Intrusion Detection and Response System Inspired by the Defense Mechanism of Plants

Rupam Kumar Sharma[1], Biju Issac[2], Senior Member, IEEE and Hemanta Kumar Kalita[3], Member, IEEE

[1]Department of CSE & IT, Assam Don Bosco University, Guwahati 781017, India

[2]Department of Computer and Information Sciences, Northumbria University, UK

[3]Department of IT, North-Eastern Hill University, Shillong 793033, India

Corresponding author: Rupam Kumar Sharma (e-mail: Rupam.Sharma@dbuniversity.ac.in).

ABSTRACT The security of resources in a corporate network is always important to the organization. For this reason, different techniques such as firewall, Intrusion Detection Systems are important. Years of long research have resulted in the contribution of different advancements in these techniques. Artificial Intelligence, machine learning techniques, soft computing techniques and bio-inspired techniques have been efficient in detecting advanced network attacks. However, very often different new attacks are mostly successful in breaching these detection techniques. This very reason has been a motivation for us to explore the biological aspects and its defense mechanisms for designing a secure network model. After much study, we have identified that plants have a very well established and evolved detection and a response mechanism to pathogens. In this research work we have proposed and implemented a network attack detection and a response model inspired by plants. It is a three-layered model in analogy to three-layer defense mechanism of plants to pathogens. We further have tested the proposed model to different network attacks and have compared the results to Open Source Intrusion Detection System, Snort. The experimental results also establish that the model is competent to detect and trigger an automated response whenever required.

INDEX TERMS Bio-inspired computing, Intrusion detection system, Fuzzy Logic, Network attacks

I. INTRODUCTION

The network security of an organization is accomplished by installation of different security software and hardware systems such as firewalls, Intrusion Detection Systems, Honeypot etc. Software based Intrusion Detection Systems (IDS) are trained with different training corpus. Learning incorporated in the IDS during training may either be static i.e. non-evolutionary or dynamic i.e. evolutionary. Evolutionary means that the learning evolves with every new encounter of instances. Such learning is also known as online learning and IDS using such approaches are known as online learning-based IDS. The literature survey shows that online IDS are susceptible to different learning attacks [1,2]. The attacker, therefore, with some knowledge of the learning algorithm can always try to influence the classification behavior of the algorithm. So, it is important that any detection system have a subsequent response component so that whenever the detection engine fails, the response engine is activated to protect the network resource from further compromise. This behaviour of attack detection and a subsequent response generation is most obvious among living organisms such as human body. Other mammals such as bats design an efficient network attack detection and a response system and so the living organisms in nature can play an important role in providing new inspiration. This led to the motivation of exploring different organisms in nature such as pitcher plants, fish, bats, vagotomized rats etc. and plants [3,4,5]. After extensive study of different living creatures in nature we discovered that plant is one such organism with a well-established evolutionary multi-layer defense system along with a bio-molecular mechanism of information flow called SAR (Systemic Acquired Resistance) as in [6,7].

Studying the defense mechanism of plants to build a defense and a response model in computer networks is the first of its kind amongst the computer researchers. The defense and the response model designed, developed and implemented has been termed as PIRIDS i.e. Plant-based

Inspiration of Response in IDS (Intrusion Detection System).

II. PLANTS DEFENSE MECHANISM

Plants have a very well established and evolved defense mechanism.

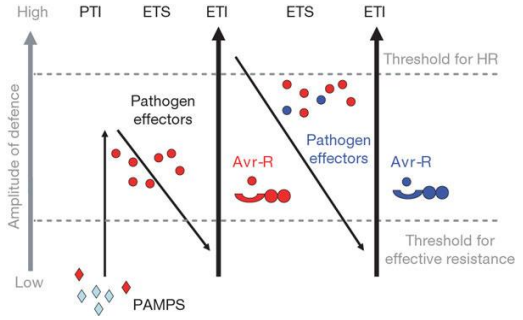

FIGURE 1. The zig-zag plant defense model [9]

The plants defense model can be represented as four phase zig-zag model [9] shown in Figure 1. In phase 1, the PRR (Pattern Recognition Receptors) of the plant recognizes any previously known PAMP (Pathogen Associated Molecular Pattern) or MAMP (Microbes Associated Molecular Pattern). The PAMP are molecules associated with pathogens and these molecules are no way associated with the molecules of plants. The pattern recognition receptors are technically a sequence of gene codes. Whenever, a pathogen tries to enter the plant body, the PRR tries to look for any matching gene sequence of the pathogen with itself. If it finds one, it identifies the pathogen and triggers primary response. One such response is closing the stomata present in the leaves of the plant so that more microbes cannot enter the plant body through the pores. The pattern recognition receptors in plants have undergone hundreds of years of evolution. With evolution the ability of pattern recognition receptors to detect varying number of pathogens have increased significantly.

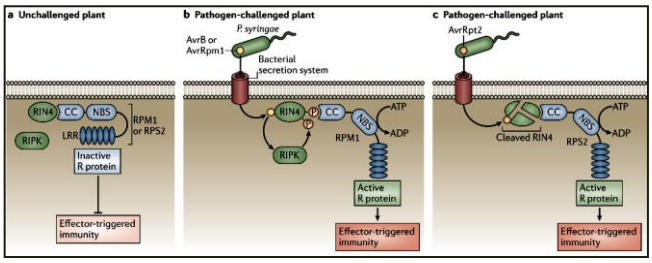

FIGURE 2. Guard model in plants

The pathogen molecules are also known as effectors. As soon as the PRR recognizes these effectors, a PTI (Pattern Triggered Immunity) is initiated. The consequence of such activity is to halt effector proliferation by closing the stomata in the leaf, deposition of callose for increasing cell wall thickness and accumulation of defense related proteins such as glucanases. Callose is a plant polysaccharide made by the Glucan Synthase-Like gene (GLS) within a plant and it is produced to act as a temporary cell wall in response to stimuli such as stress or damage. However, all the type of pathogens may not be recognized by PRR. Pathogens which successfully breach PRR, enters the plant body and targets the critical proteins of the plant. These critical proteins are always guarded by another set of proteins called the guard proteins. This form of defense mechanism is the second layer of defense in plants. The guard proteins observe for any changes in the critical proteins due to phosphorylation by effectors. These guard proteins are the R-genes in plants and encode NB-LRR protein as shown in Figure 2. If any changes are observed, localized defense are initiated which results in Hyper Response (HR) in plants [10]. This type of response is popular in the presence of multiple pathogens. This approach is called the guard model in plants [11]. In HR, the plants kill his local section such that the infection cannot spread further from that region to other uninfected parts of the plant. This is the third layer of defense that plants initiate as the highest response to pathogens.

Induced systemic resistance in plants

Plants also adopts a mechanism called systemic acquired resistance (SAR). By the mechanism of SAR, the information about primary pathogens are carried across distal parts for induced resistance, so that the plants can immediately respond to second time encounter of the known pathogens. Even though plants do not have a circulatory system the message propagation from one end to the other is well accomplished.

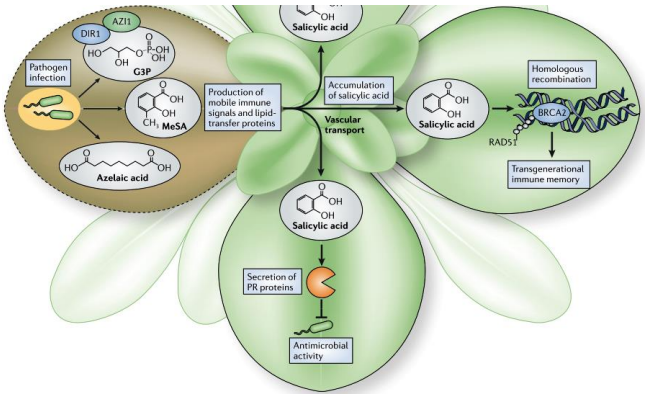

FIGURE 3. SAR in plants [12]

The beginning of SAR is marked by increased density of salicylic acid in the phloem. Phloem is the living tissue in plants that conduct foods made in the leaves to all other parts of the plant. Whenever, there is an infection by pathogen, there results in production of mobile immune signals such as Methyl Salicylic Acid (MeSA), Azelaic acid, Glycerol-3-Phosphate (G3P) and the lipid transfer proteins Defective in Induced Resistance 1 (DIR1) and Azelaic Acid (AZI1). These signals travels through the phloem in the plant body in

the form of Salicylic acid. Accumulation of SA in the distal part induces pathogenesis related (PR) proteins with antimicrobial property. This metabolic reaction is shown in Figure 3. As can be seen from the figure, after primary pathogen infection as shown in the light brown shaded leaf, and because of ETI (Effector Triggered Immunity), different molecules such as G3P, MeSA, and Azelaic acid are formed. Any protein present in a plant body is termed as SAR protein provided the protein have a role in building resistance in the plant. Literature study indicates that varying level of SA concentration in plant body is an indication of any pathogen presence in the plant body. The amount of SA found in the plant body is considerably high during pathogen infection.

III. FUZZY LOGIC

Proposed for the first time in 1965 by Lotfi. A. Zadeh while working in the electrical engineering in the university of California, fuzzy set and fuzzy systems started gaining wide popularity during late 90's. Real life problems which were otherwise difficult to represent in crisp form were started to represent using fuzzy approach. Fuzzy indicates vagueness and the degree of uncertainty [32,34,45]. In classical mathematics an object either belongs to a set of Universe say 'U' or don't belongs to the set U. If the element belongs to the set than it is 1 else 0. This set of objects is called the 'crisp set'. The membership of the object x to set A can be indicated as shown below.

$$\mu_A(x) = \quad 1, \text{ if } x \in A \text{ and } 0 \text{ if } x \notin A$$

The degree of measure in a given context can be quantified by high, moderate and low. This degree of association can be technically coined as membership value [33]. Mathematically it can be formulated as shown in equation below.

$$\mu_A : x \rightarrow [0,1]$$

μ_A:x indicates the degree of membership of x to set A. A minimum degree value of 0 indicates that x is least bound to set A and a value of 1 indicates that x is strongly bound to set A. Any other value between 0 and 1 indicates the varying degree of strength by which x is bound to set A.

As seen from figure 4, the boundary of the crisp set is rigid i.e. the membership function value can be either 0 or 1 and no other values between 0 and 1. But this is not true in cases of fuzzy set. The fuzzy set have values in closed interval [0,1] indicating the membership degree may vary from 0 to 1. The pattern variation of this membership degree may be triangular or a linear transformation. A membership degree of 1 indicates the highest degree of belonging of an element to a given set/class whereas a membership degree of 0 indicates the most loose bound of the element to the given set/class.

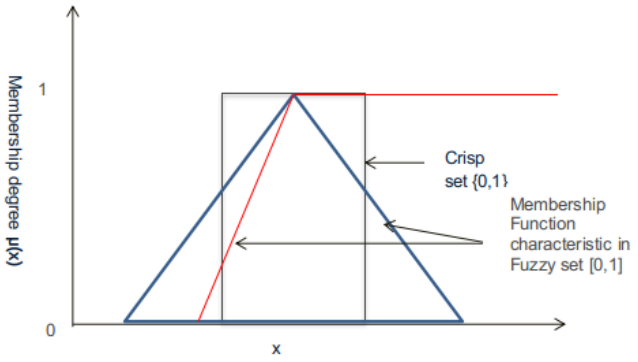

FIGURE 4. Classical set and fuzzy set boundary

Fuzzy set representation: A fuzzy set is represented by an ordered pair where the first element of the ordered pair represents the element belonging to a set and the second element represents the degree of membership of the element to the set. Mathematically it can be represented as in equation below [32,33].

$$\acute{A} = \{(x, \mu_{\acute{A}}(x) \mid x \in X\}$$

The membership function can be standard functions such as Gaussian or any user defined function in requirement to the problem domain.

IV. PROPOSED METHOD OF PIRIDS

PIRIDS is an abbreviation of Plant-based Inspiration of Response in IDS. It is a three-layered bioinspired detection and a response method derived as an inspiration from the plant biology. The defense model in plants can be shown concisely as a three-layered approach as illustrated in the figure 5 and in previous work of the authors [8]. However, the work in [8] was confined to the proposed theoretical model without any practical implementation to establish the proposed model. The research work presented in this paper establish the validity of the proposed model with experimental results.

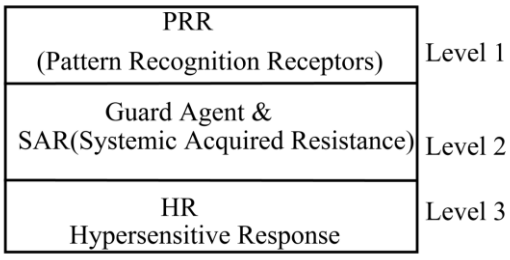

FIGURE 5. Three-layer defense in plants

As shown in Figure 4, the first layer of defense in plants is PRR (Pattern Recognition Receptors). The PRR in plants are responsible of detecting external pathogens or bacteria. After detection, PRR also initiates innate immunity called PTI (Pattern Triggered Immunity) [13]. As discussed in section II, the response of level 1 detection in plants is

production of reactive oxygen species, ethylene production, callose deposition etc. for strengthening cell wall and expression of defense genes. Pathogens that breach PTI deploy huge number of effectors for widespread pathogen virulence. These effectors target certain proteins in plants. The effectors try to form a bond with the protein components resulting in phosphorylation of RPM1 interacting protein 4 (RIN4) [14]. This modification is observed by R protein. The consequence of such action is the activation of SAR (Systemic Acquired Resistance) [15,16], where the primary pathogen information is carried to distal part of the plant. This helps in a quicker response to the secondary infection. In the extreme case to stop plant virulence propagation from the infected part to distal regions, plants adopt an effective response leading to localized cell death. This mechanism in plants is known as HR (Hyper Sensitive Response) [17].

Each of the layer consists different types of agents. The proposed three layer of defense model, PIRIDS, will be active in all the critical nodes. A node is critical if it is stores critical data or host crucial services. For example, the servers or the data warehouse systems used in a network are crucial nodes. The first layer of defense i.e. level 1 consists of different types of receptor agents. Each of the receptor agent is designed with an objective of detecting a specific type of intrusion attempt. Different receptor agents and their role is briefly discussed in the next section. All the incoming packets are sniffed, and the respective extracted feature vector of a packet is passed to every receptor agent. Different receptor agents use this feature vector for detecting different types of attacks. The receptor agents use the pattern matching or rule matching approaches like PRR in plants. Whenever a receptor agent detects an intrusion attempt, it generates a fuzzy membership value corresponding to that connection as discussed in the following sections. This newly generated fuzzy membership value is added to previously stored value. If the resultant value exceeds the fuzzy threshold the source IP address is blocked. This new fuzzy membership value is distributed to other peers in the network. Other peers update this value with the previously stored value. If the value exceeds the threshold value in other peers as well, they take similar measures. Every other critical node maintains a table of the list of fuzzy values of different connections. This becomes the universal fuzzy membership value against the respective IP address in the network. The algorithm of the proposed method PIRIDS is given in Algorithm 1.

Algorithm 1: PIRIDS (Plant-based Inspiration of Response in IDS) Method

Agents ← Receptor agents {SQL injection detector agent, Slowloris detector agent, SYN flood detector agent, TCP/UDP flood detector agent, Honeypot agent, SSH worm detector agent, Monitor agent}
CR ← Critical resource directory
Processes initialization and activation: P ← set of process running in the system.
P_M ← Malicious process breaching receptor agent-based detection i.e. level 1.
P_L ← p Є P i.e. set of all process accessing files Є CR
F ← Fuzzy membership value of P_L
F_L ← files accessed by P_L
Activate all agents;
While agents are up and running do:
    A ← Receptor agents sniff incoming packet
    EV ← Extracted feature vector of A.
    Pass EV to all receptor agents;
    FMV ← 0 //fuzzy membership value
    FMV_{EV} ← generate fuzzy membership value against EV;
    New FMV_{EV} ← FMV_{EV} + FMV;
    If (FMV_{EV} > Threshold), then:
        Take action on the IP address against a given EV;
        Distribute FMV_{EV} to peers;
    Else
        Distribute FMV_{EV} to peers;
        Peers update their respective stored FMV;
    Monitor agent identifies P_M;
    P ← P + $P_{M;}$
    Monitor agent identifies P_L;
    Compute F;
    If (F > Threshold) then:
        Block all the process in P_L;
        Update fuzzy membership value of P_L in its table.
        Distribute F to peers;
    E ← Entropy of F_L calculated
    If (E is high), then:
        Initiate file recovery;
        Request peers for all the critical files;
    Else:
        Monitor the Entropy of all the files accessed;
    If (Number of F_L is High and E is High), then:
        Initiate Hyper Response i.e. terminate temporarily the system in the network to stop probable propagation of malicious process;

In the Algorithm 1, agents are instantiated and activated. The agents are the programs created to meet different objectives. For example, the Slowloris detector agent is responsible for detecting slowloris attack attempt. Once the agents are up, the receptor agents sniff for all the incoming packets. The feature vector of each packet is extracted and stored in variable EV, where EV is the feature vector representation of each packet. The fuzzy membership value for the EV is computed. If this value exceeds the threshold, the connecting IP is blocked, and the fuzzy value will be distributed in the network. For experimental approach the fuzzy membership value threshold is fixed at 0.5. For different intrusions, different fuzzy membership function is adopted and discussed in section IV. If the fuzzy membership value corresponding to a source IP address is equal to 0.5, then it lies in the boundary of being a normal and anomaly address. If the value exceeds 0.5, the source IP address

inclines towards being more anomaly and hence it is blocked. This value of 0.5 as a standard threshold has been adopted in our work. Similar activity takes place with the monitor agent. If fuzzy membership value of a process exceeds the threshold, that process is blocked in the system. If too many intrusions are detected, the end system is temporarily terminated in the network.

The agents are different programs that are created to meet different objectives. As seen in the Algorithm 1, for example SQL injection detector agent is created to identify if any SQL injection on the critical node have been attempted or not. Likewise, the different agents are created for accomplishing its work. The different types of agent are discussed below.

a) ***Honeypot_agent:*** Honeypots are the arsenals to different new attack tools as well as attackers [18,19]. Honeypot agents are programs that create sockets with user given port value. Whenever the network is scanned by any external entity, these ports seem to be open either to the tool or to the attacker. The connection identifier is extracted and written to a file.

b) ***TCP_UDP_agent:*** This agent program extracts the total number of TCP & UDP packets corresponding to a given connection. This agent is primarily concerned of detecting any form of TCP & UDP flood attack [20].

c) ***Slowloris_agent:*** This agent program is responsible of extracting incomplete HTTP header corresponding to a given connection. A complete and normal HTTP header ends with two carriage returns. In this aspect, any HTTP packet non-terminated by two carriage returns is considered incomplete and the corresponding identity is recorded [21].

d) ***Sql_agent:*** This agent program is responsible of extracting any requested HTTP URL, and corresponding test for any SQL injection parameters in the URL. If found, the identity is recorded for further fuzzy calculation.

e) ***Syn_flood_agent:*** This agent program is responsible of identifying SYN flood attack by a remote host. If found any, the record is extracted and stored for further processing [22].

f) ***Brute_force_agent:*** This agent program records all the possible brute force login attempts including 'ssh'. The advantage of agent approach is that it is scalable. Any number of agents can be created for integration with the proposed method. Therefore, the size of the set of agents (like PRR) increases over time.

g) ***SSH_agent:*** This agent program is a ssh bot. If an open ssh port is found, it attempts brute force ssh login into the remote system. Once successful, this agent program uploads a malicious script into the remote system [23].

h) ***Monitor_agent:*** This agent program monitors all the activities that takes place on the critical directory. Whenever a process on the system performs any operation such as read, write or delete on any of the files in the critical directory, the process is recorded for further analysis. If the process turns out to be malicious, it is terminated and thereby, further casualty is restricted.

A. AGENTS AND SPACE COMPLEXITY

It is observed that each critical node in the network runs a number of agents. With higher number of agents running in each node increases the chances of detecting previously known attacks. However, the complexity both in terms of space and time increases with higher number of agents. If x_1, x_2, ... x_n are the different agents running in the critical node and a_1, a_2, a_3 ... are the respective space complexity of each agent then considering the resultant space complexity would be,

$$y = \sum_{i=1}^{n} a_i$$

If y is less than or equal to 50% (ideally 50% left for other services running on the critical node not part of PIRIDS) of total amount of available primary memory then the critical node with all active agents would function well, otherwise, the agents may not have enough primary memory for execution and the execution may fail rendering PIRIDS model to fail in a critical node.

B. ATTACK DETECTION AND AGENTS

All the agents in a PIRIDS model runs in parallel and therefore looking for a pattern of known attack in an incoming connection identifier or an incoming packet is not sequential for different types of attack. The number of attacks detected by Layer 1 of the PIRIDS model is proportional to the number of agents, i.e., if M is the number of different types of agents then the number of different types of attack that can be detected by Level 1 can be given by,

$$A=k * M,$$

where, A is the number of different types of attacks detected and 'k' is a multiplication constant which determines how many types of attack a given agent can detect.

IV. EPIDEMIC MODEL

The epidemic model gives a rough representation of influence of malicious programs on the proportion of functioning systems on the network [31]. As shown in figure 6, A_1 is the first infectious class i.e. if there are 'N' working systems in the network and the attacker attacks 'n' systems (n<N) then, these 'n' systems are placed in A_1. 'A' is the number of susceptible systems in the network. The inclusion of new number of nodes is 'b' and 'd' is the natural crush of the nodes in a given class. If 'θ' is the infectivity contact rate, then the infection will spread from class A_1 as well as new systems infected in A. Therefore, the changes in 'A' with respect to time is given by equation 1. If η is the removal rate of systems due to hyper response in analogy to plants, then R_1 is the set of removed class. If 'k' number of systems are removed due to non-probability of recovery, then n-k remaining infected nodes still can

infect other systems in the network. This set of remaining nodes is the infectious class A_2. If a system is under extensive attack of a malicious program such that it is rendered non-functional, then it is considered a dead system.

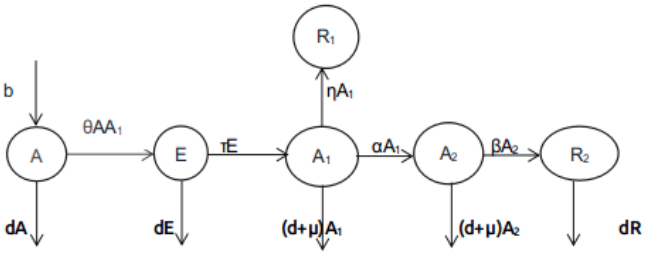

FIGURE 6. Flow of malicious program in network

The variables used are explained below.
A: Susceptible nodes in the network
B: Rate of new inclusion of nodes
E: Exposed nodes in the network
A1: First Infectious class
A2: Second infectious class
R2: Set of recovered nodes
R1: Removed class
D: Rate of natural crush of nodes
α: Infectious rate in class 1 (A1)
β: Recovery rate in infectious class 2 (A2)
θ: Infectivity contact rate
τ: Infectious rate in E
μ: Death rate due to attack
η: Removal rate in infectious class 1 (hyper response)

We can model the malicious attack in the network in a set of equations as follows.

$$\partial A/dt = b - (\theta A A_1 + dA) \quad (1)$$
$$\partial E/dt = \theta A A_1 - (dE + \tau E) \quad (2)$$
$$\partial A_1/dt = \tau E - (d + \eta + \alpha + \mu)A_1 \quad (3)$$
$$\partial A_2/dt = \alpha A_1 - (d + \beta + \mu)A_2 \quad (4)$$
$$\partial R_1/dt = \eta A_1 \quad (5)$$
$$\partial R_2/dt = \beta A_2 - dR \quad (6)$$

The entire network system will be in equilibrium provided the following is met.

$$\partial A/dt = 0$$
$$\partial E/dt = 0$$
$$\partial A_1/dt = 0$$
$$\partial A_2/dt = 0$$

Once the equilibrium point is met it is observed that the changes in the numbers in the class A, E, A_1, A_2 are zero i.e. no further changes in the infection rate of systems in the network. The work we are doing can be modeled through these equations (1) to (6).

V. UNDERSTANDING PIRIDS BY REFERRING TO WYSIWYE ATTACK

The working behaviour of PIRIDS can be better understood by assuming a scenario of attack. The steps of WYSIWYE attack are listed below.

1. The attacker searches the Internet for possible RDP open ports (3389) using Internet search engine like Shodan.
2. Once an open RDP port is discovered, popular tools like NLBrute is used for brute forcing RDP login.
3. On successful login, the entire process of encryption may run in the following steps:
   (a) The attacker generates a public key say K_p and a private key K_s.
   (b) The symmetric key K_{secret} for encrypting the files is embedded with the payload encrypted with the public key K_p.
   (c) Once the payload is downloaded in the victim system, a specific program may contact the C2 (Command and control) server for downloading the private key K_p.
   (d) Once the private key K_p is obtained, it decrypts the symmetric key that is encrypted and run the symmetric encryption algorithm with the decrypted key on the destination files/directories for encryption.
   (e) It might also happen that the payload generates a pair of asymmetric keys for encryption. The public key is then used for encryption and the private key is pushed back to the C2 server.
   (f) Upon command by the attacker, specific files/folders on the victim machine are encrypted.

Any system under this attack may end up with compromised and encrypted data in the hard disk. We try to readdress the situation by assuming that the PIRIDS agents are up and running in the victim machine. Let us consider two scenarios.

A. SCENARIO 1 – INACTIVE RDP

Remote Desktop Services (RDP) is not necessary to be enabled for the organization. In this first scenario, we are assuming that RDP protocol is not necessary to be in an active state in any of the system in the network of the organization. We are assuming all the agents of the PIRIDS model are active and running in the critical nodes of the network. One of the agents is the Honeypot_agent and is falsely hosting port 3389. If an attacker or a bot happens to scan this system in the Internet and finds this port open, it sends a connection request to this port. As mentioned in the three-layer defense of plant, the first level of defense is the PRR (Pattern Recognition Receptors). The Honeypot_agent is the PRR in our method. The Honeypot_agent extracts the metadata like the connection parameters and block this

source IP address. The source IP address is distributed to other critical nodes in the network and the list of `blocked IP' is updated. This ensures that further connection request from the bot is no longer entertained and the attack fails to progress further.

B. SCENARIO 2 – ACTIVE RDP

RDP is necessary to be in active mode for the organization. If the RDP port is necessary to be kept open in the organization, the following cases might arise with PIRIDS active in the critical nodes.

(a) **Multiple login attempt:** Once the attacker/bot discovers open RDP, it will brute force for successful login. The Brute_force_agent of the PIRIDS, which acts like a PRR of plants, extracts the total number of brute force login attempts and generates as stated in the algorithm a fuzzy membership value. If this value surpasses the fuzzy threshold, the metadata of the connection link i.e. source IP address is extracted and blocked. The IP list `blocked list' is updated and distributed to other critical nodes of the network. This behaviour is similar to SAR (Systemic Acquired Resistance) of plants. If there happens to be connection request from the same attacker machine/bot to other critical node of the organization in a later time, the connection is immediately dropped.

(b) **Successful login attempt before fuzzy threshold is met:** The RDP login may be successful in a few attempts and PIRIDS might still detect it as legitimate just because the corresponding fuzzy membership value have not exceeded the threshold. In such a case, the attacker ends up successfully establishing a connection with the critical node. A possible ransomware payload is uploaded to the victim's system. A specific malicious program downloads the cryptographic key for decrypting the symmetric key for encryption. Once all of this is done, let us assume the attacker chooses the folder that is marked as 'critical' by the user for encryption (as WYSIWYE attacks allows the attacker this facility). As previously assumed the PIRIDS agents are already up and running.

One of the agents is Monitor_agent. The role of the Monitor_agent is to monitor all the processes interacting with any files in the critical directory. The process ID of those process are extracted and certain activity record by those process such as number of read operation, write operation, delete operation, cryptographic library invocation etc. are marked. A fuzzy membership value is then computed. If the fuzzy membership value exceeds the threshold, that process interacting with the critical directory is immediately blocked and terminated. If the ransomware process starts accessing the critical directory for encryption, it starts reading the files and starts encrypting them. Thus, the number of read and write operations increases. The Monitor_agent records this value and computes the fuzzy membership value. If significant number of read, write or delete is occurring, the fuzzy threshold value exceeds and the Monitor_agent blocks the ransomware process. Therefore, further infection of files is immediately stopped in the system. The meta data of the process such as the process signature is distributed in the network so that other critical nodes can immediately look for the presence of the process with this signature in their respective system and if it finds one then block the process from future execution.

VI. EXPERIMENTAL RESULTS AND ANALYSIS

The experiment was carried out in the following phases as follows. First, different attack instances were injected into PIRIDS. Second, comparative analysis of some parameters with SNORT as the base reference [24]. In the first instance, different attack instances were generated and injected into PIRIDS. The primary attack that were tested are as follows: (1) TCP Flood (2) UDP Flood (3) Brute Force (4) SQL Injection and (5) Slowloris.

Using the Scapy tool in Linux, incoming packets were sniffed. The respective attribute values were extracted, and the corresponding fuzzy membership value computed. Every connecting source IP address in this manner gets a fuzzy membership value. The fuzzy membership function considered for detecting TCP flood, UDP flood and Brute Force in Linear and given by $\mu_A(x)= (x-a)/(b-a)$, where a is the lower bound and b is the upper bound value for x.

For SQL injection, the membership function used is singleton. The different parameter value considered for different membership function and different attacks against an IP address, are as follows:

(a) ***TCP Flood:*** Lower bound value 3000 - Upper bound value 1000
(b) ***UDP Flood:*** Lower bound value 3000 - Upper bound value 4000
(c) ***Brute Force:*** Lower bound value 50 - Upper bound value 100
(d) ***SQL Injection:*** Lower bound value 1 - Upper bound value 1

Using the fuzzy membership function, once the fuzzy value is computed, it is compared with the fuzzy threshold value. For experimental purposes the fuzzy threshold value is fixed at 0.5. Table I shows the different connection parameters injected into a system with active PIRIDS model, from different IP instances. The experiment is carried out in a LAN and the IP address are private in nature and belongs to class C type address as follows:

(a) ***TCP flood:*** 192.168.63.151
(b) ***UDP flood:*** 192.168.63.162
(c) ***Brute force:***192.168.62.157 (on a VLAN)
(d) ***SQL injection:***192.168.62.156 (on a VLAN)

(e) ***Slowloris:*** 192.168.63.156

Table II shows the fuzzy membership values corresponding to different addresses. For computing these fuzzy membership values, the different fuzzy membership functions as discussed before are used. The significance of each of the table column and row attributes are explained below.

- ***No. of TCP/UDP packets*** - This attribute indicates the total number of TCP or UDP packets from a source IP address.
- ***No. of TCP/UDP Connection*** - This attribute indicates the total number of TCP/UDP connection request from a source IP address.
- ***Incomplete HTTP*** - This attribute indicates the total number of incomplete i.e. open HTTP packets from an IP address.
- ***False Ports*** - This attribute indicates the total number of connection attempt from an IP address into the false service hosted.
- ***SQL attempt*** - This attribute indicates the total number of SQL Injection attempted from a source IP address.
- ***SU attempt*** - This attribute indicates the total number of super users attempted from a source IP address.
- ***TCP Flood*** - This attribute indicates if any attack of the form TCP flood is ongoing or likely from an IP address.
- ***UDP Flood*** - This attribute indicates if any attack of the form UDP flood is ongoing or likely from an IP address.
- ***Brute Force*** - This attribute indicates if any brute force attack is ongoing or likely from an IP address.
- ***SQL Injection*** - This attribute indicates if any attack of the form SQL injection is ongoing or likely from an IP address.
- ***Slowloris*** - This attribute indicates if any attack of the form slowloris is ongoing or likely from an IP address.

The last column of the Table II takes the maximum fuzzy value of the entire tuple. Whichever parameter have the maximum fuzzy value, is the indication of the corresponding type of intrusion attempt. If this value exceeds the threshold value, consequent actions are taken to block the source identifier soon from further hampering the network.

TABLE I. DIFFERENT ATTACK INSTANCES INJECTED INTO PIRIDS

	TCP/UDP packets	TCP/UDP Connections	Incomplete HTTP	False Ports	SQL attempt	SU attempt
TCP flood	2000	2000	20	15	0	15
UDP flood	3000	2000	20	15	0	15
Brute force	500	200	0	0	0	200
SQL Injection	100	10	0	0	5	0
Slowloris	200	30	30	0	0	0

The corresponding fuzzy values against Table I are shown in Table II. The maximum fuzzy membership value of 0.98 and 0.56 from Table II exceeds the minimum value of 0.5. Hence the source identifier corresponding to them is blocked.

TABLE II. FUZZY MEMBERSHIP VALUE CORRESPONDING TO SOURCE ADDRESS

	TCP/ UDP packets	TCP/ UDP Connections	Incomplete HTTP	False Ports	SQL attempt	SU attempt	Fuzzy value
TCP flood	0.1999	0.9763	0.25	0.005	0	0.055	0.98
UDP flood	1	0.9763	0.25	0.005	0	0.055	1
Brute force	0.049	0.004	0	0	0	1	1
SQL Injection	0.009	0	0	0	1	0	1
Slowloris	0.019	0.001	0.55	0	0	0	0.56

TABLE III. DIFFERENT OPERATIONS ON CRITICAL RESOURCES INSIDE CRITICAL DIRECTORY

No of read operation	400
No. of write operation	200
Cryptographic library invoked	1
No. of external connections	10
Process execution time	1200

From our mapping of the IP addresses, 192.168.63.151 and 192.168.63.156 are blocked from future access of the network. The iptables are updated to drop packets from respective source. At times a connection may be successful in breaching the first layer of defense and deploys a malicious process into the end system. During such situations a similar approach like the second layer of defense or indirect approach of pathogen recognition in plants is adopted [11]. The critical nodes in the network maintain a directory which is marked as critical. All critical files are stored in this critical directory. Process interacting with the critical directory are identified and assigned a fuzzy membership degree of anomalous process. The monitoring agent addressed before is responsible for this. The snapshot of a monitoring agent detecting any activity in the critical directory is shown in Figure 7. The process identifier of the process accessing files in the critical directory named test_folder is identified by the monitor agent as 19251. The monitor agent later on assigns a fuzzy membership value to this identifier and if it exceeds fuzzy threshold, that process is blocked. Table III shows different number of operations by a process, say X, in the critical directory. A critical directory is the directory holding all the critical resources. The significance of attributes in Table III are as follows:

- No. of read operation - This attribute indicates the number of read operation by a process in the critical directory.

- No. of write operation - This attribute indicates the number of write operation by a process in the critical directory.
- Cryptography library invoked - This attribute indicates the number of times the inbuilt cryptography library of a system is invoked. Process like ransomware [25] sometimes uses the inbuilt cryptography library to encrypt data. If a process calls such functions, it is likely to be suspected.
- No. of external connections - This attribute indicates the total number of external connections a process is trying to create. Process like Trojan [26] establishes a remote connection i.e. creates a backdoor to a remote bot server. Therefore, it is important to monitor any process trying to create an external connection.
- This attribute indicates how long the process have been executing in seconds.

TABLE IV. FUZZY MEMBERSHIP VALUE OF PROCESS

Process X	
No. of read operation	0.71
No. of write operation	0.0357
No. of delete operation	0.222
No. of cryptography library invoked	1
No. of external connection	0.33
Process execution time	0.51
Fuzzy membership value	0.22

TABLE V. COMPARATIVE ATTACK DETECTION BY SNORT AND PIRIDS

	SQL Injection	Slowloris attack	SSH worm detection	DoS attack
SNORT	Yes	No	No	Yes
PIRIDS	Yes	Yes	Yes	Yes

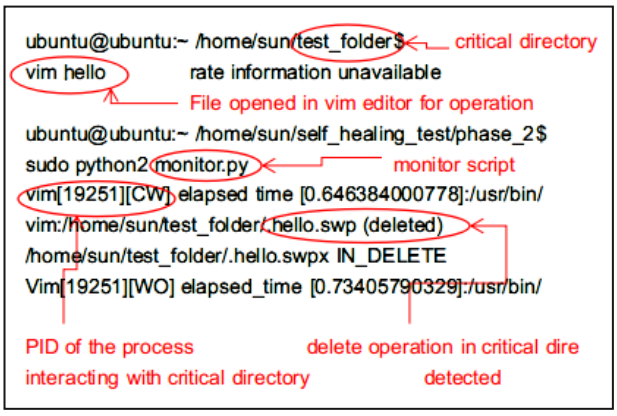

FIGURE 7. Monitor agent monitoring activity in critical directory

Table IV is the fuzzy table corresponding to process operation X shown in Table III. In Table IV the minimum fuzzy membership value is considered. This is because a process which performs all the operations as mentioned in Table III is a suspected malicious process or a probable ransomware. The suspected degree or fuzzy membership value is the intersection of all the operations mentioned in Table III.

The comparison of PIRIDS is done with respect to Snort [24]. Snort is installed on a system in a LAN. Since Snort is an open source IDS, the system installed with Snort behaves as the IDS sensor. One other system in the same LAN is installed with all the PIRIDS agent. Attacks are hosted both into PIRIDS and Snort from other systems in the LAN. In Table V, a comparison of few selected attacks for both PIRIDS and SNORT are shown, where we can see that Snort fails to detect some attacks.

A. TEST CASE WITH SSH BOT

SSH worm in a network scan for available open SSH ports. If it finds one, it establishes a connection using brute force to the remote port and uploads a copy of the malicious script into the remote host for execution. The script executed in the remote host might compromise the host system.

Thus, initially the SHH bot discovers an end system with open SSH port. It than brute force SSH for successful login. Once it gains access through SSH, it uploads the payload script. The payload script is responsible of initiating the malicious activities on the victim machine. The script further scans the network for propagation. This is how the malicious scripts accomplish exponential growth in the network. This experiment was not carried out in any sandbox environment and was run on actual machines with shared memory resources i.e. a network of machines in a LAN were considered as the test bed for the experiment. It was eventually found that the SSH bot could infect successfully other systems in the network thereby expanding the bot network. The SSH Brute force and SSH bot propagation is discussed below along with the response of both Snort and PIRIDS. SSH brute force can be successfully detected in Snort if number of attempts exceeds the defined threshold. The rule set in SNORT for detecting SSH brute force attempt is shown below. As indicative of the rule if a login attempt is made for 15 or more times, SNORT alert a message.

alert tcp EXTERNAL_NET any → HOME_NET 22 (msg:"Possible SSH brute forcing!"; flags: S+; threshold: type both, track by_src, count 15, seconds 30;sid:10000001; rev: 1;).

The significance of the parameters is not explained here and holds the same meaning as in a standard SNORT rule structure. The situation worsens in SNORT, if the brute force is successful before the defined limit. However, before exceeding the threshold limit, the SSH brute force for login parameters such as username and password, SNORT doesn't generate any alarm.

Figure 8 illustrates that after successful brute force of password, the script injects a malicious code into the remote host. The figure indicates that the attacker tries to

upload the SSH bot script *'ssh_bot_upload.py'* in the remote user 192.168.63.151. The malicious script is successfully uploaded in the remote host. This script further continues with its execution of malicious intention. SNORT have no way to monitor this uploading or operation of any malicious script in the host.

```
ubuntu@ubuntu#:~$ python ssh_bot_upload.py 192.168.63.151
ubuntu@123 payload.py

ubuntu@ubuntu#:~$ ssh ubuntu@192.168.63.151
ubuntu@192.168.63.151's password:
Welcome to ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-35-generic
x86_640)
* Documentation: https://help.ubuntu.com/
430 packages can be updated
0 updates are security updates

Last login: Sat Jun 17 18:14:40 2017 from 192.168.63.222
locale: Cannot set LC_CTYPE to default locale: No such file or
directory
locale: Cannot set LC_MESSAGES to default locale: No such file or
directory
locale: Cannot set LC_ALL to default locale: No such file or
directory

ubuntu2@ubuntu#:~$ cd /tmp
ubuntu2@ubuntu#:/tmp$ ls
Mongodb-27017.sock payload.py
```

FIGURE 8. SSH remote host payload inject

Figure 9 shows the encryption in critical directory being carried out by malicious program and the corresponding detection by the monitor agent. It clearly indicates the encryption of files in progress by the malicious program and it also indicates that the activities of malicious program are detected successfully by the monitor agent. The monitor agent immediately extracts the signature of this process and distributes it to other peers in the network. This signature will be a part of monitoring PRR among other peers as mentioned earlier and as such even after successful SSH login from a bot, its attempt to upload and execute the malicious script will fail in the very first instance next time and thereby stopping the propagation of the bot in the network.

```
7915 RO /home/ubuntu/test_folder/automation.jar
7915 O /home/ubuntu/test_folder/(encrypted)automation.jar
7915 R /home/ubuntu/test_folder/automation.jar (deleted)
7915 RO /home/ubuntu/test_folder/httpclient-4.4.1.jar
7915 O /home/ubuntu/test_folder/(encrypted)httpclient-4.4.1.jar
7915 O/home/ubuntu/test_folder/httpclient-4.4.1.jar (deleted)

Done encrypting /home/ubuntu/test_folder/groovy-all-2.4.6.jar
Done encrypting /home/ubuntu/test_folder/commons-logging-1.2.jar
Done encrypting /home/ubuntu/test_folder/automation.jar
```

FIGURE 9. Encryption detected and traced in critical directory

If SSH service is not required in the network, Snort doesn't provide the feature of hosting false services as honeypot whereas PIRIDS provides the feature to host false services. If connection is attempted to a false service, it is a potential bot program scanning open ports in the network. The SSH agent in PIRIDS detects the SSH connection attempt (honeypot service) and the source IP is blocked, restricting the worm from getting installed and further propagation.

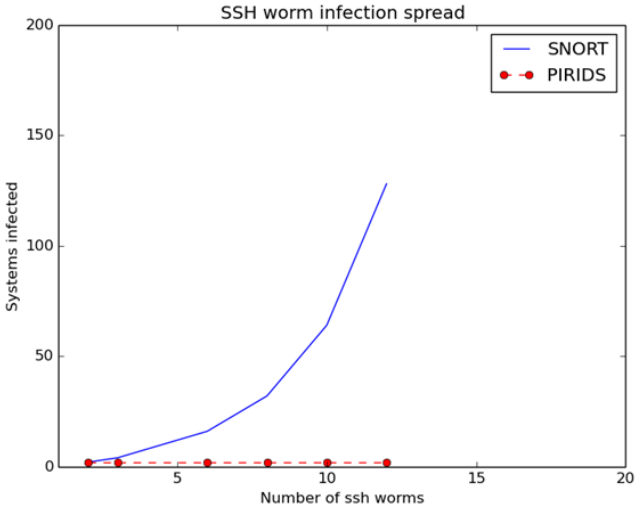

FIGURE 10. SSH worm infection spread

In Figure 10, a comparative plot of worm infection spread rate is shown. In a network with Snort installed as an IDS, the possibility of SSH worm infection spread is significantly high. However, in PIRIDS, the worm is identified by hosting false services in the network (honeypot agent). Whenever an attacker or a bot program sends a connection to the hosted false services the attacker's identity is found and blocked from accessing the network. This blocks further communication by the IP using SSH and therefore the source IP fails to upload the malicious script and therefore, the infection spread is limited. From Figure 10 it can be seen that in a network implementing Snort, there is high risk of a malware bot propagation. With open port scanning and bot installation, the attack may grow exponentially with the available number of systems. However, in a network with PIRIDS implementation, the number of infections is almost near to none. This is because, the moment a peer identifies the malicious process, it extracts the signature and distributes among its peers immediately.

B. TEST CASE WITH SLOWLORIS ATTACK

Slowloris attack have been recently popular and it is an application layer DoS attack. We have hosted this attack in the network and studied the response behaviour of both Snort and PIRIDS. The rule to detect open HTTP connection in Snort can be framed to only fire alerts if certain number of threshold say T_n connection attempts or more occur from the same source within a certain time window T_w. However, if connections are kept open or made every period T_w without meeting the threshold limit

but do an overall flood of open HTTP connection, the Snort detection will fail.

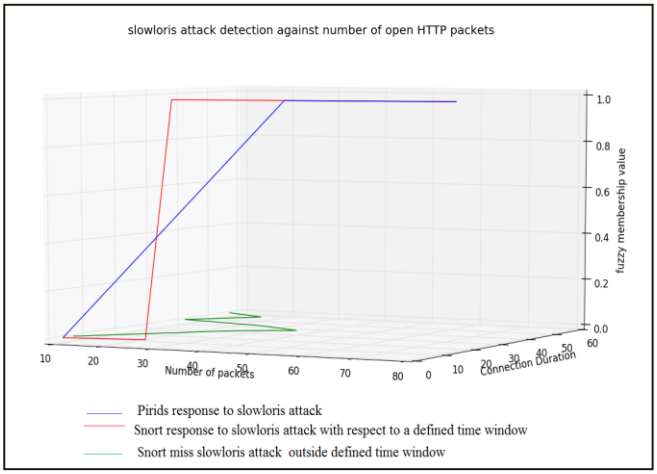

FIGURE 11. Slowloris detection

Snort rule for slowloris detection in a time window of 30 sec may look as follows:

alert tcp EXTERNAL_NET any → HOME_NET HTTP_PORTS (msg:"SlowLoris.py DoS attempt";flow:established,to_server,no_stream; content:"X-a:"; detection_filter:track by_dst, count 50, seconds 30;classtype:denial-of-service; sid:1; rev:1;)

Some of the fields are explained as follows:

- ***no_stream:*** This added to flow option tells the Stream5 preprocessor not to bother checking how the content relates in the context of the reassembled stream.
- **X-a:** Each connection is kept alive with an incomplete HTTP GET request by sending an additional X-a header field every so often.
- **count:** The total number of packets received in the time window for the rule to be active.
- **seconds:** If count number of packets are received from the same source within the ``seconds'' window, the snort rule is activated.

From the graph shown in Figure 11 it can be interpreted that PIRIDS (blue line) response to Slowloris attack is stable. With increasing number of open HTTP packets injected into the critical node, the degree of detection increases eventually. This is the reason the attack cannot be immediately detected. Once the fuzzy threshold is exceeded, the source identifier is immediately blocked, and the probable attack is terminated before it overtakes the system or lead to any form of Denial of Service attack (DoS). Irrespective of any time window frame, the fuzzy membership value for attack increases with increasing open number of HTTP connections in PIRIDS. However, this behaviour is different in Snort. It is observed that if the number of open HTTP connections exceeds the threshold value defined in the Snort rule and met within the time window, Snort successfully detects the attack (red mark indicated). Snort fails to detect the Slowloris attack in circumstances where the number of open HTTP connections are below the threshold in the time window frame but exceeds the threshold limit outside the time window frame (green mark indicated).

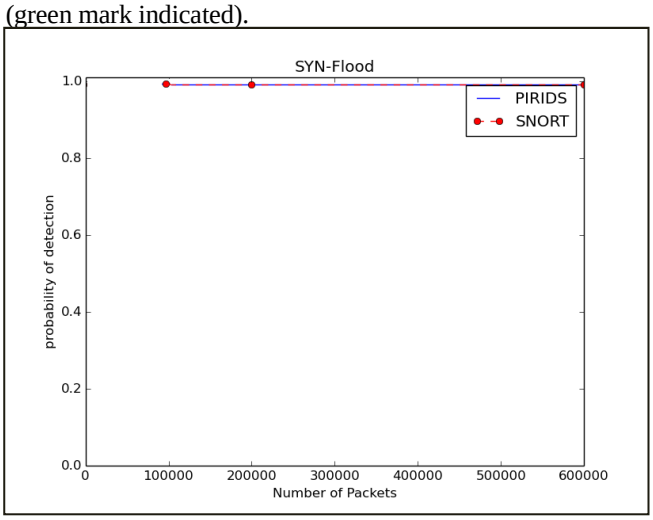

FIGURE 12. TCP SYN flood attack

C. DOS ATTACK USING TCP SYN FLOOD

Both the systems, Snort and PIRIDS were tested against DoS using TCP SYN flood attack. Both the systems could detect and raise alarm. Figure 12 gives a comparative view of both Snort and PIRIDS. It is observed from Figure 10 that the probability of detection is 1 by both Snort and PIRIDS, whenever, the SYN packets exceeds the maximum defined limit. From Figure 12, the probability of SYN flood detection is 1 when the number of SYN packets exceed or equal to the value 100000.

D. TEST CASE WITH SIMPLE RANSOMWARE

The most generic ransomware running in the Internet basically accomplish two operations. First it encrypts the file it gains access to and later deletes the original unencrypted files, leaving behind the encrypted files only in the system [27]. This basic principle is considered and the proposed method PIRIDS is endowed with the capability to detect ransomware process running on a system. It is again observed that Snort fails to achieve the same. An open source malicious activity monitoring software called ClamAV [28] was adopted for the comparison. It was observed that ClamAV failed to detect the ransomware running on the system. However, PIRIDS could detect the ransomware after some initial time. The total number of files deleted in comparison to ClamAV was significantly less. The comparative graph can be seen in Figure 13. The number of files deleted is linear with respect to number of files present in the critical directory. The initial successful deletion of files in PIRIDS is due to fuzzy membership value computation of the process. Once the fuzzy

membership value of the process exceeds the threshold, the process is immediately blocked, and data deletion depletes to zero. Table IV shows the fuzzy membership value computed for a ransomware like process.

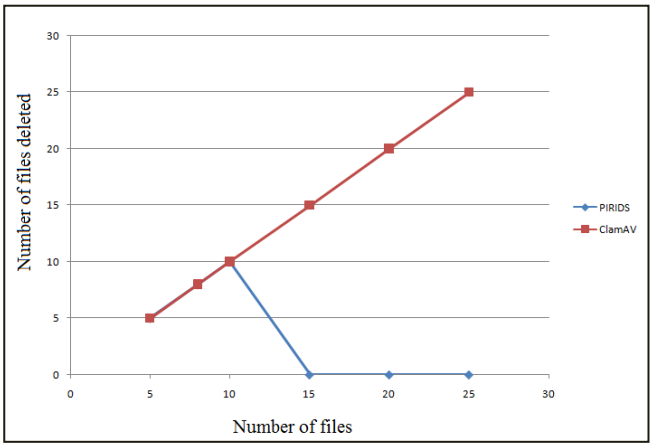

FIGURE 13. File deletion comparison under ransomware like attack

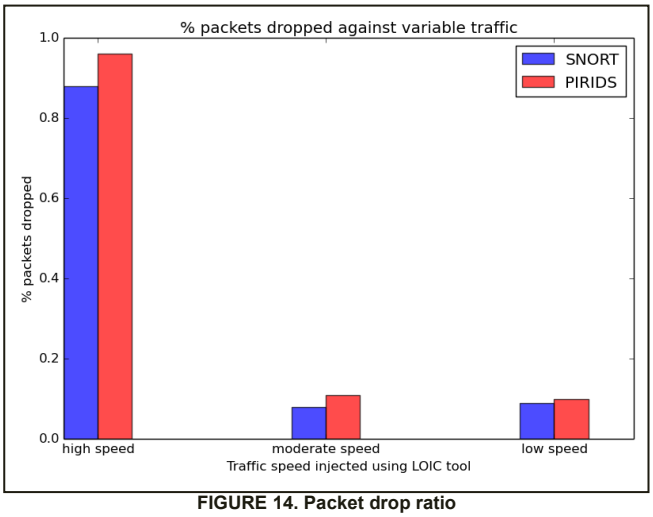

FIGURE 14. Packet drop ratio

E. RESPONSE TO NETWORK TRAFFIC

We wanted to see the response to network traffic and different traffic variations were created using LOIC [29] tool in Windows OS. In high and fast traffic packets dropped were observed in PIRIDS and Snort. Figure 14 shows the packet drop comparison in the different traffic circumstances. It can be observed that packet handling is slightly better in Snort in comparison to PIRIDS, though PIRIDS is not far behind.

F. CPU RESOURCE UTILIZATION

In any IDS system resource including CPU utilization is critical. Figure 15 shows the CPU utilization by SNORT under TCP Flood attack. During this time other rules in Snort were disabled. There is a Linux utility that can monitor program that is using CPU and consuming the most of memory. This utility 'dstat' is able of generate system resources statistics. When a system's CPU cores are all occupied executing the code of current processes, other processes must wait until a CPU core becomes free or the scheduler switches a CPU core to run their code. If too many processes are queued too often, this can represent a bottleneck in the performance of the system. As shown in Figure 15, the encircled parameters show the CPU process utilization by Snort. The format of dstat command is as follows:

dstat -c --top-cpu -dn --top-mem

The significance of some of the parameters are as follows:
--top-cpu (shows most expensive CPU process)
--top-mem (shows process using the most memory)

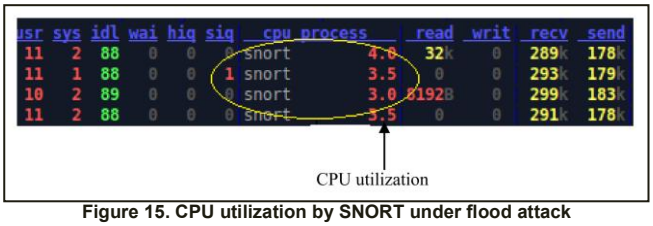

Figure 15. CPU utilization by SNORT under flood attack

Figure 16 shows the CPU utilization by the detector agent of PIRIDS. As shown in Figure 16 the process encircled is the PIRIDS agent and shows the CPU process utilization. The utilization value is significantly smaller than that of shown in case of Snort. Figure 17 displays a comparative result of resource utilization in both Snort and PIRIDS where PIRIDS is much better.

As a final discussion, the Artificial Immune System (AIS) has been widely explored for use in IDS [30]. Even though AIS has found a wide applicability in network security, there are a lot of challenges in its real implementation such as, (1) Identifying the size of detector set and generating a true detector set (2) Identifying the self and the non-self set and (3) Incorporating every new file or a new network connection feature to be a part of self or non-self and subsequently generating new detector set. The approach of AIS suffers autoimmune attack. If file resources stored in a system are considered self-cells, then the system fully trust the self-cells and the intersection of the detector set with the self-cell set is null or void. Therefore, if a file stored in a system and considered trusted starts mal-functioning, it becomes difficult for the AIS to detect such behaviour.

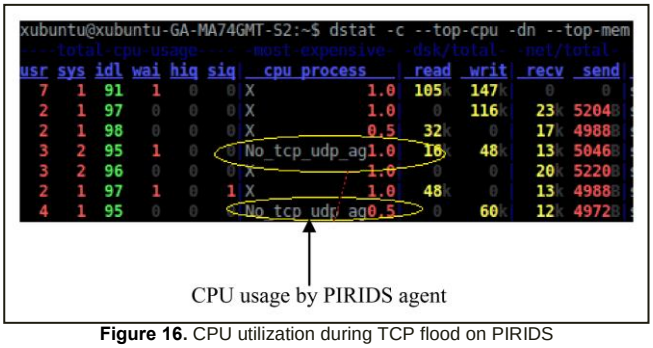

Figure 16. CPU utilization during TCP flood on PIRIDS

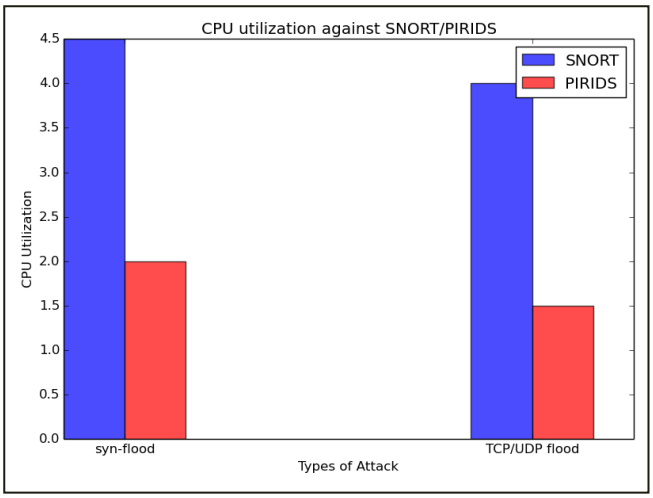

Figure 17. Comparative CPU utilization during TCP flood attack

The proposed system PIRIDS overcomes this difficulty of the AIS. If a malicious program successfully breaches the defense system and infects a system resource, the model has a way of detecting it through the guard agent of the model. The difficulty of generating self and non-self set doesn't hold. The diversity of attack can be detected in the PIRIDS by incorporating new receptor agents. This avoid matching each incoming pattern unlike in AIS with each detector set element.

VII. CONCLUSION

In this work a bio-inspired method of intrusion detection and response model namely Plant-based Inspiration of Response in IDS (PIRIDS) is proposed. The experimental evaluation of the proposed model has very well demonstrated that the model is dynamic enough to incorporate receptor agents to detect known attacks. Any zero-day attack intend to hamper any services or resources of the critical node can be identified. The signature of the potential malicious program could be extracted, and the infection spread in the network could be stopped. The experiments demonstrate that the proposed model outperforms established open source IDS Snort in many ways and specially in responding to ransomware attacks. The proposed model has also performed well with CPU resource utilization. The model with further improvements and incorporation of evolutionary attack design techniques can be made as a standard for detecting attacks and generating responses subsequent to attacks.

REFERENCES

[1] R. K. Sharma, H. K. Kalita, S Das, and B. Issac. Learning is never secure: Poison learning by intrusion detection system based on self-organizing Map. 5th IEEE-EDS International Conference on Computing, Communication and Sensor Network - Kolkata, India, 2016.

[2] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman. Towards the science of security and privacy in machine learning. arXiv preprin arXiv:1611.03814, 2016

[3] C. J. Thorogood, U. Bauer, and S. J. Hiscock. Convergent and divergent evolution in carnivorous pitcher plant traps. New Phytologist, 217(3):1035–1041, 2018

[4] V. Aliko, M. Qirjo, E. Sula, V. Mo-rina, and C. Faggio. Antioxidant defense system,immune response and erythron profile modulation in goldfish, carassius auratus, after acute manganese treatment. Fish & shellfish immunology, 76:101–109, 2018.

[5] S. Sarkis, S. Dabo, M.-C. Lise, C. Neuveut, E. F. Meurs, V. Lacoste, and A. Lavergne. A potential robust antiviral defense state in the common vampire bat: Expression, induction and molecular characterization of the three interferon-stimulated genesoas1, adar1 and pkr. Developmental & Comparative Immunology, 85:95–107, 2018.

[6] J. Mitra and P. K. Paul. A potent biocide formulation inducing sar in plants. Journal of Plant Diseases and Protection, 124(2):163–175, 2017.

[7] D. F. Klessig, H. W. Choi, and D. A. Dempsey. Systemic acquired resistance and salicylic acid: Past, present and future. Molecular Plant-Microbe Inter-actions, (ja), 2018.

[8] R. K. Sharma, H. K. Kalita, and B. Issac. PIRIDS: A model on intrusion response system based on biologically inspired response mechanism in plants. In Innovations in Bio-Inspired Computing and Applications pages 105–116. Springer, 2016

[9] J. D. G. Jones and J. L. Dangl. The plant immune system. Nature, 444(7117):323–329, 2006

[10] N. Suzuki, S. Koussevitzky, R. O. N Mittler, and G. A. D Miller. Ros and redox signaling in the response of plants to abiotic stress. Plant, Cell & Environment, 35(2):259–270, 2012.

[11] A. M. E. Jones, J. Monaghan, and V. Ntoukakis. Mechanisms regulating immunity in plants. 2013

[12] S. H. Spoel, and X. Dong. How do plants achieve immunity? Defence without specialized immune cells. Nature Reviews Immunology, 12(2):89-100, 2012

[13] S. Mazzotta, and B. Kemmerling. "Pattern recognition in plant innate immunity." Journal of Plant Pathology (2011): 7-17.

[14] Y. Belkhadir, R. Subramaniam, and J. L. Dangl. Plant disease resistance protein signaling: Nbs-lrr proteins and their partners, Current opinion in plant biology, 7(4):391–399, 2004.

[15] J. A Ryals, Urs H Neuenschwander, M. G. Willits, A. Molina, Henry-York Steiner, and Michelle D Hunt. Systemic acquired resistance. The plant cell, 8(10):1809, 1996.

[16] W. E. Durrant and X. Dong. Systemic acquired resistance. Annu. Rev. Phytopathol. , 42:185–209, 2004

[17] L. A. J. Mur, P. Kenton, A. J. Lloyd, H. Ougham, and E. Prats. The hypersensitive response; the centenary is upon us but how much do we know? Journal of experimental Botany, 59(3):501–520, 2007.

[18] L. Spitzner. Honeypots: tracking hackers, volume 1. Addison-Wesley Reading, 2003.

[19] N. Provos. Honeyd-a virtual honeypot daemon. In 10th DFN-CERT Workshop, Hamburg, Germany, volume 2, page 4, 2003.

[20] C. L. Schuba, I. V. Krsul, M. G. Kuhn, E. H. Spafford, A. Sundaram, and D. Zamboni. Analysis of a denial of service attack on tcp. In Security and Privacy, 1997. Proceedings., 1997 IEEE Symposium on, pages 208–223. IEEE, 1997

[21] E. Cambiaso, G. Papaleo, and M. Aiello. Taxonomy of slow dos attacks to web applications. Recent Trends in Computer Networks and Distributed Systems Security, pages 195–204, 2012.

[22] M. H. Wang, D. Zhang, and K. G Shin. Detecting syn flooding attacks. In INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, volume 3, pages 1530–1539. IEEE, 2002.

[23] S. E. Schechter, J. Jung, W. Stockwell, and C. D. McLain. Inoculating SSH against address harvesting. In NDSS, 2006.

[24] M. Roesch et al. Snort: Lightweight intrusion detection for networks. In Lisa, volume 99, pages 229–238, 1999.

[25] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge, and E. Kirda. Cutting the gordian knot: A look under the hood of ransomware attacks. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, pages 3–24. Springer, 2015

[26] N. Gisin, S. Fasel, B. Kraus, H. Zbinden, and G. Ribordy. Trojan-horse attacks on quantum-key-distribution systems. Physical Review A, 73(2):022320, 2006

[27] A. Gazet. Comparative analysis of various ran-somware virii. Journal in computer virology, 6(1):77–90,2010.
[28] K. Tomasz. Clam AntiVirus 0.88.3, User Manual, 2006.
[29] A. Pras, A. Sperotto, G. Moura, I. Drago, R. Barbosa, R. Sadre, R. Schmidt, and R. Hofstede. Attacks by anonymous WikiLeaks proponents not anonymous. Technical report, University of Twente, Centre for Telematics and Information Technology (CTIT), 2010
[30] A. Cooper. Experiments with applying artificial immune system in network attack detection. 2017
[31] M. B. Kumar, and A. Prajapati. “Mathematical model on attack by malicious objects leading to cyber war”, International Journal of Nonlinear Science 17.2 (2014): 145-153.
[32] D. Franck. "Introduction to fuzzy logic." Massachusetts Institute of Technology 21 (2013).
[33] H. Tzung-Pei, and C.-Y. Lee. "Induction of fuzzy rules and membership functions from training examples." Fuzzy sets and Systems 84.1 (1996): 33-47.
[34] H. T. Nguyen, et al. A first course in fuzzy and neural control. Chapman and Hall/CRC, 2002.
[35] B. Dmitry. "Lecture Notice. Introduction to Soft Computing", Accessed 15 January 2019.