International Journal of Engineering & Technology, 7 (4.25) (2018) 298-303

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Automation of Dynamic Multi-Layer Signature based Intrusion Detection System with Pattern Similarity and Recognition

T.S.Urmila[1], Dr.R.Balasubramanian[2]

[1]Research Scholar, Mother Teresa Women's Univeristy, Kodaikannal
[2]Dean, Karpaga Vinayagar College of Engineering and Technology, Mathuranthagam

Abstract

Every computer on the Internet is a potential target for a new attack at any moment nowadays. Network intrusion Detection System (NIDS) is one of the fundamental components to monitor and analyze the traffic to find out any possible attacks in the network. Intrusion Detection based on the application requires exploration of network packet payload data. A special model is required for each services while, every service has different behavior. This paper aims on network packet payload data and it will improve suspicion and recognize signature based attack patterns using pattern matching strategy distantly more accurate than approaches that consider only header information. This paper focuses on developing a multi-layered design known to be a *Dynamic Multi-layered Signature Pattern Similarity and Recognition (DMSP-SR)*. The concept of DMSP-SR is introduced for payload data intrusion that would verify the packets, cluster the packets, measuring pattern similarity and recognize the intrusion signature pattern to diminish these attacks. This is a Multi-layered framework design would enhance the overall performance of the signature based intrusion detection system with the set of attack patterns. The performance analysis shows that the proposed framework can improve the accuracy by increasing the detection rate and effectiveness by reducing the false positive rate and increase true positive rate of identifying payload intrusion compared to the existing systems.

Keywords: *Intrusion detection, Signature based IDS, Payload Data, Multi-layer framework, Pattern Similarity.*

1. Introduction

In modern era of information security systems all key network intrusion detection systems are still using signature based approaches for attack detection. Snort [1] and Bro [2] are admired examples of signature based intrusion detection systems. Such systems use attack recognition mechanisms based on signatures of previously known attacks or vulnerabilities. This method works well if the signature database is up to date. Though, it has been observed that signature based detection method fails for zero day attacks or transformation of known attacks due to lack of signature accessibility when a zero day attack is instigated. Another choice for this difficulty is anomaly detection systems [3]. In anomaly detection system variation from normal behavior is identified to signal probable novel attack. While signature-based network intrusion detection systems have turn into an essential component of network security and established its ability to detect various network attacks, the problems of network packet load, exclusive signature matching and enormous false alarms (positives) can still greatly obstruct the performance of these detection systems in a large-scale network environment. For example, the expensive progression of signature matching can source a signature-based NIDS (e.g., Snort) to drop a large number of network packets during its detection since the number of packets in a large-scale network often significantly exceeds the maximum processing potential of such a detection system. The payload is that the actual contents of the packets, whereas the header contains further data as well as the packet's supply and destination address. As information flows across the network, the packet analyzer captures every packet and analyzes its content for applicable intrusion detection. In addition, this also observed that services tend to have the ability of port locality, which means services get used to assign ports to communicate in a consecutive manner. During communication, if a service session needs to build a new connection, the service will assign continuous ports for each new connection, even for the services using randomized destination port numbers first. Once one certain port is used for the first connection, the remaining connections would very likely use continuous port numbers from the first port assigned.

Systems are limited when a worm has been identified, and a signature has been developed and distributed to signature-based detectors, such as a virus scanner or a firewall rule. Several well-known illustrations of attacks are portraying that they propagate at very high speeds on the internet. These are simple to note by analyzing the speed of scanning and searching from external sources which might indicate that attack propagation is underway. Unfortunately, this approach detects the first onset of propagation; however the worm has already with success penetrated a number of victims, infected it and commenced its harm and its propagation. (It ought to be evident that slow and surreptitious attack propagations could go disregarded if one depends entirely on the detection of rapid or busty changes in flows or probes.) This work aims to observe the primary occurrences of attack either at a network system gateway or among an internal network from a rogue device and to prevent its propagation. Though it forged the payload anomaly detection problem in terms of attacks, the strategy is beneficial for a good range of exploit attempts against many if not all services and ports.

2. Related Review

There are two types of systems that are called anomaly detectors: those based upon a specification (or a set of rules) of what is regarded as "good/normal" behavior, and others that learn the behavior of a system under normal operation.
Further systems are simply packet based like PHAD [7], NATED and NATE [8]. They identify anomalies in network packets really not including renovation. This approach has the significant benefit of being fast and effortless, and they are moderately superior at detecting those attacks that do not achieve in appropriate associations or sessions, for example, scanning and probing attacks. In the paper [9] used protocol information, headers, and packet payloads of packet captures to reduce the total number of false alarms from DDoS attacks. The rules defining the source address, destination address and destination port were used to increase the speed of signature detection in another study [10].
In [11], the signature of a priori algorithm from the MySQL database logs was proposed to detect known network level attacks. Based on PHP source code, paper [12] used a controlled VM to detect XSS attacks achieving 0% FP and 0% FN rate. In [13], examined Apache logs to collect data on string length and sequence and exploited mutations to detect buffer overflows, directory traversals and other attacks. The paper [14] used web servers that were hosted in a Cloud environment to detect SQL injections, XSS, and brute-force attacks. Finally, in [15] the researchers looked at innerHTML properties such as GET, HTTP header and cookies to determine the presence of mutation-based XSS attacks, denoted as mXSS. This work, in contrast, uses logs collected from both the web server and the MySQL database for analysis.

3. Components in dynamic multi-layering

This section describes two significant inspections for identifying the intrusions: (a) *Packet Capture Component* and (b) *Pattern Restoration Table (PRT) Component*. First observation involves in the inspection of the incoming packet over the network. The signature patterns are included in this module for the intrusion detection.

3.1 Packet capture component

Network Administrator evaluates and controls overall network traffic and presentation. To come across at and confine period of time running packets over a network, completely different packet capturing techniques are used. The packet capturing is detained filtering, during which filters are applied over network nodes or devices wherever information is captured. The complete packet includes two things: a payload and a header. The payload is that the actual contents of the packets, whereas the header contains further data as well as the packet's supply and destination address. As information flows across the network, the packet analyzer captures every packet and analyzes its content for applicable intrusion detection.

3.2 Pattern restoration table (PRT) component

To develop the representative centers of pre-selected payload from a set of traffic data, it need a representation of each link. In order, to assist and speed up identification, this system propose an auxiliary structure called *Pattern Restoration Table (PRT)*, which is a matrix used to store the information of attack patterns. The attack signature pattern is stored in this structure with 4-tuple,
(Srv, P_{len} P_{Start}, P_{End})
is the signature pattern which denotes Srv is the service, packet length is indicated as P_{len}, starting pattern string is indicated as P_{Start} and the ending pattern string is represented as P_{End}. This information will be recorded in PRT with the form *(Srv, $P_{len,}$ P_{Start}, P_{End})*. For any other connection, if one of its hosts is already in PRT and the pattern string in the pair *(Srv, P_{Start})* or *(Srv, P_{End})* is continuous compared with one entry of the PRT, this connection is inferred to belong to the same session.
Assume that there are multiple packets collected in a connection $P_{traffic}$, and the packets of the same length are put together and that are clustered into a group. Then the set of different packet sizes are sorted in sequence,
$P_{SEQ} = (P_{len1}, P_{len2}, P_{len3,} \ldots P_{leni})$
which denotes the grouping of the i^{th} larger packet length in $P_{Payload}$. The packets are verified using the 4-tuple based structure of PRT. As an example, the following table illustrates the PRT representation for a service type.

Table 1: PRT Representation

Srv	P_{len}	P_{Start}	P_{End}
TCP	6	Update	1=1
HTTP	8	GET//	///////

4. Proposed methodology

The proposed framework for signature based intrusion in payload data of network traffic is based on multi-layered approach that would identify the network payload packets as normal or abnormal. This framework is divided into three phases. The first phase of this framework involved in classifying the incoming packet payload data by performing transformation process and clustering the packet content. In the second phase, the dynamic similarity of pattern analyzing is performed. Finally, the third phase of this framework recognizing the attack signature pattern to provide alarm. The architectural design of DMSP-SR packet payload data is depicted in figure 1.

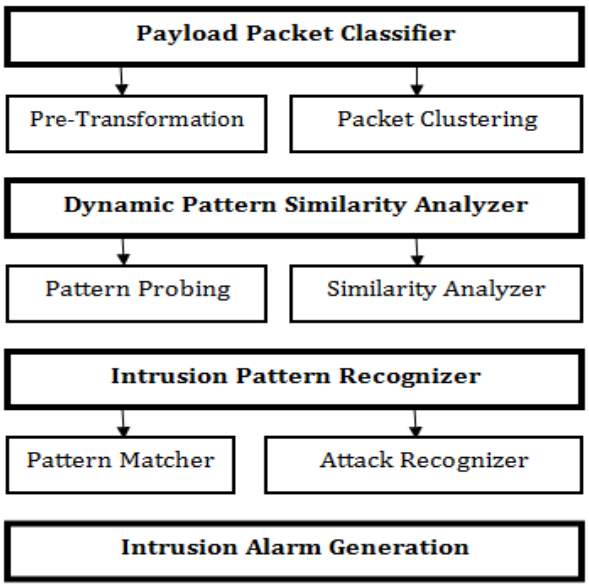

Figure 1: Architectural Design DMSP-SR for Packet Payload Data

In capturing of packets, system captures incoming payloads payload patterns and compares the payload with stored attack patterns. If the new payload pattern matches with any stored pattern for same service, then an alert is generated indicating suspicious packet. The system can be deployed at network entry point level or at an end host. This proposed framework of DMSP-SR is consists of the following layer:

- Payload Packet Classifier
- Dynamic Pattern Similarity Analyzer
- Intrusion Pattern Recognizer

4.1 Payload packet classifier

This layer of the proposed design involves in the development of packet payload data with the profile conception of the normal packets to classify the incoming network traffic payload with association among the payload length and the service of the payload data.

4.1.1 Pre-Transformation

Pre-transformation converts network traffic into a series of observations, wherever every observation is delineated as a feature vector. The method of pre-transformation is to eliminate the unprintable characters, redundant information and null information values from the traffic information packet payload data. The network traffic classification phase first parses all connections in real-world traffic and extracts the 5-tuple,

(source$_{IP}$, destination$_{IP}$, service, P_{len}, $P_{Payload}$)

with the packet length distribution of the connection. In the pre-transformation, the payload data that handles with the processed data with reduced unprintable characters which would have unformatted payload content data. These unformatted content data may have unnecessary spaces which would be truncated to avoid the complicated content from the payload data.

4.1.2 Packet clustering with similarity features

The extracted incoming packets utilizing a highly precise network measurement device, this system investigate router's inherent variation on packet processing time and its effect on inter-packet delay and packet clustering. The packet classification is done with the extraction of incoming packet feature set and weigh among the packet. The weigh measure of the incoming packets to cluster the packets for the similarity pattern probing and to speed up the identification and recognition process the packets are clustered with the extraction of following 3-tuple,

(service, P_{len}, $P_{Payload}$)

that form the clusters. The clusters are formed or re-formed based on the service and the length of the incoming packets. The clusters are formed with the same service type of incoming packets and weigh measure of the payload length within the same type of the service. This would create number of clusters that would speed up the process of dynamic pattern similarity probing.

The algorithm of this layer described below:

Algorithm Packet Classifier (Int No_P) // No_P is the number of packets

{

String $P_{Srv,}$ $P_{Payload}$; // Service and Packet Payload

Int Port, P_{len}; // Port number and Pattern length

Double S_{IP}, D_{IP}; // Source and Destination IP

Array PRT=GetArray[Srv, P_{len}, P_{Start}, P_{End}]; // Pattern Restoration Table

For each i from No_p

{

Read(P_{Srv});

Read(Port); Read(P_{len});

Read(S_{IP}, D_{IP});

Read $P_{Payload}$(i) from $P_{payload}$; //

For each C from $P_{Payload}$(i)

// performing pre-transformation for the extracted payload content

{

GetASCII(C);

If (Exists(C)) then // non-printable characters like $, #, etc

{

Remove(C) from the $P_{Payload}$(i);

}

} // The Pre-transformed payload is extracted

Srv=Probe_Srv($P_{Cluster}$)

If(P_{Srv}==Srv)

{

For each S from $P_{Payload}$(i)

// extracting string from the pre-transformed payload content

For each len from P_{len} // computing length of the pattern and cluster the patterns

{

M = μ(len) // Compute Mean value M for each len of signature pattern

//Calculate Distance among M and len

$$K = \frac{1}{M_j} \sum_{n}^{j=1} M_j \square len$$

If (M_j <= len && P_{Srv}($P_{Payload}$(i)) then

{

$P_{Clustter}$(Srv, P_{len}, $P_{Payload}$(i)); // clustering the pattern without same service type and nearer length

}

Else

{

$P_{Recluster}$(Srv, P_{len}, $P_{Payload}$(i)); // Re-clustering the pattern with same type and length

}

}

}

}

Then the incoming packets features are associated with the formed clusters data items with the incoming packet payload. If the incoming packets are with the same service type and length are clustered together to reduce the workload of the identification process. To calculate the neighboring length distance, the following parameter is considered:

$$K = \frac{1}{M_j} \sum_{n}^{j=1} M_j \square P_{lenj}$$

After the clustering, the pattern similarity analysis is performed in the next layer of the framework. This analysis is carried out with this clustered signature pattern.

4.2 Dynamic pattern similarity analyzer

In this layer, the pattern similarity for the payload content is analyzed, unlike the network packet headers; payload doesn't have a fixed format. It has small set of keywords or expected tokens, or a limited range of values. The service and the length are two obvious choices for modeling. In one service type, the payload length differs above a considerable range. The main common TCP packets have payload lengths from 0 to1460. Different length ranges have differing kinds of payload. Thus to modeling these payload patterns this framework uses *Dynamic Pattern Similarity Analyzer (DPSA)* which corresponds to all the possible pattern with various payload length and service type. The DPSA can be divided into two sub-layers as follows:

4.2.1 Pattern probing

In order to search out the suspicious packets, most of IDSs use a pattern matching rule. The rule checks the presence of a signature within the incoming packet patterns and outputs the location of the string among the packet. In this sub-layer, the pattern probing is carrying out with the clustered packet set. The *pattern probing* is the approach that searches for the attack pattern with the clustered packets. After grouping the payload, it may contain an attack signature patterns which differs from the every service type. In this layer of the framework, the clustered incoming packets are extracted and the patterns are probed in the format of PRT structure. Then the clustered packet payload pattern is verified by searching for the pattern string occurrence in the PRT.

4.2.2 Similarity analyzer

In order to build the model of intelligent behavior of intrusion detection based on dynamic signature pattern analysis, a dataset containing only legitimate signature pattern is accomplished. Initially, the packets are clustered, based on the following two criteria: (1) the observed service and length; (2) the number of chunks a payload pattern contains. Subsequently, for each different cluster, a model is built that will be then exploited to assess the nature of never seen packets.

The first criterion is needed in order to identify signature pattern of different nature. Each service implies specific signature patterns, so it is important to separate traffic pertaining different services to capture information about specific characteristics of the pattern from the clusters.

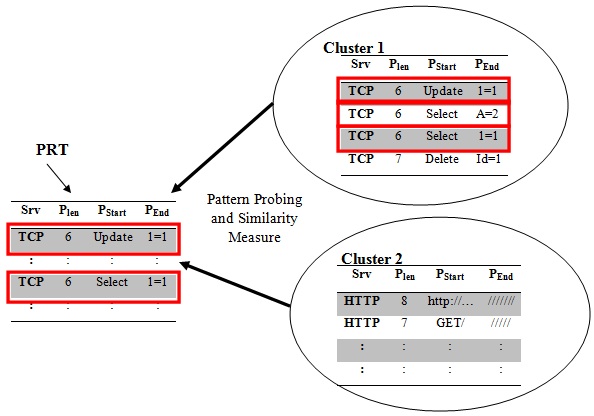

Figure 2: Pattern Probing and Similarity Measure

The similarity measure among the cluster and PRT searching and recognize the signature patterns. To measure the similarity of cluster $P_{Cluster}=\{P_{Cluster(i)}\ldots P_{Cluster(n)}\}$ and PRT structure is then defined as,

$$Sim(P_{Cluster}, PRT) = \frac{\sum_{j<=m}^{i<=n} S_{ij}}{Max(n,m)}$$

The second criterion is used to build a model from patterns with similar signature; therefore the chunks of the signature is extracted to measure the similarity of the pattern occurred. The signature pattern is extracted with the pattern string or pattern chunk of $P_{Start,}$ and P_{End} with Srv and P_{len} of the clustered packets. Then the P_{Start} pattern string is probe for the starting and also the P_{End} string is searched with ending pattern from PRT of the signature in clusters with same service type and length. If the pattern is occurred in the clustered packet payload, then the packet payload is suspected as signature pattern is expressed as intrusion signature. If not, the signature is conceded out to the next layer of the framework to recognize the attack.

4.3 Intrusion pattern recognizer

In order to recognize the suspected incoming packets from the similarity analyzer, this layer follow the pattern matching rule. The time period of inspecting network packets needs the utilization of a pattern search which will sustain with the speeds of networks. In the layer of DPSA, the method of pattern and similarity measure that identifies the similar patterns from the clustered packets compared with PRT. But the limitation is that the signature pattern suspected may not be an attack pattern, so the framework has to recognize the attack signature with two-phase matching. Thus in this layer, the pattern matching rule is employed with the use of *Regular Expression (RegEx)*. This layer contains two different inner-layers: Patten matcher and Attack Recognizer.

4.3.1 Pattern matcher

A novel signature rule pattern or modify an existing pattern of a signature rule to identify a string or expression that illustrates a facet of the attack that the signature matches. To work out that patterns an attack exhibits, this could examine the signature patterns, or acquire the string or expression from a pattern report regarding the attack. However, the performance needs of traditional strategies for matching the incoming events against regular expressions are prohibitively high. This limits the utilization of regular expressions in majority of contemporary IDS. During this work, present an approach for selective matching of regular expressions. Rather than serially matching all regular expressions, this framework compile a group of shortest patterns most often seen in regular expressions that permits to quickly filtering out events that don't match any of the IDS signatures.

4.3.2 Attack recognizer

RegEx are used to depict security threats' signatures in network intrusion detection (NID) systems. To recognize suspicious packets using regular expression similarity, which is rapid and permits dynamic updates is employed in this layer. However, a number of practical signature patterns generally establish in a variety of NID systems. The RegEx consists of constants and operators denoting the sets of strings and the operations over these sets, correspondingly. Given a finite alphabet set Σ and subsets R, S of Σ. Characters in an alphabet Σ play two roles: one is literals, and the other meta-characters, i.e., characters representing extra operations to combine RegEx in various ways. In the former type, this system could find all sub-strings starting with P_{Start} when the input is scanned from start to end. In order to achieve this goal, this system should prepend a regular expression ".*" to each pattern without "^".

Algorithm Similarity Pattern Analyzer ($P_{Payload}$, $P_{Cluster}$, Srv)

{
String $P_{Signature}$;
Read ($P'_{Payload}$)
For each i from n
If(Srv==Srv($P_{Cluster}$))
{
Search (P_{Start}) in $P_{Cluster}$(i) // searching for starting string pattern from PRT to Cluster
Search (P_{End}) in $P_{Cluster}$(i) // searching for Ending string pattern from PRT to Cluster
If (P_{Start}== Start ($P_{Cluster}$(i)) && P_{End} == End ($P_{Cluster}$(i)))
{
Similarity_Measure(PRT, $P_{Cluster}$(i)) // Similarity Measures among the cluster and PRT
Suspect_Packet($P_{Payload}$(i)) // the packets are suspected
RegEx ($P_{Payload}$(i)) //Regular Expression pattern sets
Find Similar Suffix from the $P_{Payload}$(i)
For each j from 1 to k-1 // k number of suffix patterns in payload
{
Split_Prefix_String($P_{Payload}$(i))
If(Str_Len($P_{Payload}$(i))>8)
{
Compare (Suffix($P_{Payload}$(i), Prefix($P_{Payload}$(i))
}
}
}
Alarm_Generate (P_{IDS})
}
}

}

This layer employs the recognizing process of P_{Start} to find all the possible matches. In other words, if it finds all matches and reach the end of the input from start position P_{Start}, this layer need to

start the next recognizing process from the next character position in input P_{End}. The approach involves signature transformation that results into RegEx. The results indicate that the signatures are of intrusion pattern/normal pattern would generate intrusion alarm with IP labeling as untrusted IP and exhibit low false negatives and false positives.

5. Performance evaluation

The performance evaluation of this payload intrusion detection is illustrated in this section. The proposed payload detection engine provides less time complexity and high detection rate. The performance metrics such as time complexity for packet string length, time complexity for string length range and pattern string detection rate of the packet payload content are evaluated for proposed technique DMSP-SR and existing systems GA(Genetic Algorithm based)-IDS [16], MAPCG[17].

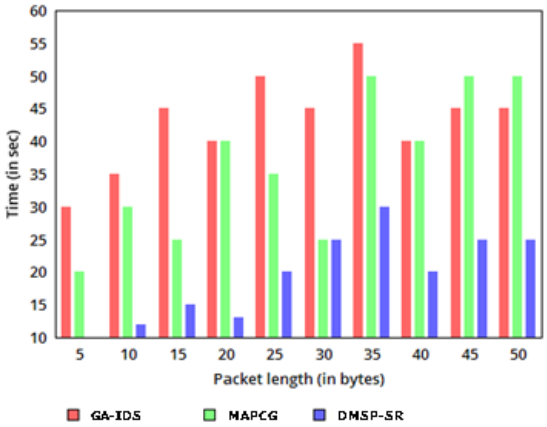

Figure 3: Time Complexity for Various Signature based Intrusion Detection System

The time complexity for the proposed payload intrusion detection system DMSP-SR is shown in figure 3. The figure illustrates that the time taken to recognize the payload content concerning to the packet string length and it depicts that the proposed DMSP-SR consumes a reduced amount of time to recognize the packet payload than the existing systems GA-IDS and MAPCG.

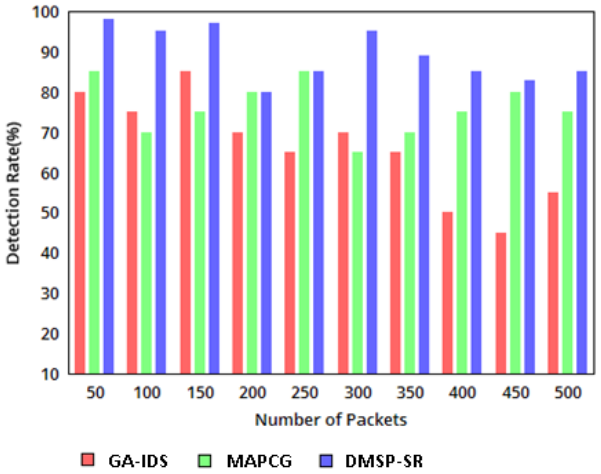

Figure 4: Pattern Detection Rate for Various Signature based Intrusion Detection System

Figure 4 illustrates the detection rate for the number of packets in the network traffic. This depicts that the detection rate for DMSP-SR provides high rate for the number of incoming packets in the network than the existing methodologies.

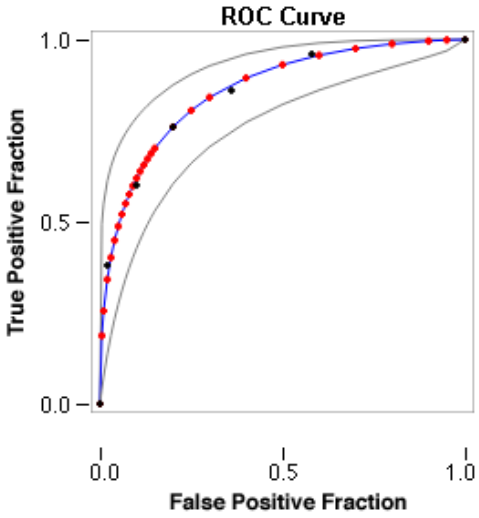

Figure 5: FPF and TPF for Various Existing Intrusion Detection System

The false positive rate of the packet traffic is evaluated in figure 5. This figure defines that the time intervals acquires less false positive rate for the DMSP-SR than GA-IDS and MAPCG.

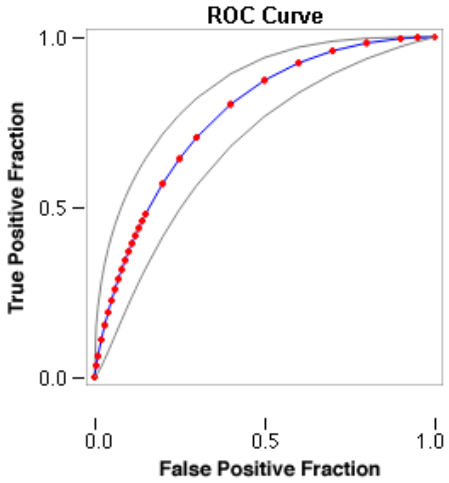

Figure 6: FPF and TPF for DMSP-SR IDS

Figure 6 illustrates that the false negative rate that involves in detecting the intrusion signature pattern and recognize the intrusion from the network packets. This shows that the proposed DMSP-SR IDS provides better results with less false negative rate.

6. Conclusion

In this paper, the signature based intrusion detection framework termed as DMSP-SR is established to detect and recognize the intrusion occurred in the payload data. The previous work of the payload data intrusion detection based on the signature patterns are employed using a signature pattern matching algorithms. This system employs multi-layered system to analyze, detect and recognize the intrusion patterns to reduce the false positive rate and increase the true positive rate during the recognition of the attack patterns. This system outcomes a better results than the existing IDS with reduced FTF rate and increased TPF rate. This multi-layer scheme enhances the IDS with novel framework.

References

[1] Rishabh Gupta Soumya Singh Shubham Verma Swasti Singhal 2017 Intrusion Detection System Using SNORT. International Re-

search Journal of Engineering and Technology (IRJET).Vol 4(4):2100-2104.
[2] Vern Paxson 1999 Bro: a system for detecting network intruders in real-time.Computer Networks (Amsterdam, Netherlands), 31(23-24):2435–2463.
[3] Kruegel C Vigna G 2003 Anomaly detection of web-based attacks. In Proceedings of 10th ACM Conference on Computer and communications Security CCS'03. 251–261.
[4] Mahoney M V 2003 Network Traffic Anomaly Detection Based on Packet Bytes. Proceedings. ACM-SAC.
[5] Lee W Stolfo S 2000 A Framework for Constructing Features and Models for Intrusion Detection Systems. ACM Transactions on Information and System Security, 3(4): 227-261.
[6] Vigna G Kemmerer R 1998 NetSTAT: A Network-based intrusion detection approach. Computer Security Application Conference.
[7] Mahoney M, Chan P K 2002 Learning Models of Network Traffic for Detecting Novel Attacks. Florida Tech, Technical report 2002-08.
[8] Taylor C Alves-Foss J 2001 NATE – Network Analysis of Anomalous Traffic Events, A Low-Cost approach. New Security Paradigms Workshop.
[9] Neelakantan S Rao S 2008 A Threat-Aware Signature based Intrusion-Detection approach for Obtaining Network Specific Useful Alarms. The Third International Conference on Internet Monitoring and Protection. 80-85.
[10] Kruegel C Toth T 2003 Using Decision Trees to Improve Signature Based Intrusion Detection. Advances in Intrusion Detection, Pittsburgh, Pennsylvania: Springer Link. 173–191.
[11] Modi C N Patel D R Patel A Rajarajan M 2004 Integrating Signature Apriori based Network Intrusion Detection System (NIDS) in Cloud Computing. Procedia Technology. 62(12):905-912.
[12] Gupta M Govil M Singh G Sharma P 2015 XSSDM: Towards detection and mitigation of cross-site scripting vulnerabilities in web applications. International Conference on Advances in Computing, Communications and Informatics (ICACCI). Doi: 0.1109/ICACCI.2015.7275912.
[13] GUNDAL, S. S., & NARWADE, P. HANDWRITTEN CHARACTER RECOGNITION USING NEURAL NETWORK WITH FOUR, EIGHT & TWELVE DIRECTIONAL FEATURE EXTRACTION TECHNIQUES.
[14] Vigna G Robertson W Balzarotti D 2004 Testing network based intrusion detection signatures using mutant exploits. In Proceedings of the 11th ACM conference on Computer and communications security(CCS '04) ACM, New York, NY, USA, 21-30. Doi: 10.1145/1030083-1030088.
[15] Chou T 2013 Security Threats on Cloud Computing Vulnerabilities. International Journal of Computer Science and Information Technology. 5(3):79–88.
[16] Heiderich M Schwenk J Frosch T Magazinius J Yang E 2013 mXSS attacks: attacking well secured web applications by using innerHTML mutations. In Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security(CCS '13). ACM, New York, NY, USA. 777-788 Doi: 10.1145/2508859.2516723.
[17] Bronte Robert Shahriar Hossain Haddad M Hisham 2016 A Signature-Based Intrusion Detection System for Web Applications based on Genetic Algorithm. 32-39. 10.1145/2947626.2951964.
[18] Monther Aldwairi Ansam M Abu-DaloMoath Jarrah 2017 Pattern matching of signature-based IDS using Myers algorithm under MapReduce framework. EURASIP Journal on Information Security.