Volume 3, No. 12, December 2012
Journal of Global Research in Computer Science
ISSN-2229-371X
RESEARCH PAPER
Available Online at www.jgrcs.info

INTRUSION DETECTION SYSTEM FOR WEB APPLICATIONS WITH ATTACK CLASSIFICATION

Talasila Vamsidhar[1], Reddyboina Ashok[2] and RayalaVenkat[3]
[1]Asst.Professor, (CSE), DVR & Dr HS MIC College of Technology, Kanchikacherla, KRISHNA, A.P, India
[1]talasila.chinna@gmail.com
[2]Asst.Professor, (CSE), DVR & Dr HS MIC College of Technology, Kanchikacherla, KRISHNA, A.P, India
[3]Asst.Professor, (CSE), DVR & Dr HS MIC College of Technology, Kanchikacherla, KRISHNA, A.P, India

Abstract -Nowadays, web-based applications are being widely used. The exorbitant variety of these applications and their development by the programmers, who don't have much experience on the field of security, Intrusion methods and preventing them, have caused these applications to turn in to some kind of a challenge for the web-servers. This article is an attempt to increase the awareness of the system manager about the possible occurrence of these attacks, by presenting a detection system for web-based applications abnormalities, and by using the requests received by the web-server which can be available through the web-server Log files. The system, detects the abnormality in two separated phases of "Education" and "Detection". The education phase is an attempt to make a model based on the normal behavior of the system by extracting the needed features through the Log files, and using the model in the next level to detect the abnormal input data in the web-server. The system is made of several modules, and each module has the responsibility of reviewing a specific kind of abnormality. This system also uses another module to detect the class of the occurred attacks to the web-server, so that the system manager will be able to prepare some preventive measures by dividing the attacks into separated groups. The obtained results of implementing this method, shows its high ability of detecting the unknown attacks.

Keywords—Intrusion detection, Text Mining, Machine learning, Classification

INTRODUCTION

Nowadays, Web-servers are considered to be one of the main and sensitive elements for many organizations. In the other hand, because of the importance of these web-servers, attacking against them has become very common these days.

Unfortunately, the application layers are basically written by people who don't have much information about the attacks and the security in web-layer, and this will cause the reduplication of the vulnerability of these applications. In the other hand, since these applications are normally available on port 80, the attacks cannot be prevented by using the existing firewalls, because if we want to provide the accessibility to the relevant websites, the port should be available to everyone. Some of the intrusion detection systems based on web, are configured according to the signature, and as a result, several well-known attacks are detected.

Since each signature represents an individual kind of attack, in order to defend the attacks we need a large database of signatures which can be updated regularly. Whereas, anomaly detection systems don't need a large database of signatures and use less memory space on the system, they will be able to detect unknown attacks as well. In the proposed system, we model a normal behavior for each one of the applications on the web- server by using the reviews related to the existing logs in web-servers and extracting the received requests, and then use them as raw data to create a model based on these requests in the Education phase, after that, in the Detection Phase, by reviewing the received requests from the server and comparing them with the existing models for each application, any kind of abnormality will be reported by the various modules and in the end, the system will announce whether the requests are normal or abnormal. If the received request is an abnormal one, the system will also announce the type of the detected attack by using the module which reviews the level of the attacks. The details of the proposed plan are presented in the 3rd section. The intrusion detection community has been deals mainly on wired networks, but it is lack of security in wireless networks. Anomaly detection and misuse detection or signature detection are the two techniques used for intrusion detection system. Anomaly detection describes the abnormal patterns of behavior, where "abnormal" patterns are defined beforehand. Misuse detection relies on the use of specifically known patterns of unauthorized behavior. Thus these techniques rely on sniffing packets and using the sniffed packets for analysis. In order to realize these ID techniques the packets can be sniffed on each of the end hosts. This is called as host intrusion detection (HID). It is also possible to sniff these packets on certain predetermined machines in the network. This is called as network intrusion detection (NID).

Mobile agents are a special type of agents defined as "processes capable of roaming through large networks such as the adhoc wireless network, interacting with machines, collecting information and returning after executing the tasks adjusted by the user". The nature of mobility in a wireless networks creates vulnerability due to the open medium, dynamically changing networks. In order to avoid such circumstance, to develop new architecture and mechanisms to protect the wireless networks and mobile computing applications.

RELATED WORK

One of the efforts in the field of classifying the web-based

© JGRCS 2010, All Rights Reserved

attacks which has occurred in year 2006 was made by Robertson [6]. In this effort, with the use of exploratory methods, the proper class for each group of occurred attacks will be determined.This approach will recognize the following four categories of attacks:

a. Butter Overflow Attacks
b. XSS Attacks
c. SQL Injection Attacks
d. Directory Traversal Attacks

The dependence of the system on the exploratory functions written by the security experts is considered to be one of the weak points of it.

Also, in order to diagnose the web based abnormalities by using the abnormality distribution techniques, it will be able to turn the abnormal requests, in to signatures [6]. The created signature can be used to classify the similar abnormal requests. In addition, the presented approach can extract the type of the attack which has produced the warning of the abnormality by using an intelligent technique. In order to classify the produced warnings in this method, a set of functions are created by the security experts for each one of the incoming attacks. After clarifying the abnormality of the request, it must be compared with the functions created for the attacks, and then the class of the attacks will be extracted.

In year 2009, a system was presented for classifying the HTTP attacks [16]. The main purpose of this system was to identify the abnormal codes in the incoming requests and eliminating them before they were processed by the system. In order to create the classifier, this approach will use the Vector space model which is normally being used for the data recovery purposes. The presented method which is based on a statistical approach or the calculation of the frequency of the repeating words, is trying to classify the attacks. The significant point is that in the classification of the attacks, presence or absence of a key word in the input string cannot be a proof whether the attack will occur or not.

In year 2009, a new approach was presented for automatically classifying the attacks identified by the new abnormality identifying system [4]. In this method SVM, Ripper algorithms which are considered to be machine learning methods, were used to create the classifier system. The whole idea is like this, same attacks have some bit strings similar to each other and by extracting the bit strings from the requests and comparing them with the previously collected normal strings, the class of the occurred attacks can be identified.

In year 2010, another technique was proposed in order to detect the attacks against the web-based applications, which uses the hidden Markov model to detect the abnormality [5]. This system uses the log files to process the queries of the web-based application in the education phase. In order to detect the attacks, modules which are created for the web-based application are being used in this approach. And the system will be able to detect the occurred attacks on a particular set of applications. In this system, each application will assign a module which contains a number of hidden Markov models. Then its URI input parameters will be extracted and it will enter these Markov models as the input data. The output of these Markov models is a possibility which indicates the existing similarity between the input data and the educated model.

Our proposed system also uses the server's log files to model the behavior of the applications and tries to detect any kind of an abnormal behavior by changing the requests received on the web server into distinct tokens and using the Finite state machine. As we continue, the building and using methods will be explained.

PROPOSED APPROACH

The proposed anomaly detection system for web is constructed by several modules and each one of them is responsible for a part of the stage of investigating and detecting the requests received by the server. An overview of the system is presented in fig 1. As we continue, we review each part of the system.

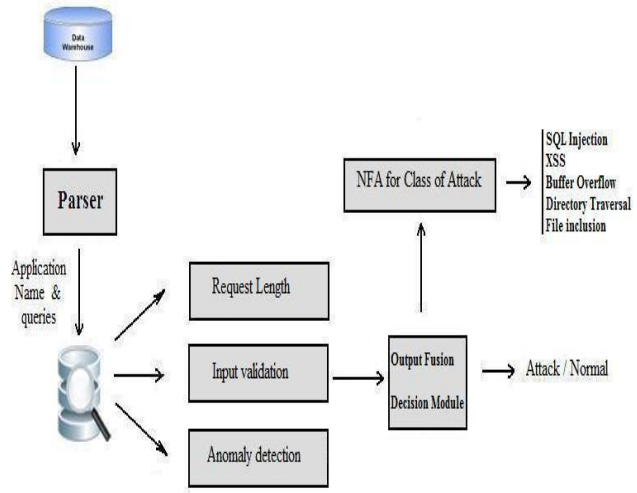

Figure.1. An overview of proposed system

Dataset:

In order to test and simulate the proposed system, we need a set of data which can simulate a situation similar to the existing traffic and attacks in the real life. Unfortunately, obtaining a dataset like that is a bit difficult because of the existing personal and important data of various organizations in the data collection, and most of these organizations, simply refuse to share this kind of data. For that reason, a dataset is produced for the test and experimental use of the researchers. We refer to them in the fallowing kind of data. For that reason, a dataset is produced for the test and experimental use of the researchers. We refer to them in the fallowing.

The MIT University Dataset:

The most famous Dataset for testing the anomaly detection systems, is the "1998, 1999 DARPA/MIT Lincoln Laboratories" Dataset. The set of data which has been collected in this dataset is sampled from the real traffic in the network, in which some systems act as the attackers and some of them act as the producers of the traffic. This dataset is right now being used as a "Test dataset for anomaly detection systems" by many researchers as well

© JGRCS 2010, All Rights Reserved

[12].Although this dataset has played a huge helping role in the development of the anomaly detection systems, it is not perfect in the end. In [13], it has been referred to the point that in the process of producing this dataset, no testis presented based on the reality of the produced data. Also, there's no proof available to make sure that the existing attacks in this dataset are based on the attacks in the real world.

Also, the relevant dataset is in the tcpdump1 format, but none of the existing HTTP requests in it, include the attacks in the values of the parameters [3]. Therefore, the relevant dataset is not suitable for our proposed system which tries to make the model based on the input parameters.

CML/PKDD 2007 Dataset:

In year 2007 a dataset of normal and abnormal HTTP requests were presented in a format of an XML file, in order to participate in a competition of creating an anomaly detection system, named "ECML/PKDD 2007Discovery challenge" [8].

This dataset is divided in to two groups of test and education. In the education phase, a normal or abnormal sticker has been attributed to each one of the samples. This dataset includes approximately 50000 requests which20000 of them, contain an attack. The main target of the competition is to classify the attacks based on the context and separate the existing pattern of the attacks in the relevant requests. The other two datasets are designed based on the log files of the web server in many universities by Dr.Ariu and Dr.Igham [5, 1], which are suitable for meeting our needs in simulating the proposed system. But since these 2datasets are private, and contain susceptible data, using them is not possible for the public. So we decided to prepare this dataset by ourselves.

In order to do that, the process of collecting and preparing this dataset from many different web servers was done during one month, and in the end, after reviewing and sampling, 61 observed applications and the requests they have received was collected.

Table 1: The Collected Dataset

Dataset	Number of all the records	Number of records containing an attack
Education	1600	0
Test	15000	2832

192.168.176.2--
[22/Jun/2011:19:06:06+0430]"GETkayer/administrator/index.php?option=com_content&view=articles HTTP/1.1" 200 193403

http://192.168.176.128/kayer/administrator/index.php?option=
com_content&view=article&layout=edit&id=76"Mozilla/5.0 (Windows NT 6.0; rv:5.0) Gecko/20100101 Firefox/5.0"

192.168.176.16 - - [22/Jun/2011:19:22:21 +0430] "GET /kayer/administrator/index.php?option=../../../../../../../../../../etc/
Passwd%00 HTTP/1.1" 200 5706 "-" "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0)"

In addition to the existing attacks in this dataset, a few attacks were also organized manually. This was done by extracting the attacks in this dataset by searching through the web sites which report the discovered vulnerabilities every day2, then running each one of these exploits on the applications, and using the relevant log files for our test dataset. In the following, the IDs of some of these attacks are presented.

BugTraq ID = {35836, 44060, 39307, 38784, 39184,47281, 41875, 38895, 37854, 27842}

Parser:

This module has the responsibility of extracting applications name and the requests received by them, from the existing records in dataset. After extracting theserequests, they will be preserved in a database which includes the name of the application and the received request. The reason of using a database in this system is to preserve the requests which have been received by each application and make a model of each one of them, and also implementing fast queries in the test phase.

After extracting the required feature in the dataset, in order to build the model in the education phase or comparing the received request with the premade models in the test phase we send them to the detection modules.

Request Length Module:

We can use the length of the input requests to the web server, to detect the occurred attacks [11], if μ is considered to be the average length of n requests received by an application with the parameters of $L_1,L_2,L_3,\ldots\ldots,L_n$in which **Li** represents the length of the received requests of i, and σ2 will be the variance of these requests, then according to the relation 1, the possibility of P for a request with the length of I will be as the fallowing.

$$p = \frac{\sigma^2}{(l-\mu)^2} \qquad (1)$$

The values of μ and σ^2 are calculated separately in the education phase according to the received requests. After calculating these values in the test phase and while considering the pre-defined values and the size of the newly received request, the value of P will be calculated and if it's higher than a threshold, that request will be considered as an anomaly request. On fig 2, you can see an example of the normal and abnormal requests which can be detected by this module. This method can easily detect attacks like Directory Traversal and Buffer overflow. Because these attack inherently, have request sizes larger than the normal size.

© JGRCS 2010, All Rights Reserved

Figure. 2. An example of the attacks which can be detected by the Request Length Module

Input Validation Module:

This Module uses the finite state automata and tries to make a DFA, based on the observed parameters and their values in the request. In the end of the education phase, for each one of the observed applications in the dataset and the dispatched variables, we have a model which contains the name of the variable, their order of priority and the values. These models are made according to the requests which have been received by each one of the applications. Making this model, will be a great help for detecting the occurred abnormalities in the input data for each applications. In the testing phase, for each one of the applications which is being observed, first the specifications of the application and the relevant model will be loaded from the database and with analyzing of the observed input parameters in the new request and comparing them with the model which has been made before in the education phase.

If an input request isn't similar with model, a negative point of anomaly will be given to the request by the module. Fig 3 shows a model which has been made based on the observed parameters in the request. Since observing all of the parameters may not be possible in one request, we're going to have more than one final state in the end.

Figure. 3. The created model based on the observed parameters in the request

Assume that after Var1, Var2 a new parameter named Var5 is observed which has not existed in the DFA, when observing a new parameter which has not been in the request we assign one negative point of anomaly to the token. Also in this module, the type of the input parameters will be learned based on reviewing their values in the education phase. in the testing phase, the values of each parameter is being reviewed and for example, if the value of String is assigned for a specific parameter which has always had the value of Integer in the education phase, this case will be considered as an anomaly.

Anomaly Detection Module:

If we pay attention to the requests which have been received by the web server, we can notice that the requests which only include words or digits or a combination of these two, can never contain attacks against the web server [5]. By assigning non-alphabetical characters from the URL chain, we can detect the anomalies occurred in these requests. The working method of this module is also based on the exact same method. To do this, we encode the requests received by the server before they're reviewed in this module.

a. We put an A for each alphabetical character which has been observed in the requested chain.
b. We put an N for each digit which has been observed in the requested chain.
c. For each pre-defined character like =, {.,?, &, _, = }, in the request, we put them in the chain as they are.
d. For other cases, we replace them with H.

For example, the input chain of (1) on the web server will enter the Anomaly detection module like this after being encoded.

kayer/administrator/index.php?option=com_content&view=
art icles&articles_id= -1"; (1)
kayer/administrator/index.php?AAAAAA=AAA_AAAAA
AA
&AAAA=AAAAAAAA&AAAAAAAA_AA=HNHH

Then with creating the fallowing automata, we will change the status and move to a new state when observing each one of the characters, and per each transition to the H state, we assign an anomaly point for the request and in the end, after subtracting all of the points, we send the total anomaly points of the request received by the sever in to the decision makingmodule.

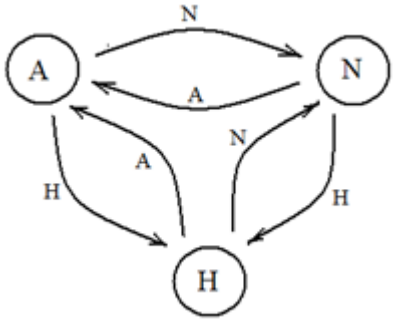

Figure 4. State transition in the anomaly detection module
3.6 The decision making module

As it was mentioned in the previous sections as well, by reviewing the input request in the Length Request and Anomaly Detection modules, an anomaly point will be assigned for the relevant request and it will be sent to the Decision Module. In this Module, by comparing the assigned values of each request and comparing it with a threshold value, a decision will be made whether the request is a normal or an abnormal one. Making a careful selection for the value of this threshold, plays a very important role in the anomaly detection system's decision on detecting whether the request is normal or abnormal, and the value of False Alarm and Detection Rate has a direct relation with the selection of this threshold's value.

IMPLEMENT AND COMPARISON

The remaining results of an anomaly detection system are presented by using the ROC3 graph which represents the comparison between the two values of True Positive and False Positive. The first item shows the amount of the existing attacks in the dataset which have been detected correctly, and the second item is for presenting the normal requests which have been falsely detected as attacks.

False Positive Rate =

$$\frac{\text{Number of normal Requests incorrectly identified as attack}}{\text{Total Number of Normal Request}}$$

True positive Rate =

© JGRCS 2010, All Rights Reserved

$$\frac{\text{Number of attacks Correctly Identified}}{\text{Total number of attack requests}}$$

The simulation levels are based on two phases of education and test. First we put the system in the education phase and by reading the dataset which has been mentioned before and making the models and… this phase will be done. Then we put the system in to the test phase and by reading the test dataset which contains the attacks and comparing it with the pre-learned models, it will be announced whether the request is a normal or an abnormal one. Since the design of the system is modular, each module is efficient for detecting a specific kind of attacks.

Due to implementation result, the most detection ability is achieved by the Anomaly Detection Module. The proposed system is simulated with the Perl 4 programming language and works in the Centos5 Linux Operating system. The following figure shows the implementation result with detection rate of each module in certain class of attack.

With regard to the essence of the Algorithm which has been used in the Request Length Module and the given explanations, and as it is obvious in the results, there is a high expectation for the detection rate of the Buffer Overflow attacks. Because these attacks naturally have large sizes in order to cause the overflow in the buffering. On the contrary, this module has a lower rate in the case of input and XSS attacks, because these attacks have some specific characters which the relevant algorithm is usually unable to detect.

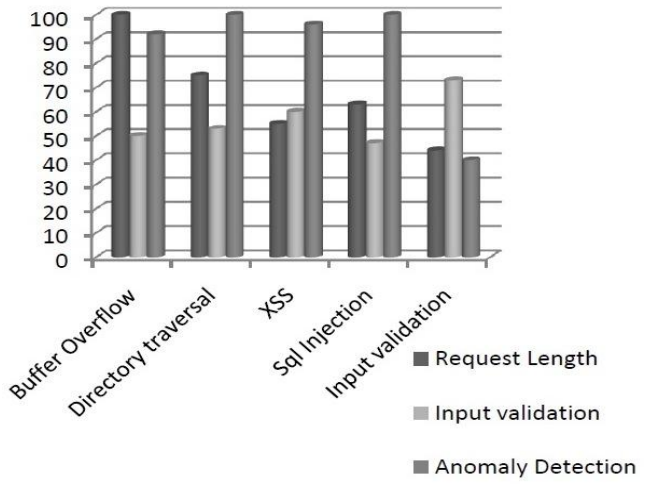

Fig. 5.Detection rate of each module.

As it is expected from the essence and functional method of this module, it should be sensitive to attack containing any kind of abnormalities, and as it is obvious based on the results, the input validation attacks have a high rating in this module.Since this module is somehow searching for the anomaly characters in the input request chain, it has a high ability of detecting any kind of abnormality in the input data, which is pretty obvious in the SQL injection, XSS, Directory Traversal attacks and the presented results.

The module for identifying the class of the Attacks:

In order to identify the class of the occurred attacks, another module is used. This way, when an incoming request is identified to be an attack, the class of the occurred attack can be identified as well. The general mechanism of this module's function is similar to the Anomaly detection module.

With the analysis performed on each category of the occurred attacks, it was determined that a set of common words are being used in each one of the classes and by every attack being occurred, all or some of these words are present. Then, similar to the method which has been used in the Anomaly Detection Module, we encode each one of the input strings. However, for observing each group of these common words, we put a specific character in the string. The following table presents a general view of how this method works. In the end, similar to the Anomaly Detection Module, we use a DFA to rate the input string and when the process is done, if the ratings of one class is higher than the others, the input string will be dedicated to the latter class of attack.

Table 2: The Common Words Of Each Class With The Alternative Character

Alternative character	Common Words	Class of attack
S	Select,insert,update,union,--,having,orderd by, group by, from, where,set	SQL Injection
X	<script>,alert, document, <javascript>,http://,</script>,write,	XSS
D	../ , ./ , .. , /. , include	Directory traversal
B	Exec , exe , nop , /c , % ,\x	Buffer Overflow

During the implementation of the Detection Module of the class of the occurred attacks, a threshold is set to minimize the possibility of making any kind of an incorrect detection by the module. This threshold is playing a huge role especially for cases in which the common words in the input string are used as the application's variables and practically, don't implement any kind of an attack. If the number of the common words observed in a string passes a certain limit, this module will categorize the string with a specific class of attacks.

The following figure represents the rating of the detection after the process of implementing the module is done for three different thresholds.

In year 2011, a web-based intrusion detector system was implemented by using the Markov model which was making the effort to detect the class of the occurred attacks automatically, with the usage of the Markov model. In the following figure, you can observe the comparison between the results for the implementation of this method with the threshold of 1, with the proposed method being mentioned in this paper.

© JGRCS 2010, All Rights Reserved

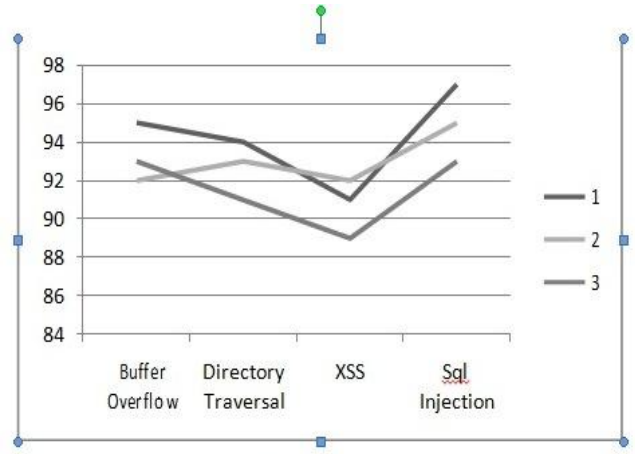

Fig. 6. The detection rate results of the implementation of the attack classifier module

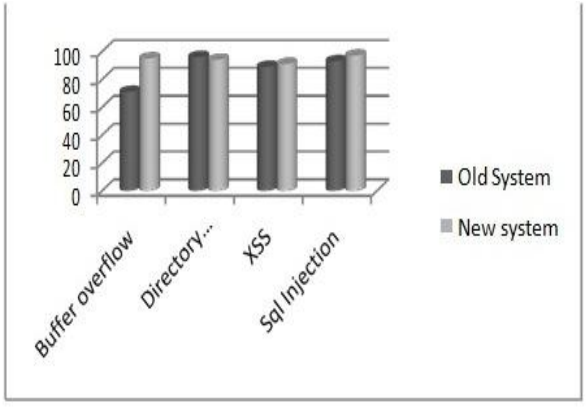

Fig. 7. the comparison between the results for the proposed system and a similar system

As it is obvious, except for the Directory Traversal class, the proposed system has been able to create more favorable results.

For a more specific review on the results, we're going to compare the total output of the system, which all the modules are used in it, with a similar system. In year 2007, the anomaly detection module for web, which extracts the needed Tokens from the HTTP requests and tries to model the normal and abnormal requests received by the web server by using the DFA, was created by Doctor Ingham. Fig 6 presents a comparison between the two systems. As it is obvious in the picture as well, when the threshold value passes a certain limit, the True Positive rate will stay fixed, and it will cause the system to make a False Positive rate.

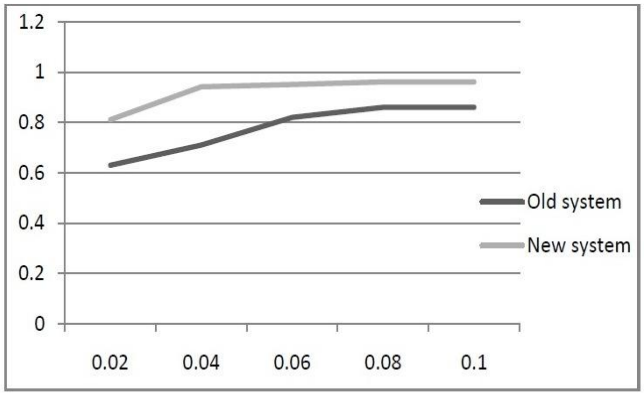

Fig. 8.The comparison of the proposed system with a similar one

CONCLUSION

This paper presents a web application IDS for detecting attacks against web servers based on input requests by using some heuristic algorithms such as request length, input validation with DFA and anomaly detection for input URL. Each algorithm try to detect certain kind of attacks and total system attacks will be available due to maximum attack detection of each module. Although the level of sensitivity to anomalous data can be configured via thresholds to suit different site policies. The system has been tested on data gathered at websites with 61 separate applications, showing promising results.

Attack classification also included to the system, this modules try to suppose class of occurred attacks and help system manager to easily identify attacks. Future works will focus on further decreasing the number of false positives by refining the algorithms developed so far, and by looking at additional features. The ultimate goal is to be able to perform anomaly detection in real- time for web sites that process millions of queries per day with virtually no false alarms.

ACKNOWLEDGEMENTS

We are greatly delighted to place my most profound appreciation to Dr. K.B.K Rao **Principal of DVR & Dr HS MIC College of Technology**, C.V.V.D.Srikanth **Head of CSE Department and**Dr.A. Jaya Lakshmi**coordinator for Research and Development (CSE) in MIC** under their guidance and encouragement and kindness in giving us the opportunity to carry out the paper. Their pleasure nature, directions, concerns towards us and their readiness to share ideas enthused us and rejuvenated our efforts towards our goal. We also thank the anonymous references of this paper for their valuable comments.

REFERENCES

[1]. K.Ingham, "anomaly Detection for http intrusion detection: algorithm comparisons and the effect of generalization on accuracy" Ph.D. dissertation ,Department of computer science , University of new Mexico, 2007

[2]. G.Spathoulas ,S.Katsikas , "Reducing false positive in IntrusionDetection systems" , Computer and Security, 2010

[3]. R. G. Sriraghavan ,"Data processing for anomaly detection in web based applications" ,Master of science dissertation, Oregon State University ,2008

[4]. D.bolzoni ,S.Etalle ,P.H. Hartel ," panacea: Automating attackclassification for anomaly based network intrusion detectionsystems" , 12th International symposium on recent advances inIntrusion detection, 2009

[5]. D. Ariu , "Host and Network Based anomaly detector for HTTP attacks", Ph.D. dissertation, Department of computing science, University of Cagliari 2010

[6]. W.Robertson, G,Vigna, C.kruegel and R.kemmerer ,"Using Generalization and characterization techniques in the anomaly based detection of web attacks", 13th annual network and distributed system security symposium, san diego 2006

© JGRCS 2010, All Rights Reserved

[7]. C.krugel ,Gvigna "Anomaly detection of web-based attacks",ACM conference on computer and communications security,2003

[8]. 18th European conference on principles and practice on machine learning (ECML) and the 11th conference on principles and practice of knowledge discovery in database (PKDD) 2007

[9]. S.Axelsson . "Intrusion detection systems: A taxonomy and survey" technical report, Department of computerengineering ,University of Sweden 2000

[10]. S. Yun Wu ,E.Yen , "Data mining Based Intrusion Detectors" ,Expert System and application , 2009

[11]. C.krugel ,G.Vigna and W.K Robertson, "A multi-model approach to detection of web-based attacks" Computer network 2005

[12]. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[13]. J.Machugh, "Testing Intrusion Detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory", Presented at ACM Trans.info.Sys.Secur,2002

[14]. C.Krugel, T.Toth, E.Kirda , " Service Specific Anomaly detection for network intrusion detection",Distributed Systems Group, Technical University of Vienna, 2002

[15]. E.trameshloo,M.hosseinkhani,B.sadeghian, "A Newtwo Dimensional Approach for Detecting Input Validation Attacks Based on HMM", Dept.of Computer Engineering and Information TechnologyUniversity of Amirkabir, 2011

[16]. B.Gallagher,T.Eliassi-Rad. "Classification of http attacks: A studyon the ecml/pkdd 2007 discovery challenge", 2009.

© JGRCS 2010, All Rights Reserved