. Let $V[C(n, 1)] = \{v_1, v_2, \ldots, v_n\}$ with v_1 as apex and v_1v_i be the chord $3 \leq i \leq n-1$. We observe that, there are vertex disjoint subgraphs such as a Cycle C_i and a Path P_{n-i} in C(n, 1) as Illustrated in the graph below:

Let $V(C_i) = \{v_1, v_2, \ldots, v_i\}$ and $V(P_{n-i}) = \{v_{i+1}, v_{i+2}, \ldots, v_n\}$. It is clear that, $V(C(n, 1)) = V(C_i) \cup V(P_{n-i})$. We know that, in C_i, a vertex v_j:$1 \leq j \leq i$ is in any one of the minimum dominating sets of C_i. Without loss of generality, let $v_1 \in S$, where S is one of the minimum dominating sets of C_i. We know that, $\gamma[C_n] = \left\lceil \frac{n}{3} \right\rceil$, $n \geq 3$ and $\gamma[P_n] = \left\lceil \frac{n}{3} \right\rceil$, $n \geq 2$ from [4].

Case (i) : $i \equiv 0,\ 2(\text{mod}\ \ 3)$.

Clearly, $\gamma[C_i] = \left\lceil \frac{i}{3} \right\rceil$. Without loss of generality, $v_1 \in S,$ where S is the minimum dominating set. Here, v_1 dominates v_n. Hence, minimum dominating set of C(n, 1) is the union of the dominating set of C_i and the dominating set of P_{n-i-1}.

when $i \equiv 0(\text{mod}\ \ 3),$ let $i = 3t, t \in Z$, then

$$\begin{aligned}
\gamma[C(n,\ 1)] &= \gamma[C_i] + \gamma[P_{n-i-1}] \\
&= \left\lceil \frac{i}{3} \right\rceil + \left\lceil \frac{n-i-1}{3} \right\rceil \\
&= \left\lceil \frac{3t}{3} \right\rceil + \left\lceil \frac{n-1}{3} - \frac{3t}{3} \right\rceil \\
&= t + \left\lceil \frac{n-1}{3} \right\rceil - t \\
&= \left\lceil \frac{n-1}{3} \right\rceil
\end{aligned} \tag{2}$$

when $i \equiv 2(\text{mod}\ \ 3),$ let $i = 3t + 2, t \in Z$, then

$$\begin{aligned}
\gamma[C(n,\ 1)] &= \gamma[C_i] + \gamma[P_{n-i-1}] \\
&= \left\lceil \frac{i}{3} \right\rceil + \left\lceil \frac{n-i-3}{3} \right\rceil \\
&= \left\lceil \frac{3t+2}{3} \right\rceil + \left\lceil \frac{n-3t-2-1}{3} \right\rceil \\
&= \left\lceil \frac{3t}{3} + \frac{2}{3} \right\rceil + \left\lceil \frac{n-1}{3} - \frac{(3t+2)}{3} \right\rceil \\
&= \left\lceil t + \frac{2}{3} \right\rceil + \left\lceil \frac{n-1}{3} - \frac{3t}{3} - \frac{2}{3} \right\rceil \\
&= t + 1 + \left\lceil \frac{n-1}{3} \right\rceil - t - 1 \\
&= \left\lceil \frac{n-1}{3} \right\rceil
\end{aligned} \tag{3}$$

Case (ii): $i \equiv 1(\text{mod}\ \ 3)$.

Since $i \equiv 1(\text{mod}\ \ 3)$, let $\{v_1, v_i\} \in S$. Here, $v_1v_n \in E(C(n, 1))$, Also, v_1 dominates v_n and $v_iv_{i+1} \in E(C(n, 1))$, v_i dominates v_{i+1}. Hence, minimum dominating set of C(n, 1) is the union of the dominating set of C_i and the dominating set of P_{n-i-2}.

When $i \equiv 1(\text{mod } 3)$, let $i = 3t + 1$, $t \in Z$, then

$$\begin{aligned}\gamma[C(n,\ 1)] &= \gamma[C_i] + \gamma[P_{n-i-2}] \\ &= \left\lceil \frac{i}{3} \right\rceil + \left\lceil \frac{n-i-2}{3} \right\rceil \\ &= \left\lceil \frac{3t+1}{3} \right\rceil + \left\lceil \frac{n-(3t+1)-2}{3} \right\rceil \\ &= \left\lceil \frac{3t}{3} + \frac{1}{3} \right\rceil + \left\lceil \frac{n}{3} - \frac{3t}{3} - \frac{3}{3} \right\rceil \\ &= \left\lceil t + \frac{1}{3} \right\rceil + \left\lceil \frac{n}{3} - t - 1 \right\rceil \\ &= t + 1 + \left\lceil \frac{n}{3} \right\rceil - t - 1 \\ &= \left\lceil \frac{n}{3} \right\rceil.\end{aligned} \tag{4}$$

$$\text{Therefore, } \gamma[C(n,\ 1)] = \begin{cases} \left\lceil \frac{n}{3} \right\rceil, & i \equiv 1(\text{mod } 3) \\ \left\lceil \frac{n-1}{3} \right\rceil, & \text{otherwise} \end{cases} \tag{5}$$

$$\text{Theorem 2: } \gamma_c[C(n,1)] = \begin{cases} n-3, & \text{if } C_3 \text{ is a subgraph of } C(n,1) \\ n-4, & \text{otherwise} \end{cases} \tag{6}$$

Proof: Let C(n, 1) be any Cycle with a chord where v_1 is the apex and v_1v_i is the chord such that $3 \le i \le n-1$. From [5], we have $\gamma_c(P_n) = n-2$, for all $n \ge 3$ and $\gamma_c(P_n) = n-2$, for all $n \ge 3$. We observe that, a Cycle with a chord will yield a non-empty vertex disjoint of a Cycle (C_i) and a Path (P_{n-i}) as in Fig. 1.

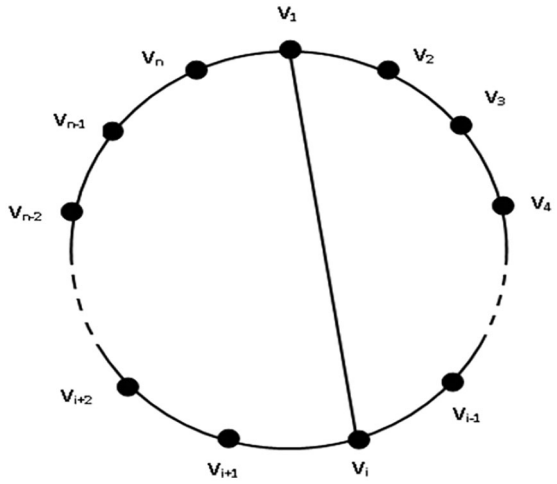

Figure 1: Cycle with one chord

The Cycle $C_i : [v_1, v_2, \ldots, v_i]$ and P_{n-i}:$[v_{i+1}, v_{i+2}, \ldots, v_n]$ are vertex disjoint graphs such that $V(C_i) \cup V(P_{n-i}) = V[C(n,\ 1)]$. Let S be the required connected dominating set for the graph C(n, 1).

Case (i) : C_3 is a subgraph of C(n, 1).

Without loss of generality, let $v_1 \in S$. Hence, v_1 dominates v_1, v_3 and v_n. By $\gamma_c[P_{n-4}] = n - 6$, with $\{v_5, v_6, \ldots, v_{n-2}\}$ as a minimum connected dominating set of P_{n-4}. However, the set $\{v_1\} \cup \{v_5, v_6, \ldots, v_{n-2}\}$ is not a connected dominating set of C(n, 1). Hence, we get $S = \{v_1\} \cup \{v_5, v_6, \ldots, v_{n-2}\} \cup \{v_{n-1}, v_n\}$ (or) $S = \{v_1\} \cup \{v_3, v_4\} \cup \{v_5, \ldots, v_{n-2}\}$ is a minimum connected dominating set for C(n, 1).

Case (ii) : C_3 is not a subgraph of C(n, 1).

Here, v_1v_i is the chord of C(n, 1) where $i \in \{4, \ldots, n-2\}$. By $\gamma_c(C_i) = i - 2$, without loss of generality, let $\{v_1\} \cup \{v_4, v_5, \ldots, v_i\}$ be the minimum connected dominating set for C_i. Here, v_1 dominates v_2, v_n and v_i dominates v_{i-1}, v_{i+1}. Hence, by $\gamma_c(P_{n-i-2}) = n - i - 4$, without loss of generality, $\{v_{i+3}, v_{i+4}, \ldots, v_{n-2}\}$ be the minimum connected dominating set for P_{n-i-2}. However, the set $\{v_1\} \cup \{v_4, v_5, \ldots, v_i\} \cup \{v_{i+3}, v_{i+4}, \ldots, v_{n-2}\}$ is not a minimum connected dominating set of C(n, 1). Hence, the set $S = \{v_1\} \cup \{v_4, v_5, \ldots, v_i\} \cup \{v_{i+1}, v_{i+2}\} \cup \{v_{i+3}, \ldots, v_{n-2}\}$ (or) $S = \{v_1\} \cup \{v_4, v_5, \ldots, v_i\} \cup \{v_{i+3}, \ldots, v_{n-2}\} \cup \{v_{n-1}, v_n\}$ is a minimum connected dominating set for C(n, 1). Hence, $\gamma_c[C(n, 1)] = n - 4$, whenever C_3 is not a subgraph of C(n, 1).

$$\text{Therefore, } \gamma_c[C(n,1)] = \begin{cases} n-3, & \text{if } C_3 \text{ is a subgraph of } C(n,1) \\ n-4, & \text{otherwise} \end{cases} \tag{7}$$

$$\text{Theorem 3 : } \gamma_{sc}[C(n,1)] = \begin{cases} n-3, & \text{if } C_3 \text{ is a subgraph of } C(n,1) \\ n-4, & \text{otherwise} \end{cases} \tag{8}$$

Proof: Let C(n, 1) be any Cycle with a chord where v_1 is the apex and v_1v_i is the chord such that $3 \leq i \leq n - 1$. From [6], we have, $\gamma_{sc}(P_n) = n$, for all $n \geq 3$ and $\gamma_{sc}(C_n) = n - 1$, for all $n \geq 4$. We observe that, a Cycle of order n with a chord will yield two subcycles of order n_1 and n_2 such that C_{n_1} : $v_1, v_2, \ldots, v_i$ and C_{n_2} : $v_1, v_i, \ldots, v_n \rightarrow n = n_1 + n_2 - 2$. Let S_1 and S_2 be the secure connected dominating set for C_{n_1} and C_{n_2} respectively. And, S be the required secure connected dominating set of C(n, 1). We know that, $\gamma_{sc}(C_n) = n - 1$, for all $n \geq 4$. Without loss of generality, let $v_1 \in S_1$ and S_2 also $v_2 \in S_1$ and S_2. Therefore, $S = S_1 \cup S_2$. Here, $|S| = |S_1| + |S_2| - 2 \rightarrow |S| = n - 2$.

Claim: S is the minimum secure dominating set of C(n, 1).

Without loss of generality, let $S = \{v_1, v_3, \ldots, v_i, v_{i+1}, \ldots, v_{n-1}\}$. By previous theorem, we know that, $\gamma_c(C(n, 1))$ is at most $n - 3$. We get, $\gamma_{sc}(C(n, 1)) \geq n - 3$. Suppose if, $\gamma_{sc}(C(n, 1)) = n - 3$, then $V(C(n, 1))\backslash S = \{v_2, v_j, v_n\}$, where j:3, 4, …, i − 1, i + 1, …, n − 1. Here, if v_j is adjacent to v_2 (or) v_n, then secure domination fails. If not, then connectedness fails. This is a contradiction to our assumption. And so, $\gamma_{sc}(C(n, 1)) \neq n - 3$.

Therefore, $\gamma_{sc}(C(n, 1)) = n - 2$, for all $n \geq 5$

4 Methodologies

The proposed the IoT based hybrid honeypot system is used to monitor the malicious activity in the system. The proposed system clustering into three honeypot modules, first module is detecting the medium to high interaction in honeypot. Second is high interactive honeypot which can analysis the real services in the network. Finally last one is hybrid module to simulating the interaction in the system with real time devices [34–36]. Last part is also protecting the most exposed layer in the honeypot is Simple Object Access Protocol (SOAP) [37–39]. This module analysis the vulnerabilities of the protocol and detect the malicious activity. This module can deployed in independent way to connect the nodes with centralized control centres to transfer the collected data from different honeypot devices with secured manner. This can be moved to DOCKER framework easy deployment and simulations [40].

4.1 Overall Architecture

a) Data Server

This study is used to give a control centre more privileges on coordination on honeypot system and boosting of clusters deployment capability. The logs have been maintained over every entry of messages from honeypot whether it may be a input or output messages, it was maintained through the control centre for distribution of files through command control and server. The script which contains the python language has the powers to connect the remote host of the IP list with the libraries which contains it by utilising the multi-threaded concepts. The messages transferring through this libraries have secure shell key for running the specific commands and get the access over the honeypot as well as monitoring the systems too [41]. The honey pots are controlled by honeynet systems which delivers honeypots to various physical machines which the setup is shown in Figs. 2 and 3. The nodes which have less interaction with honeynet system has able to communicate with master systems to manage the honeypots [42]. The uninteracted honeypot systems directly communicate the master the systems which is running on real firmware can handle the multiple request from core and able to work as a multicore honeynet systems as well as single honeynet systems.

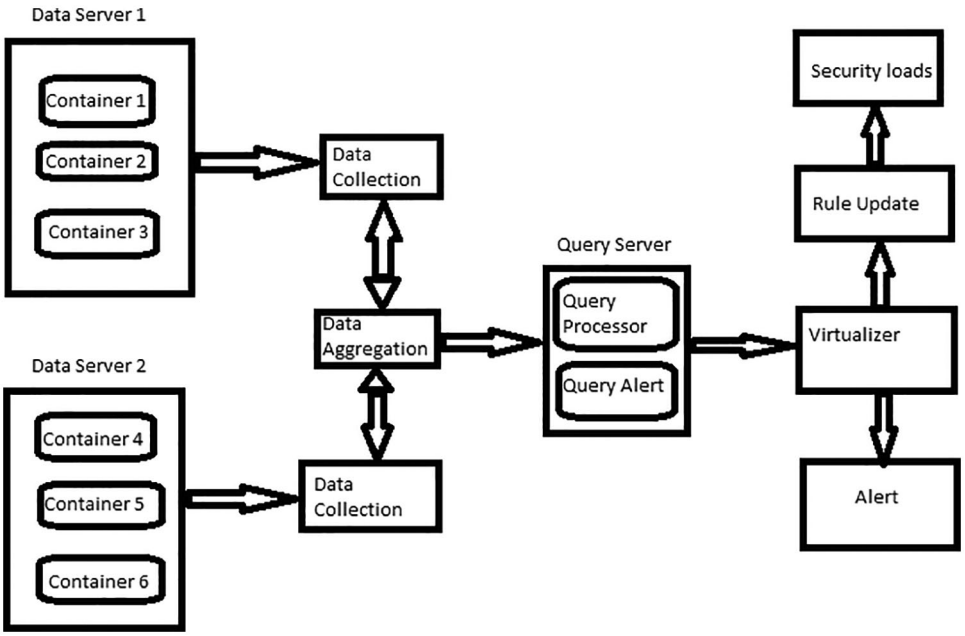

Figure 2: Self learning honeypot architecture

4.1.1 Data Collection

Honeypots are also used to inject in a router remotely for finding the vulnerabilities and it degrades the performance of the router, which also provides unprotected SOAP for up gradation of services. The request packet 36126 is used to monitor the port using the Uphp services. After the delivering of the service packet, it is used to monitor the services of the routers remotely with arbitrary commands which is shown in Figs. 4 and 5. As from the literature survey we collected the vulnerabilities for the systems in a different networks which have created through honeypot. The self learning honeypot have been structured in the following names: core module, service module, daemon module and monitor module. The proposed honeypot as able to learn the behaviour and keep is operation very stable at all time. If any discrepancy occurs in between time, it may able to restart the entire process within a fraction of time and able to collect all the details from last known

configuration. It can record the entire process by utilising its ability, it can re assures the stability of monitoring which is shown in Fig. 6.

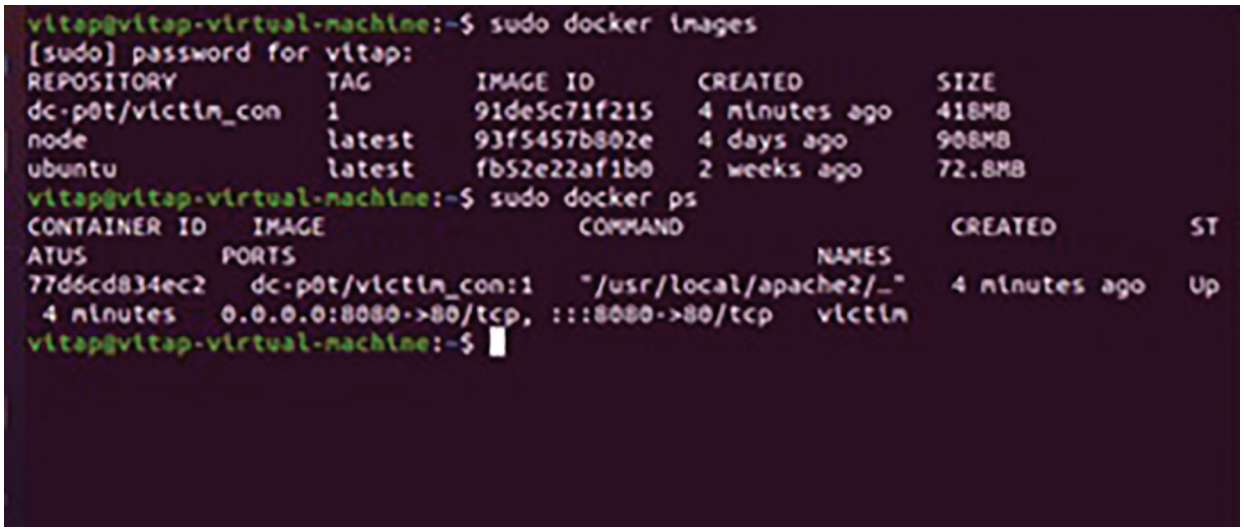

Figure 3: Docker environment setup

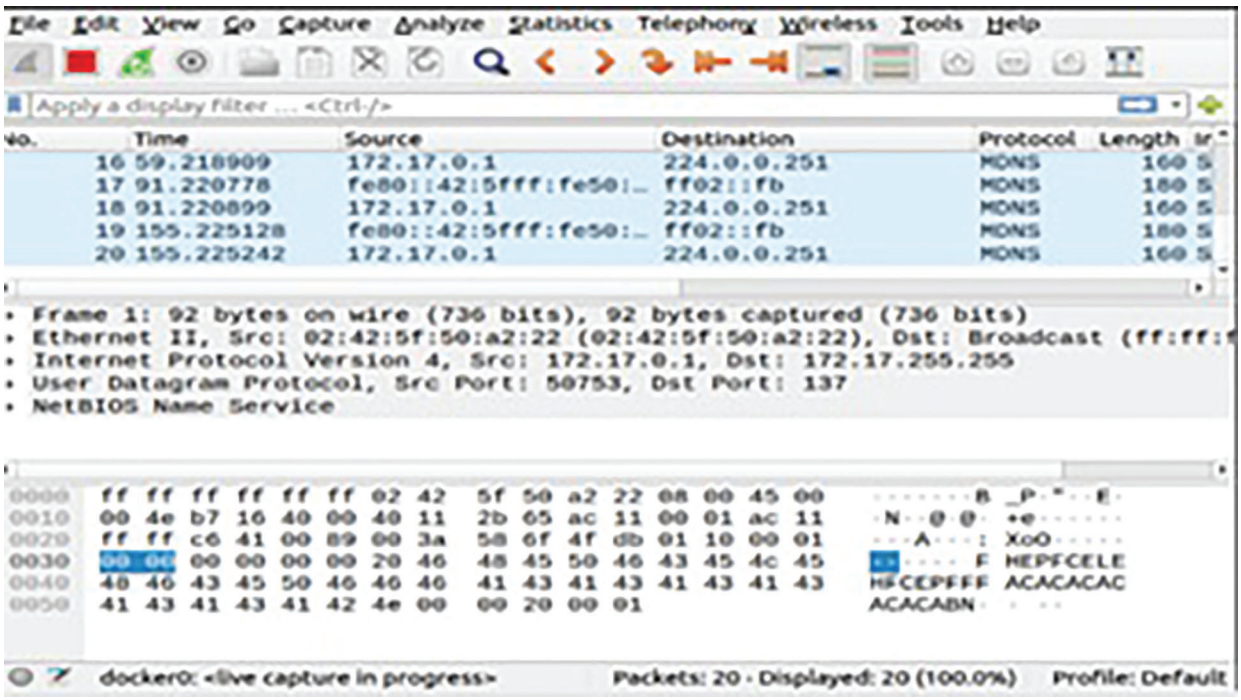

Figure 4: Packet accessing

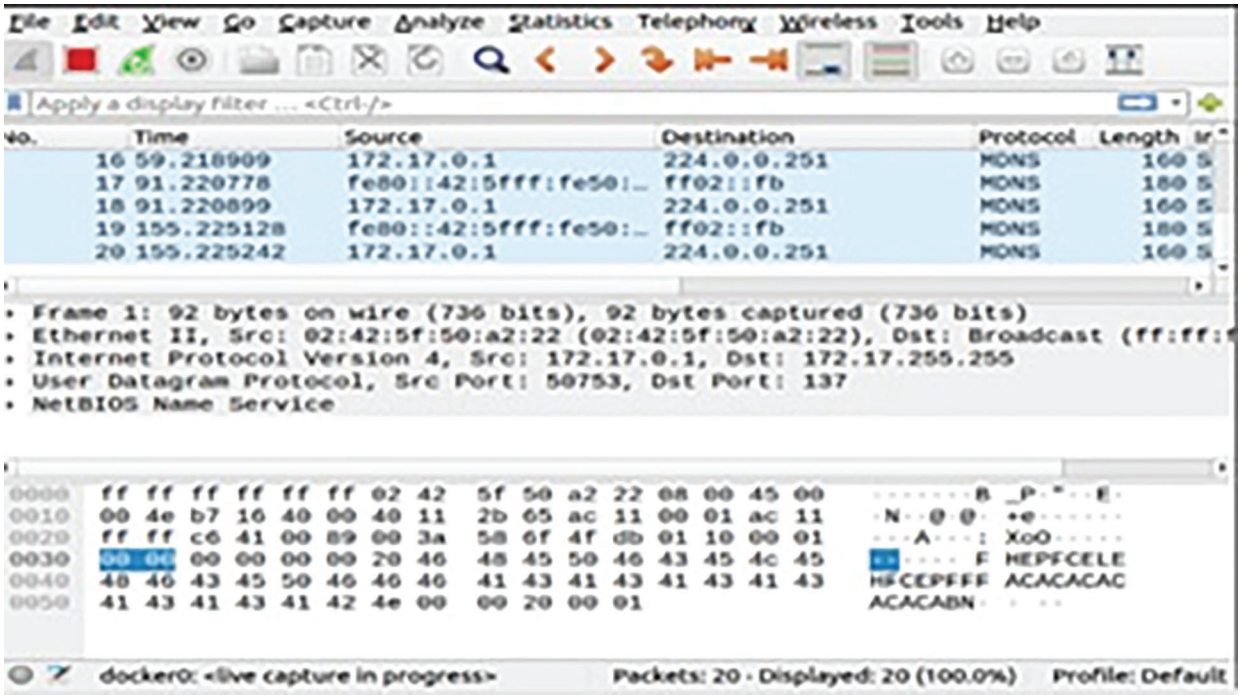

Figure 5: Packet accessing

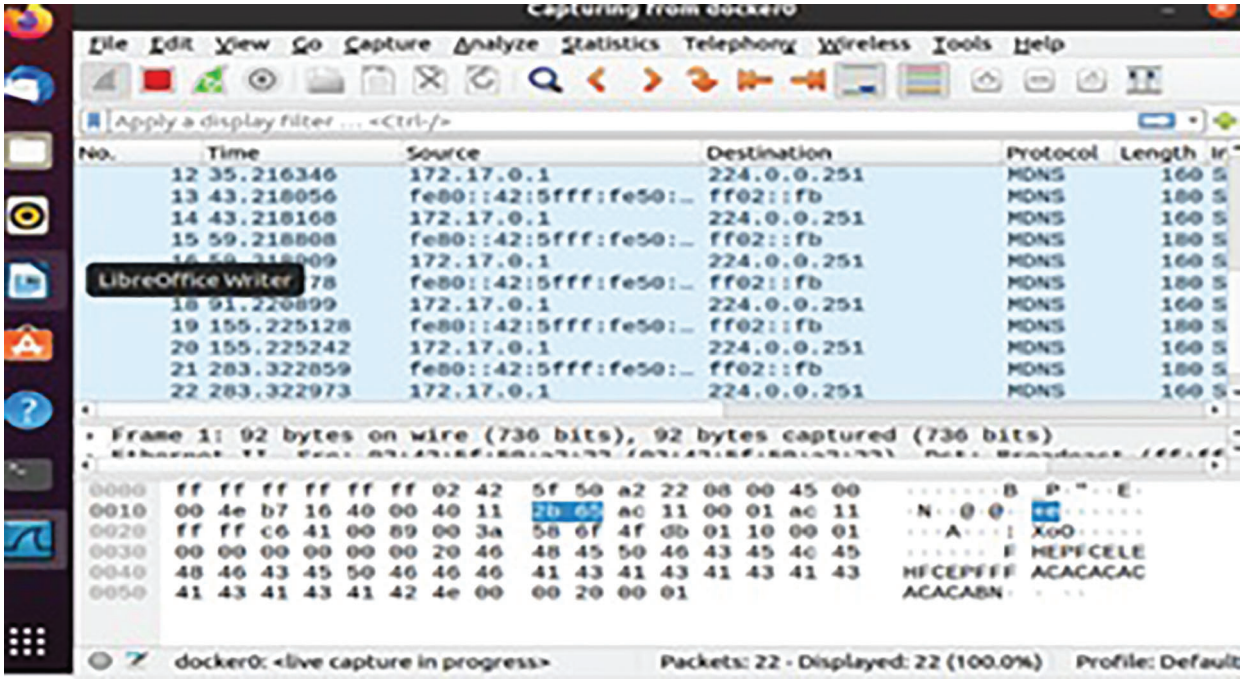

Figure 6: Live packet capturing from docker

4.1.2 Data Aggregation

In this data aggregation section, data's collected through the different servers. There are different modules present in this data aggregation part which has malware sample processing, web service, service

and log modules. This modules are used to serve to collect the data over the servers. It keeps the record of each logs with run time handling events, it is the recursive process to check the logs events eventually after the end of the day. It keeps the records upto date. The SOAP service module is used to get the request updations over Uphp server handler which his represented through Fig. 7. If any malware samples are downloaded then the malware processing module will take the action for processing the downloaded samples. The log files modules are used to audit the input and output response request.

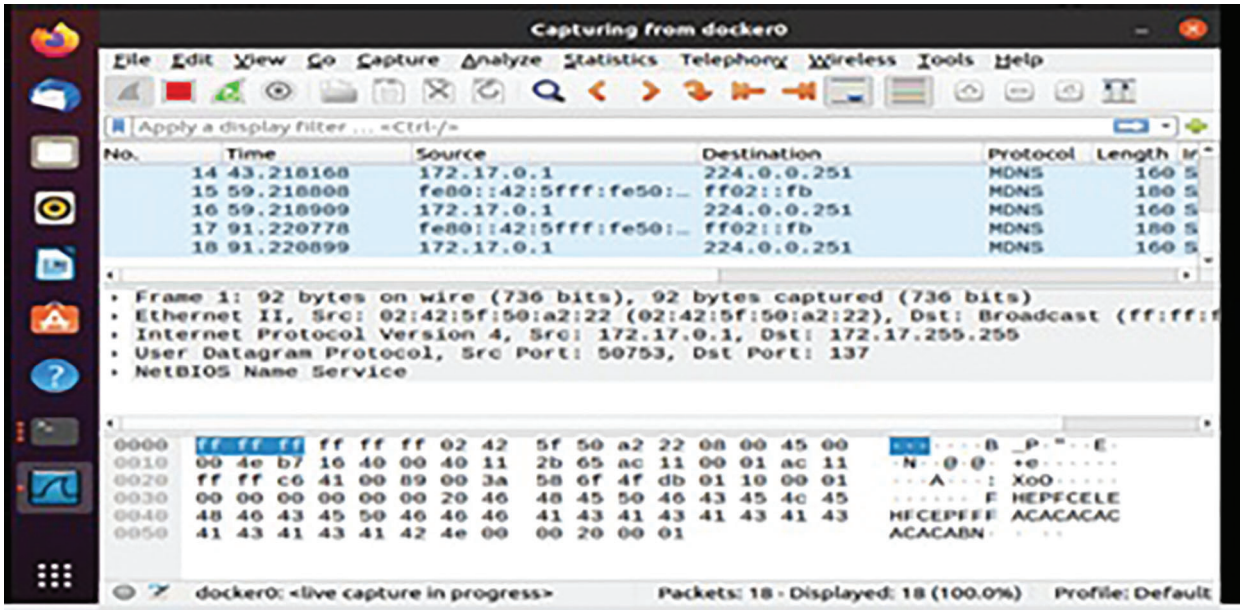

Figure 7: Frame analysis in docker

4.2 Query Server

In this query server section sockets are used to communicate between web servers and log servers. The download-virus-by-requests function is used to download samples after the download resource information has been extracted. The function looks for the output directory and sample name first. If the sample name already exists, the renaming duplicate-virus function is used to rename the sample by appending the sample suffix. The code then makes a request to the requests library for sample downloads. If the sample download is successful, the deep-analyse function is used to perform a more in-depth analysis, which involves examining the sample content to see if a "big Trojan downloaded by little Trojan" scenario occurs which is shown in Fig. 8. This part additionally characterizes the unmistakable copy test capacity to eliminate the deduplication of the example, which computes the hash worth of the example by calling the md5sum work.

4.2.1 Rule Updation

The SOAP device information in the SoapXML subdirectory is modelled after the susceptible router. It contains information such as the device type, model, URL, UUID, serial number and list and so on. This file returns genuine device configuration information in response to SOAP service searches for port 37215 which is shown in Fig. 9. The operations of the above pieces combine to form the honeypot's core, completing the honeypot's basic functions and simulating real systems and services.

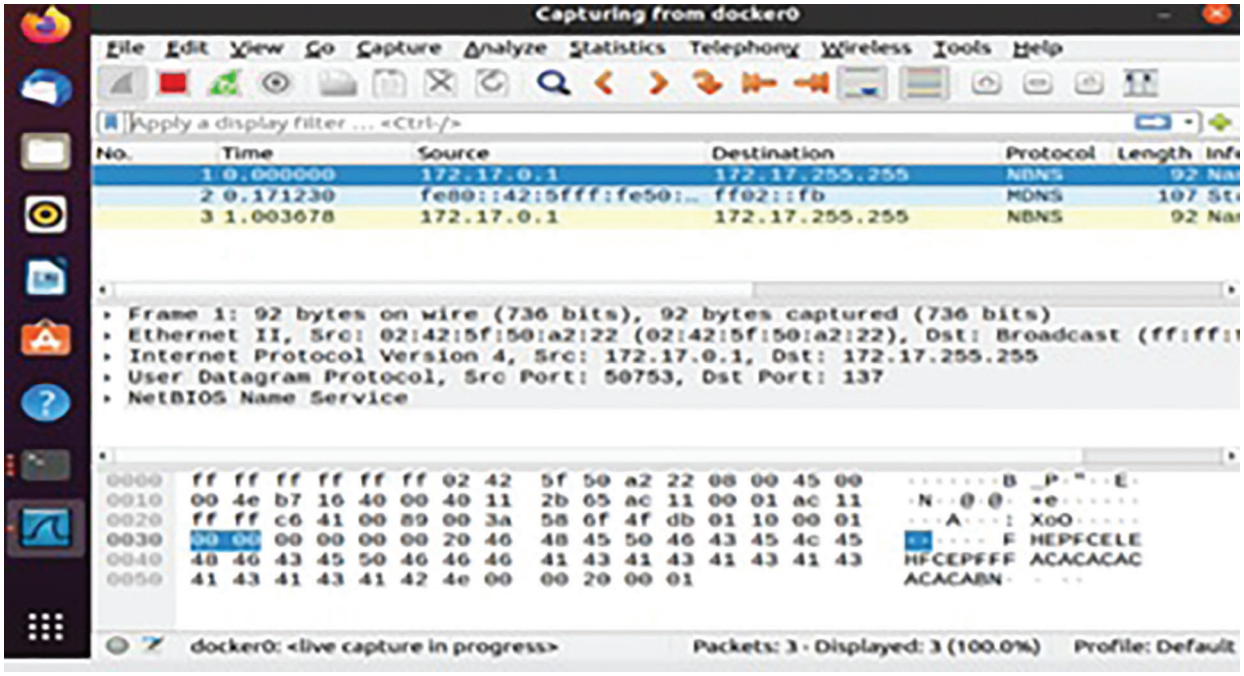

Figure 8: Deep level packet inspection in docker

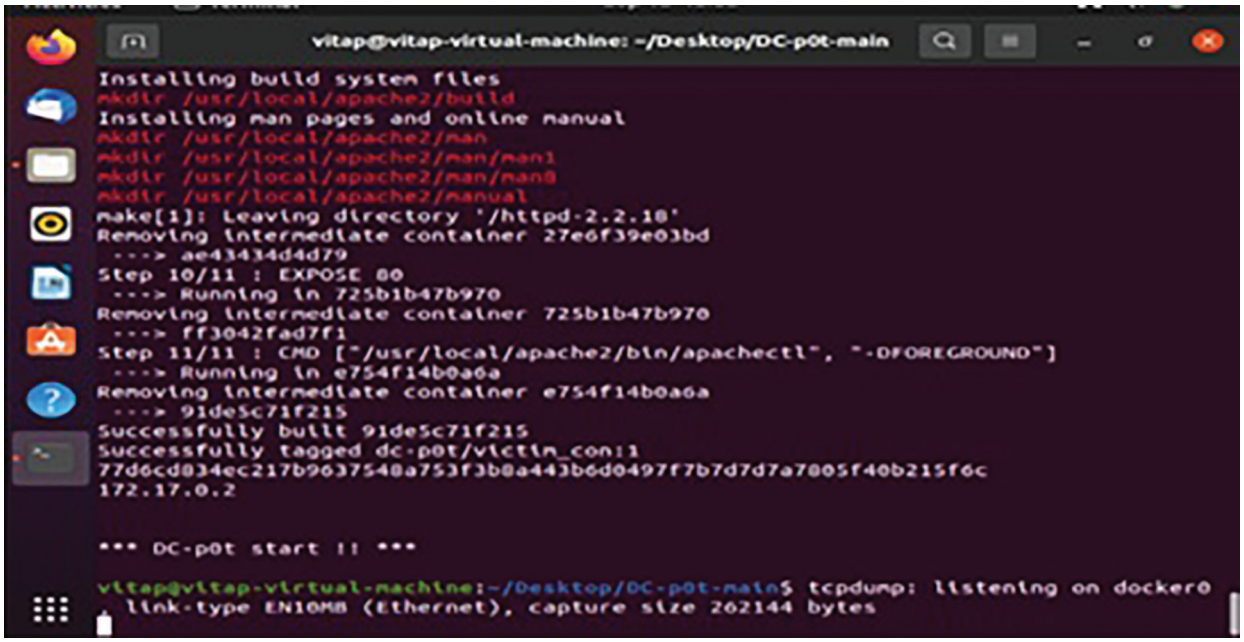

Figure 9: Live data access from different containers in data server

Honeypot Daemon: This daemons are probably used to start, close, restart, open and initializing. It also has the ability to control the entire honeypot system. In the flow, the empty file is routed to different locations because most of the daemons are already closed with the terminal. Mostly pids are used to store the processing address of entire environment. To ensure singleton mode, the static information is prescribed if

the access with the system. The Daemon instance is initialised using the function demonize. Automatically flushed the buffer, and the stream which is deleted after determining whether it already exists via pidfile. Finally, the atexit.register and signal.signal routines ensure that the pidfile is removed at the end of the process. The SystemExit exception is thrown by the static function.

Daemon Service: This service module which inhibits the class of its own class from the scar libraray, which defines the class MyDaemon. When the instance is up and running, it begins the honeypot core service. The daemon is created with the instance for the function, the instance's associated functions govern the commencement of the core and honeypot daemon. The external interface of the daemon is Potcore Daemon. The sub process function called Popen is used to invoking the main function. That is, instead of calling the external instance of the class MyDaemon, the instance of PotCore is invoked. To implement the indirect call, the instance of PotCore calls the instance of MyDaemon through its own method. The honeypot must be virtualized and transplanted to allow for quick deployment of the transplant and to avoid harmful activity after it has been cracked. The honeypot is then bundled into a black box that is simple to use and deploy. Docker is an application packaging technology. Images, containers, and warehouses are the primary components. An image is a file that the Docker engine recognises as containing the fundamental contents of a container. The following items that the developer requires can be found in an image: operating system, specialised environment, apps, and so on. After running, the container is an image. The image is saved in the repository. In order to be deployed quickly, the image must be streamlined as follows:

1) Make the fundamental image as good as it can be. Ubuntu 16.04 is used in the honeypot's basic image, and it also includes several packages required by Python 3.6 at the same time, preventing the image from being downloaded at a lower rate during packaging.
2) Join the DockerFile commands together. To save space, many commands are combined into a single RUN command, the number of image layers is minimised, and unneeded components are deleted.
3) Improve your company's performance. The image cache is maximised, Docker builds are performed on a dedicated machine, and huge dependent libraries are kept separate from regularly changing own code.
4) Improve the run command. When using the apt command, suggested dependencies are avoided.

During honeynet's rapid deployment, the master node sends commands to each physical host's master programme, and each physical host connects to the remote image warehouse to download and run the appropriate honeypot image.

5 Evaluation

The self learning honeypots have been deployed and collected the results and traces have been monitored with time events and the differentiation between normal traffic and attack traffic has clearly stated with Figs. 10–12. The resultant graphs have been generated with continuous time frame event which consists of five minutes interval in between them. Through Tabs. 1–3 shows the detection of packets which flows in a particular time event. The difference between the time frame and packet flow is calculated. It clearly indicates that abnormal traffic can be detected through our graph based self learning honeypot.

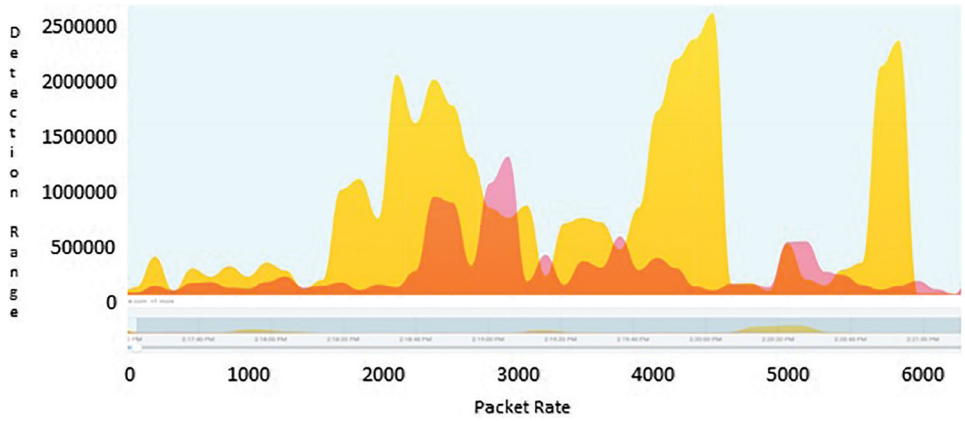

Figure 10: Normal traffic *vs.* attack traffic

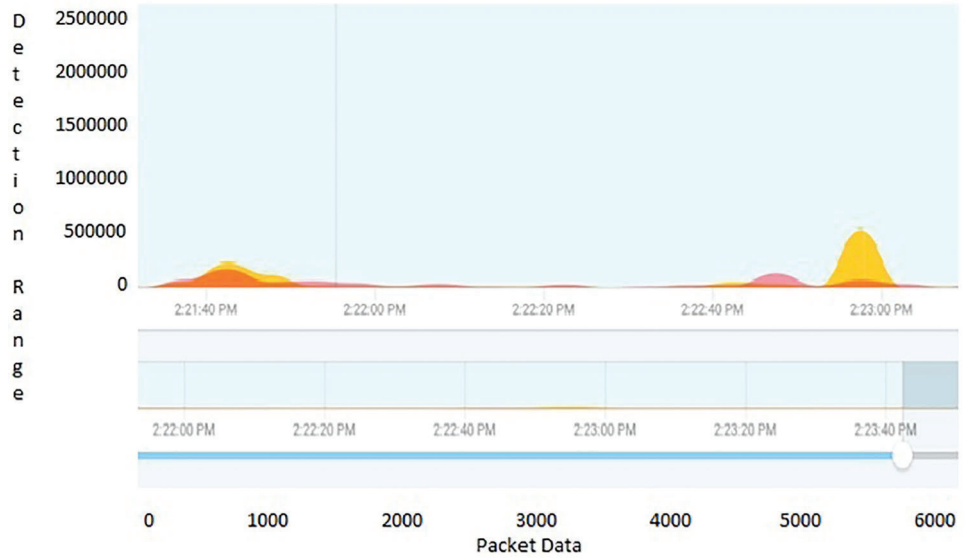

Figure 11: Normal traffic in proposed environmental setup with different time stamp

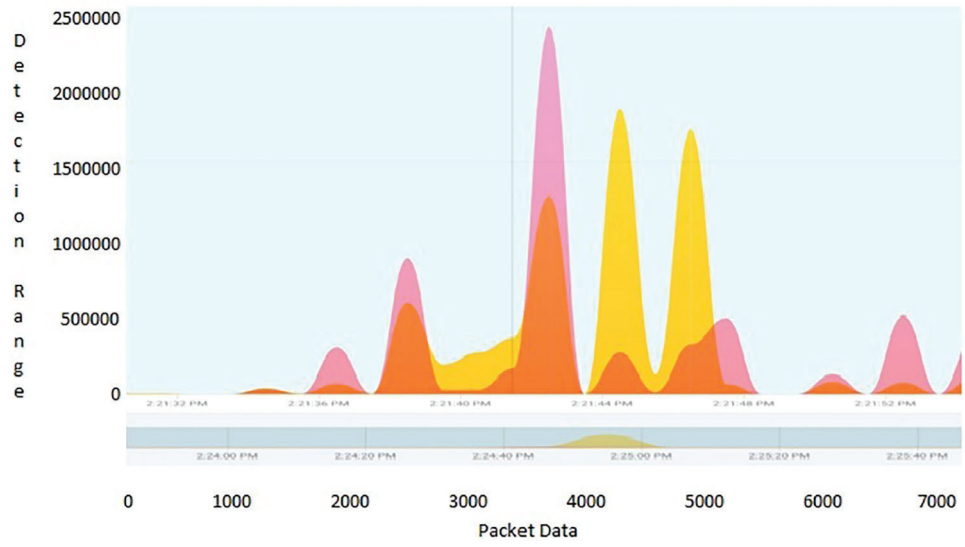

Figure 12: Abnormal traffic flow detected through self learning honeypot

Table 1: Abnormal traffic flow indication through packet data

S.No.	Detection range of packets	Packet rate
1.	0	0
2.	500	1000
3.	20000	2000
4.	1000000	3000
5.	1500000	4000
6.	1800000	5000
7.	5000	6000

Table 2: Normal traffic flow indication through packet data

S.No.	Detection range	Packet rate	Time
1.	0	0	2.22
2.	1000	1000	2.21.40 to 2.22.00
3.	500	2000	2.22.00 to 2.22.20
4.	150	3000	2.22.20 to 2.22.40
5.	75	4000	2.22.40 to 2.23.00
6.	39	5000	2.23.00 to 2.23.20
7.	22	6000	2.23.20 to 2.24.00

Table 3: Abnormal traffic flow detection with honeynet setup with different time stamp

S.No.	Detection range	Packet rate	Time
1.	0	0	2.22
2.	50	1000	2.21.40 to 2.22.00
3.	10000	2000	2.22.00 to 2.22.20
4.	700000	3000	2.22.20 to 2.22.40
5.	2400000	4000	2.22.40 to 2.23.00
6.	1600000	5000	2.23.00 to 2.23.20
7.	20000	6000	2.23.20 to 2.24.00
8.	500000	7000	2.23.25 to 2.25.00

6 Conclusion

The honeypots are mostly used in cyber defence strategies, especially in reconnaissance for finding the information about the attacker. The honeypots deployed for this work is under the environment of honeynet system, so the entire applications are controlled by this honeynet system including self learning honeypots. In this paper, the proposed self learning honeypot which is completely based on graph theorems, the strategies employed for this self learning honeypot is routed from connected dominating set graphs, theorems are proposed results $\gamma(C(n, 1))$, $\gamma_c(C(n, 1))$, $\gamma_{sc}(C(n, 1))$ solely for honeypot discovery. Self Learning

honeypot is setup with honeynet systems which has behaviour understanding and model analysis with reluctant priority based application. It is evident from the obtained results over the figures and tables our proposed graph theory based algorithm detects the attack traffic and its pattern with the detection accuracy of 98.2%.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] C. Elmer, L. Rose and R. Sergio, "Secure domination in the join of graphs," *Applied Mathematical Sciences*, vol. 8, no. 105, pp. 5203–5211, 2014.

[2] C. Berge, "Theory of graphs," *Methuen & Co Ltd.*, London, 1962.

[3] O. Ore, "Theory of Graphs," *Colloquium Publications (American Mathematical Society)*, vol. 38, 1962.

[4] E. J. Cockayne, P. J. P. Grobler, W. R. Grundlingh and J. H. Van Vuuren, "Protection of a graph," *Utilitas Mathematica*, vol. 67, no. 2, pp. 19–32, 2005.

[5] E. Sampathkumar and H. B. Walikar, "The connected domination number of a graph," *Journal of Mathematical and Physical Sciences*, vol. 13, no. 6, pp. 607–613, 1979.

[6] G. C. Amerkhan, S. C. Sergio and S. A. Imelda, "Secure connected domination in a graph," *International Journal of Mathematical Analysis*, vol. 8, no. 42, pp. 2065–2075, 2014.

[7] G. Chartrand, L. Lesniak and P. Zhang, "Graphs and Digraphs," in *Fourth Edition, A Chapman and Hall Book*, CRC Press, USA, 2005.

[8] T. W. Haynes, S. T. Hedetniemi and P. J. Slater, in *Domination in Graphs*, Marcel Dekker New York, 1998.

[9] T. W. Haynes, S. T. Hedetniemi and P. J. Slater, in *Fundamentals of Domination in Graphs*, Marcel Dekker New York, 1998.

[10] C. Pretty Diana Cyril, J. Rene Beulah, N. Subramani, P. Mohan, A. Harshavardhan *et al.,* "An automated learning model for sentiment analysis and data classification of twitter data using balanced CA-SVM," *Concurrent Engineering: Research and Applications*, vol. 29, no. 4, pp. 386–395, 2021.

[11] J. Baliga, R. W. A. Ayre, K. Hinton and R. S. Tucker, "Green cloud computing: Balancing energy in processing, storage, and transport,"*in Proceedings of the IEEE*, vol. 99, no. 1, pp. 149–167, 2011.

[12] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta *et al.,* "On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[13] K. Mershad, H. Artail, M. A. R. Saghir, H. Hajj and M. Awad, "A study of the performance of a cloud datacenter server," *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 590–603, 2017.

[14] H. Yan, W. Huaimin, L. Xi, W. Yuan, L. Dongsheng *et al.,* "Cost-efficient consolidating service for aliyun's cloud-scale computing," *IEEE Transactions on Services Computing*, vol. 12, no. 1, pp. 117–130, 2016.

[15] S. Manikandan, S. Satpathy and S. Das, "An efficient technique for cloud storage using secured de-duplication algorithm," *Journal of Intelligent & Fuzzy Systems*, vol. 41, no. 2, pp. 2969–2980, 2021.

[16] A. T. Lo'ai and B. Waseem, "A mobile cloud system for different useful applications," in *2016 IEEE 4th Int. Conf. on Future Internet of Things and Cloud Workshops (FiCloudW)*, Austria, pp. 295–298, 2016.

[17] C. Chuan-Yen, C. Yen-Lin, K. Kun-Cing and Y. Shyan-Ming, "Real-time pedestrian detection technique for embedded driver assistance systems," in *2015 IEEE Int. Conf. on Consumer Electronics (ICCE)*, Las Vegas, USA, pp. 206–207, 2015.

[18] S. Neelakandan, "Social media network owings to disruptions for effective learning," *Procedia Computer Science*, vol. 172, no. 5, pp. 145–151, 2020.

[19] N. Subramani and D. Paulraj, "A gradient boosted decision tree-based sentiment classification of twitter data," *International Journal of Wavelets, Multiresolution and Information Processing*, vol. 18, no. 4, pp. 1–21, 2021.

[20] B. Li, R. Lu, G. Xiao, T. Li and K. -K. R. Choo, "Detection of false data injection attacks on smart grids: A resilience-enhanced scheme," *IEEE Transactions on Power Systems*, 2021, https://doi.org/10.1109/TPWRS.2021.3127353.

[21] D. K. Jain, P. Boyapati and J. Venkatesh, "An intelligent cognitive-inspired computing with big data analytics framework for sentiment analysis and classification," *Information Processing & Management*, vol. 59, no. 1, pp. 1–15, 2022.

[22] S. Neelakandan and D. Paulraj, "An automated exploring and learning model for data prediction using balanced ca-svm," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 5, pp. 1–12, 2020.

[23] C. Ramalingam, "An efficient applications cloud interoperability framework using i-anfis," *Symmetry*, vol. 13, no. 2, pp. 268, 2021.

[24] A. O. Almashhadani, M. Kaiiali, S. Sezer and P. O'Kane, "A Multi-classifier network-based crypto ransomware detection system: A case study of locky ransomware," *IEEE Access*, vol. 7, no. 11, pp. 47053–47067, 2019.

[25] A. C. F. Petri and D. F. Silva, "Towards logical association rule mining on ontology-based semantic trajectories," in *IEEE Int. Conf. on Machine Learning and Applications (ICMLA)*, Florida, pp. 586–591, 2020.

[26] R. Kamalraj, M. Ranjith Kumar, V. Chandra Shekhar Rao, R. Anand and H. Singh, "Interpretable filter based convolutional neural network (IF-CNN) for glucose prediction and classification using PD-SS algorithm," *Measurement*, vol. 183, pp. 109804, 2021.

[27] M. Bahrami and M. Singhal, "A Light-weight permutation based method for data privacy in mobile cloud computing," in *IEEE Int. Conf. on Mobile Cloud Computing, Services, and Engineering*, San Francisco, CA, USA, pp. 189–198, 2015.

[28] N. Bansal and M. Dutta, "Performance evaluation of task scheduling with priority and non-priority in cloud computing," in *IEEE Int. Conf. on Computational Intelligence and Computing Research*, Coimbatore, TN, INDIA, pp. 1–4, 2014.

[29] T. Ravichandran, "An efficient resource selection and binding model for job scheduling in grid," *European Journal of Scientific Research*, vol. 81, no. 4, pp. 450–458, 2012.

[30] C. Zhang, R. Green and M. Alam, "Reliability and utilization evaluation of a cloud computing system allowing partial failures," in *IEEE 7th Int. Conf. on Cloud Computing*, India, pp. 936–937, 2014.

[31] M. Bahrami and M. Singhal, "A dynamic cloud computing platform for eHealth systems," in *17th Int. Conf. on E-Health Networking, Application & Services (HealthCom)*, Boston, USA, pp. 435–438, 2015.

[32] P. Manjula and S. B. Priya, "Intelligent chimp metaheuristics optimization with data encryption protocol for wsn," *Intelligent Automation & Soft Computing*, vol. 32, no. 1, pp. 573–587, 2022.

[33] G. Anitha and B. P. Sankarlingam, "Vision based real time monitoring system for elderly fall event detection using deep learning," *Computer Systems Science & Engineering*, vol. 42, no. 1, pp. 87–103, 2022.

[34] N. Kanagavalli and P. Baghavathi, "Social networks fake account and fake news identification with reliable deep learning," *Intelligent Automation & Soft Computing*, vol. 33, no. 1, pp. 191–205, 2022.

[35] M. Prakash and K. K. Dhawan, "Fault tolerance-genetic algorithm for grid task scheduling using check point," in *in IEEE Sixth Int. Conf. on Grid and Cooperative Computing*, China, pp. 676–680, 2007.

[36] A. Youseef, A. Saleh, O. I. Khalaf and U. Sakthi, "Improved metaheuristics-based clustering with multihop routing protocol for underwater wireless sensor networks," *Sensors*, vol. 22, no. 4, pp. 1–15, 2022.

[37] P. Mohan and R. Thangavel, "Resource selection in grid environment based on trust evaluation using feedback and performance," *American Journal of Applied Sciences*, vol. 10, no. 8, pp. 924–930, 2013.

[38] A. Arun, R. R. Bhukya, B. M. Hardas and T. Ch, "An automated word embedding with parameter tuned model for web crawling," *Intelligent Automation & Soft Computing*, vol. 32, no. 3, pp. 1617–1632, 2021.

[39] D. Arivudainambi and S. Sibi Chakkaravarthy, "LION IDS: A meta-heuristics approach to detect ddos attacks against software-defined networks," *Neural Computing & Applications*, vol. 31, no. 2, pp. 1491–1501, 2019.

[40] D. Arivudainambi, K. A. Varun Kumar, S. Sibi Chakkaravarthy and P. Visu, "Malware traffic classification using principal component analysis and artificial neural network for extreme surveillance," *Computer Communications*, vol. 147, no. 2, pp. 50–57, 2019.

[41] D. Arivudainambi, K. A. Varun Kumar, R. Vinoth Kumar and P. Visu, "Ransomware traffic classification using deep learning models: Ransomware traffic classification," *International Journal of Web Portals*, vol. 12, no. 1, pp. 1–11, 2020.

[42] D. Arivudainambi, K. A. Varun Kumar and S. C. Satapathy, "Correlation based malicious traffic analysis system," *International Journal of Knowledge-Based and Intelligent Engineering Systems*, vol. 25, no. 2, pp. 195–200, 2021.