Detection and Implementation of Web-based Attacks using Attribute Length Method

Snigdha Agrawal
B.Tech Student (IT)
Bharati Vidyapeeth's College of Engineering, New Delhi

Priya Gupta
B.Tech Student (IT)
Bharati Vidyapeeth's College of Engineering, New Delhi

Vanita Jain, Ph.D
HOD(IT)
Bharati Vidyapeeth's College of Engineering, New Delhi

Achin Jain
Assistant Professor (IT)
Bharati Vidyapeeth's College of Engineering, New Delhi

ABSTRACT

With the increasing demand of web-based applications, they have become more prone to be exploited by the attackers. The purpose of this paper is to study the effects of web-based attacks and analyze the log files generated during the attacks. We have implemented Attribute Length Method proposed by Krugel for the detection of web-based attacks. In the implementation of the Attribute Length method, two different phases are used in our system i.e., learning and detection phase. In the learning phase, our implementation in Java trains the normal dataset and calculates the threshold probability value which is used in the Detection phase for the estimation of web-based attacks. In order to estimate the performance of attribute length method, we have used a log file having three different web attacks, i.e. Cross Site Scripting attack, Path Traversal attack, and Buffer Overflow attack. This method is more effective as we have considered the parameters as fixed-size tokens.

Keywords

Web Based Attack, Attribute Length Method, Cross-site Scripting Attack (XSS), Buffer Overflow attack, Path Traversal Attack

1. INTRODUCTION

The internet access has become very common due to the user conveniences such as surfing, posting, and uploading being provided by the web applications. There is a lot of sensitive information accessed by web applications such as financial data, medical records, intellectual property, social security numbers, and national security data which is crucial to customers, organizations and community. To maintain integrity it is necessary to secure web applications [17].

Among all kinds of cyber attacks, web based attacks are considered to pose prodigious risks related to confidentiality, availability, and integrity. The persistence of a web-based attack is considerably unlike other attacks. Web-based attacks function on layer 7 of the OSI and concentrates on the application only [18]. Different web-based application vulnerabilities, offers way to mischievous end users to infringe a system's protection mechanism with the intention to gain access to sensitive information or system resources.[21] Attackers normally try to steal information like credit card information, social security numbers, etc., which are often employed in identity theft [12]. There are five fundamental categories of application attacks: Denial of Service, Information Disclosure, Spoofing, Repudiation, and Elevation of Privileges [1].

All organizations, including Banks, Colleges, Government Offices etc., which maintain a web presence are at risk of attack. In figure 2 application security risk is explained that will help in determining the factors associated with web security. It is very important to analyze the rise in an organization. It can be done by estimating the technical and business impact on the organization after web attack [2].

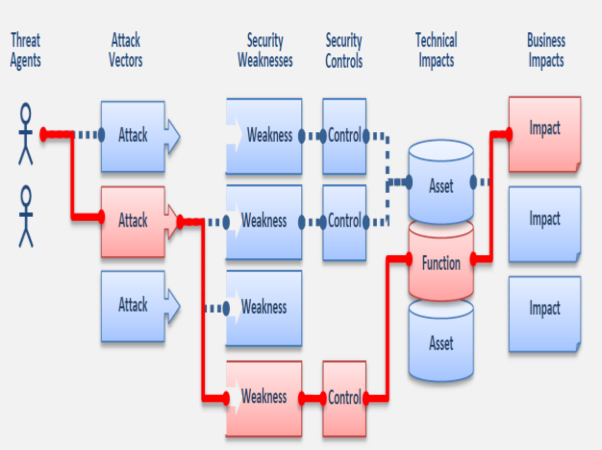


Fig 1: Web application security risk [2]

2. RELATED WORKS

In [5] Bolzoni, Etalle and Hartel modified the original PAYL method to take benefit of the unsupervised classification to use it as a preprocessing stage. They designed an anomaly-based NIDS System is known as POSEIDON that has a two-tier architecture: first stage has SOM, and second stage is a personalized PAYL system. In [6] Shyu, Chen, etc, proposed a system that is used to classify Intrusion Detection problems using a robust principal component. The improvement of the approach is its ability to differentiate the nature of the anomalies whether they are different from the normal instances in terms of tremendous values or various correlation structures.

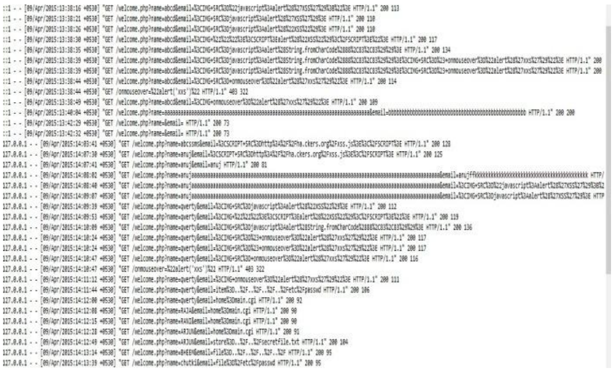
In [7] Kruegel, have used various Anomaly detection techniques to propose an IDS for detecting and preventing web attacks from occurring. Noble and Cook in [8] introduced two techniques for recognizing unusual patterns within a graph-based statistics. They presented a technique for calculating the regularity of a graph that is useful for ruling anomalies, and also for finding the chances of successful anomaly detection within a graph-based data and also calculated the regularity of a graph, using the model of conditional entropy. In [9] Maxion and Tan have found that fundamental structure greatly affects the probabilistic discovery. They have proposed metric for the characterization formation in data environments. Tapiador, Teodoro, and Verdejo in [10] have proposed a new approach which was based on incoming HTTP request monitoring. In that approach authors has used the Markovian model to detect web based attacks. In [11] Gomez and Dasgupta proposed a technique -genetic algorithm to create a fuzzy system that is able to detect anomalies and some particular intrusions. The aim behind the design of a classification process for the intrusion detection problem is to allow use of fuzzy logic and genetic algorithms for the detection of a variety of attacks. Tapiador, Teodoro, and Verdejo presented a new anomaly-based approach that makes use of Markov chains to identify attacks carried out over HTTP traffic. This could enhance the detection capabilities of the current differences in network environments [12]. In [13] Mehta, and Jamwal used QualysGuard WAS tool, to minimize the vulnerabilities to cause any harm to web applications and to detect Cross Site Scripting malicious code which is the main reason to attack the security of the client side as well as server side and QualysGuard WAS tool can satisfy the client needs as well as server needs. In [14] Robertson, Vigna, Kruegel, and Kemmerer presented an approach that uses an anomaly generalization technique for anomaly based detection of web-based attacks and having the advantage that it translates apprehensive web requests automatically into anomaly signatures. They also implemented anomaly signature generation system and developed attack class inference by dropping the attempt needed to analyze the output of the intrusion detection system.

3. WEB BASED ATTACK DETECTION

Intrusion detection approaches can be separated into two techniques, i.e., misuse detection, and anomaly detection [15]. Misuse detection techniques identify attacks as instances of attack signatures. This approach is best suited for the detection of known attacks precisely, but when it comes to detecting unknown attacks this method doesn't work very well. The reason is non-availability of signature of unknown web-based attacks. Anomaly detection [16] overcomes the drawback of misuse detection by concentrating on normal system behaviors, rather than attack behaviors. This approach is characterized in two phases: Learning and Detection Phase. In the Learning phase, the performance of the system is analyzed during normal operation of the process while in the Testing (detection) phase, normal profile is compared against the current behavior of the system and any divergence are flagged as potential attacks. [3]

3.1 Attribute Length Method

This method is based on the fact that the length of attribute in the query string can be used to detect malicious request. This method is most effective in cases where parameter is set as fixed-size tokens. In normal cases when the parameter value is passed for certain case like username (which we have used in this paper for evaluation) doesn't deviate much.[4] However, in the case of malicious input attackers tends to use increase the parameter value. This work mainly focuses on the probabilistic evaluation of three different types of web-based attacks, i.e. Cross site scripting (XSS) attacks, path traversal attack and buffer overflow attack. Out of many anomaly based detection methods that are proposed by different authors we have evaluated detection of web-based attacks with "Attribute Length" technique. We have implemented the ALM approach in Java and have two phases in our system. During the Learning phase threshold probability value is calculated using the normal log file. For detection of web-based attacks, we have used a malicious log file containing both normal and malicious entries. Figure 2. Shows a snapshot of malicious file used for the detection phase.



```
::1 - - [09/Apr/2015:13:38:16 +0530] "GET /welcome.php?name=abcd&email=%3CIMG+SRC%3D%22javascript%3Aalert%28%27XSS%27%29%3B%22%3E HTTP/1.1" 200 113
::1 - - [09/Apr/2015:13:38:21 +0530] "GET /welcome.php?name=abcd&email=%3CIMG+SRC%3Djavascript%3Aalert%28%27XSS%27%29%3E HTTP/1.1" 200 110
::1 - - [09/Apr/2015:13:38:26 +0530] "GET /welcome.php?name=abcd&email=%3CIMG+SRC%3Djavascript%3Aalert%28%22XSS%22%29%3E HTTP/1.1" 200 110
::1 - - [09/Apr/2015:13:38:30 +0530] "GET /welcome.php?name=abcd&email=%3CIMG+%22%22%22%3E%3CSCRIPT%3Ealert%28%22XSS%22%29%3C%2FSCRIPT%3E%22%3E HTTP/1.1" 200 117
::1 - - [09/Apr/2015:13:38:35 +0530] "GET /welcome.php?name=abcd&email=%3CIMG+SRC%3Djavascript%3Aalert%28String.fromCharCode%2888%2C83%2C83%29%29%3E HTTP/1.1" 200 134
::1 - - [09/Apr/2015:13:38:39 +0530] "GET /welcome.php?name=abcd&email=%3CIMG+SRC%3Djavascript%3Aalert%28String.fromCharCode%2888%2C83%2C83%29%29%3E%3CIMG+SRC%3D%23+onmouseover%3D%22alert%28%27xxs%27%29%22%3E HTTP/1.1" 200
::1 - - [09/Apr/2015:13:38:39 +0530] "GET /welcome.php?name=abcd&email=%3CIMG+SRC%3Djavascript%3Aalert%28String.fromCharCode%2888%2C83%2C83%29%29%3E%3CIMG+SRC%3D%23+onmouseover%3D%22alert%28%27xxs%27%29%22%3E HTTP/1.1" 200
::1 - - [09/Apr/2015:13:38:44 +0530] "GET /welcome.php?name=abcd&email=%3CIMG+SRC%3D+onmouseover%3D%22alert%28%27xxs%27%29%22%3E HTTP/1.1" 200 114
::1 - - [09/Apr/2015:13:38:44 +0530] "GET /onmouseover=%22alert('xxs')%22 HTTP/1.1" 403 322
::1 - - [09/Apr/2015:13:38:49 +0530] "GET /welcome.php?name=abcd&email=%3CIMG+onmouseover%3D%22alert%28%27xxs%27%29%22%3E HTTP/1.1" 200 109
::1 - - [09/Apr/2015:13:40:04 +0530] "GET /welcome.php?name=aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa&email=bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb HTTP/1.1" 200 200
::1 - - [09/Apr/2015:13:42:29 +0530] "GET /welcome.php?name=&email= HTTP/1.1" 200 73
::1 - - [09/Apr/2015:13:42:32 +0530] "GET /welcome.php?name=&email= HTTP/1.1" 200 73
127.0.0.1 - - [09/Apr/2015:14:03:41 +0530] "GET /welcome.php?name=abcssms&email=%3CSCRIPT+SRC%3Dhttp%3A%2F%2Fha.ckers.org%2Fxss.js%3E%3C%2FSCRIPT%3E HTTP/1.1" 200 128
127.0.0.1 - - [09/Apr/2015:14:07:30 +0530] "GET /welcome.php?name=anuj&email=%3CSCRIPT+SRC%3Dhttp%3A%2F%2Fha.ckers.org%2Fxss.js%3E%3C%2FSCRIPT%3E HTTP/1.1" 200 125
127.0.0.1 - - [09/Apr/2015:14:07:41 +0530] "GET /welcome.php?name=anuj&email=anuj HTTP/1.1" 200 81
127.0.0.1 - - [09/Apr/2015:14:08:02 +0530] "GET /welcome.php?name=anujaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa&email=anujffkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkk HTTP/
127.0.0.1 - - [09/Apr/2015:14:08:40 +0530] "GET /welcome.php?name=anujaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa&email=%3CIMG+SRC%3D%22javascript%3Aalert%28%27XSS%27%29%3B%2
127.0.0.1 - - [09/Apr/2015:14:09:07 +0530] "GET /welcome.php?name=anujaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa&email=%3CIMG+SRC%3Djavascript%3Aalert%28%27XSS%27%29%3E HTTP
127.0.0.1 - - [09/Apr/2015:14:09:39 +0530] "GET /welcome.php?name=qwerty&email=%3CIMG+SRC%3Djavascript%3Aalert%28%22XSS%22%29%3E HTTP/1.1" 200 112
127.0.0.1 - - [09/Apr/2015:14:09:53 +0530] "GET /welcome.php?name=qwerty&email=%3CIMG+%22%22%22%3E%3CSCRIPT%3Ealert%28%22XSS%22%29%3C%2FSCRIPT%3E%22%3E HTTP/1.1" 200 119
127.0.0.1 - - [09/Apr/2015:14:10:09 +0530] "GET /welcome.php?name=qwerty&email=%3CIMG+SRC%3Djavascript%3Aalert%28String.fromCharCode%2888%2C83%2C83%29%29%3E HTTP/1.1" 200 136
127.0.0.1 - - [09/Apr/2015:14:10:24 +0530] "GET /welcome.php?name=qwerty&email=%3CIMG+SRC%3D%23+onmouseover%3D%22alert%28%27xxs%27%29%22%3E HTTP/1.1" 200 117
127.0.0.1 - - [09/Apr/2015:14:10:24 +0530] "GET /welcome.php?name=qwerty&email=%3CIMG+SRC%3D%23+onmouseover%3D%22alert%28%27xxs%27%29%22%3E HTTP/1.1" 200 117
127.0.0.1 - - [09/Apr/2015:14:10:47 +0530] "GET /welcome.php?name=qwerty&email=%3CIMG+SRC%3D+onmouseover%3D%22alert%28%27xxs%27%29%22%3E HTTP/1.1" 200 116
127.0.0.1 - - [09/Apr/2015:14:10:47 +0530] "GET /onmouseover=%22alert('xxs')%22 HTTP/1.1" 403 322
127.0.0.1 - - [09/Apr/2015:14:11:12 +0530] "GET /welcome.php?name=qwerty&email=%3CIMG+onmouseover%3D%22alert%28%27xxs%27%29%22%3E HTTP/1.1" 200 111
127.0.0.1 - - [09/Apr/2015:14:11:44 +0530] "GET /welcome.php?name=qwerty&email=item%3D..%2F..%2F..%2F..%2Fetc%2Fpasswd HTTP/1.1" 200 106
127.0.0.1 - - [09/Apr/2015:14:12:00 +0530] "GET /welcome.php?name=qwerty&email=home%3Dmain.cgi HTTP/1.1" 200 92
127.0.0.1 - - [09/Apr/2015:14:12:08 +0530] "GET /welcome.php?name=RAJA&email=home%3Dmain.cgi HTTP/1.1" 200 90
127.0.0.1 - - [09/Apr/2015:14:12:15 +0530] "GET /welcome.php?name=RANI&email=home%3Dmain.cgi HTTP/1.1" 200 90
127.0.0.1 - - [09/Apr/2015:14:12:28 +0530] "GET /welcome.php?name=ARJUN&email=home%3Dmain.cgi HTTP/1.1" 200 91
127.0.0.1 - - [09/Apr/2015:14:12:49 +0530] "GET /welcome.php?name=ARJUN&email=store%3D..%2F..%2Fsecretfile.txt HTTP/1.1" 200 104
127.0.0.1 - - [09/Apr/2015:14:13:14 +0530] "GET /welcome.php?name=B+EEN&email=file%3D..%2F..%2F..%2F..%2F HTTP/1.1" 200 95
127.0.0.1 - - [09/Apr/2015:14:13:39 +0530] "GET /welcome.php?name=chutki&email=file%3D%2Fetc%2Fpasswd HTTP/1.1" 200 95
```

Fig 2: Malicious Web Log File

3.2 Methodology

In this Paper we have carried out the experimental work to detect the web based attacks using Attribute Length Method. We have used Java as the programming language to model the ALM method. There are two phases in the system, i.e., Learning Phase and Detection Phase. In the Detection phase, we have analyzed the log file [19] when there is no attack and calculated the threshold probability value (p_{th}). In learning phase, Log file with both malicious and normal request is used to check the effectiveness of the developed software. [20]

3.2.1 Learning

In the process of calculation we have approximated various parameter values i.e., variance σ^2 and mean μ. Using the parameter values, we have calculated threshold probability p_{th} which comes out to be 0.467 for our training data.

3.2.2 Detection

The next step in detection of web based attacks is carrying out the developed model for Malicious web log file. In this process, we have used Chebyshev inequality equation shown below:

$$p(|x-\mu| > t) < \frac{\sigma^2}{t^2} \qquad (1)$$

To compare the log file entries with Pth, we have used the following equation to find resulting probability value p(l) for an attribute with length l [4].

$$p(|x-\mu| > |l-\mu|) < p(l) = \frac{\sigma^2}{(l-\mu)^2} \qquad (2)$$

Figure 3 shows the complete flowchart which has been used to implement the ALM method using Java for web-based attack detection.

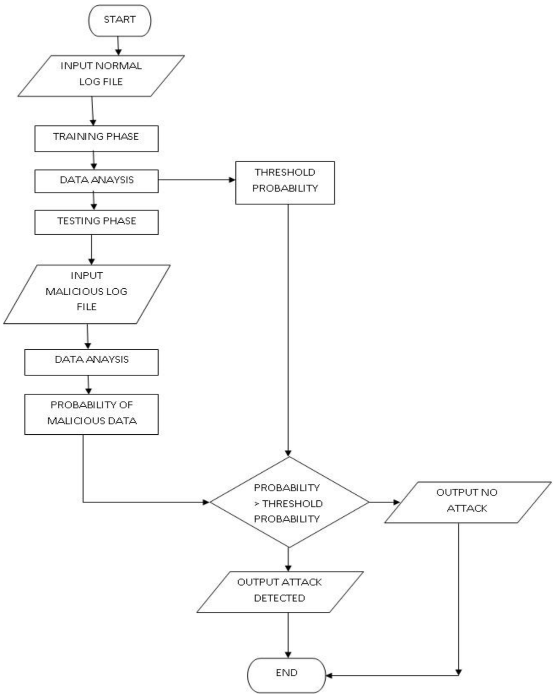
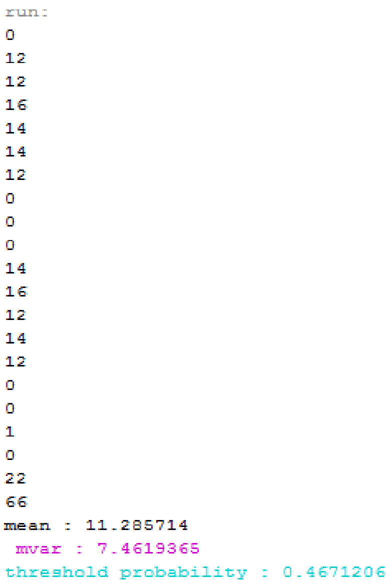


Fig 3: Flowchart for ALM Method Implementation

4. RESULT ANALYSIS

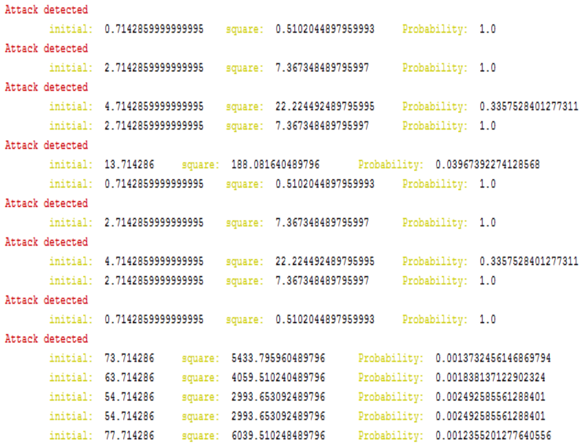
In the Learning phase, we have used a normal log file to train our model. After running the software results obtained are shown in the figure 4.



```
run:
0
12
12
16
14
14
12
0
0
0
14
16
12
14
12
0
0
1
0
22
66
mean : 11.285714
 mvar : 7.4619365
threshold probability : 0.4671206
```

Fig 4: Learning Phase Results

As shown in the figure 4 above, Threshold Probability (p_{th}) for the normal log file we have used comes out to be 0.4671. After training the software, we have used log file with both normal and malicious data. Figure 5 shows the results in Detection phase.



```
Attack detected
        initial:  0.7142859999999995   square:  0.5102044897959993   Probability:  1.0
Attack detected
        initial:  2.7142859999999995   square:  7.367348489795997     Probability:  1.0
Attack detected
        initial:  4.7142859999999995   square:  22.224492489795995   Probability:  0.3357528401277311
        initial:  2.7142859999999995   square:  7.367348489795997     Probability:  1.0
Attack detected
        initial:  13.714286    square:  188.081640489796      Probability:  0.03967392274128568
        initial:  0.7142859999999995   square:  0.5102044897959993   Probability:  1.0
Attack detected
        initial:  2.7142859999999995   square:  7.367348489795997     Probability:  1.0
Attack detected
        initial:  4.7142859999999995   square:  22.224492489795995   Probability:  0.3357528401277311
        initial:  2.7142859999999995   square:  7.367348489795997     Probability:  1.0
Attack detected
        initial:  0.7142859999999995   square:  0.5102044897959993   Probability:  1.0
Attack detected
        initial:  73.714286    square:  5433.795960489796     Probability:  0.0013732456146869794
        initial:  63.714286    square:  4059.510240489796     Probability:  0.001838137122902324
        initial:  54.714286    square:  2993.653092489796     Probability:  0.002492585561288401
        initial:  54.714286    square:  2993.653092489796     Probability:  0.002492585561288401
        initial:  77.714286    square:  6039.510248489796     Probability:  0.0012355201277640556
```

Fig 5: Detection Phase Results

It can be seen clearly in the figure that our software is detecting the attacks and marking the log entry with "Attack Detected" keyword.

Since our log file contains lots of data and it is impossible to show the complete log file, we have plotted a graph showing a threshold value and the probability value of each entry of log files. Figure 6 shown the comparison of p_{th} and Probability value of log file.

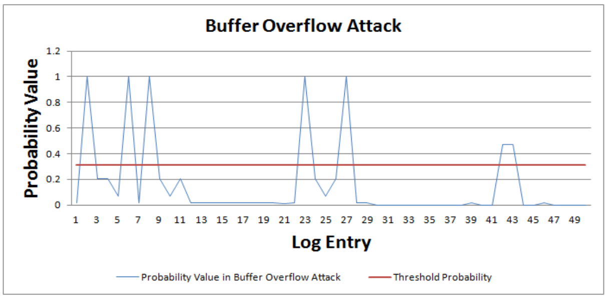


Fig 6: Buffer Overflow Attack

In the figure 6 red line shows the Threshold Probability value and the blue line shows probability values for each log file entry in Buffer Overflow Attack. All the points above Red line are detected as Buffer Overflow Attack in the model and points below the red line are normal requests.

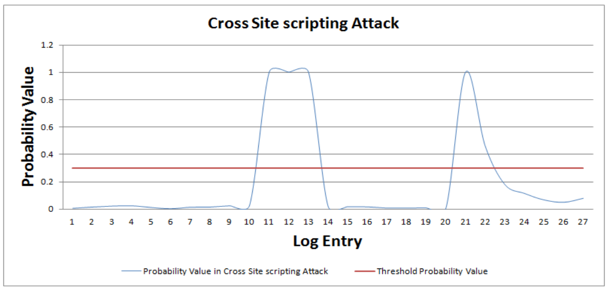


Fig 7: Cross Site Scripting Attack

In the graph red line shows the Threshold Probability value and the blue line shows probability values for each log file entry in Cross Site Scripting Attack. All the points above Red line are detected as Cross Site Scripting Attack in the model and points below the red line are normal requests.

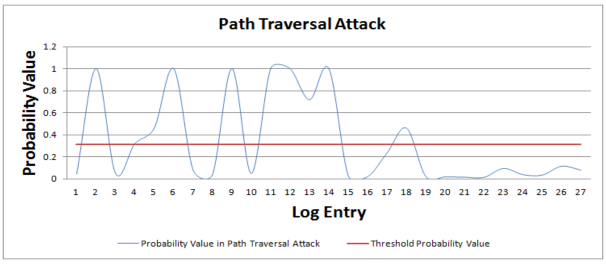


Fig 8: Path Traversal Attack

In the graph red line shows the Threshold Probability value and the blue line shows probability values for each log file entry in Path Traversal Attack. All the points above Red line are detected as Path Traversal Attack in the model and points below the red line are normal requests.

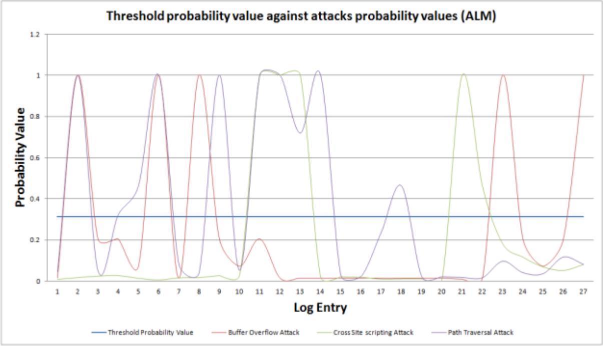


Fig 9: Threshold probability value against attack probability values (ALM)

In the graph blue line shows the Threshold Probability value and the blue line, green line, violet line shows probability values for each log file entry in Buffer Overflow Attack, Cross Site Scripting Attack, and Path Traversal Attack. All the points above Red line are detected as web-based attacks in the model and points below the red line are normal requests.

5. CONCLUSION

Web Server log files in detecting web based attacks is a key parameter, which can be useful to improve the detection of attacks on websites. Analysis of web server log files will be helpful in detecting the malicious scripts that are the basic cause of web based attacks. This paper shows that, analysis of web server log files not only helps in improving the structure of websites, but also plays a crucial role in the detection of web based attacks.

In this paper, we have developed a system in Java that reads and analyses the log files generated during the attacks in order to study the effects of web based attacks. In this work we were able to train our system, and based on the learning our model has detected web-based attacks from log file. After implementation of Attribute Length Method and testing the work on three different web-based attacks, it can be said that ALM method is best suited for Buffer overflow attack.

6. REFERENCES

[1] Justin Crist (2007), *Web Based Attacks*, SANS Institute, As part of the Information Security Reading Room, http://www.sans.org/reading_room/whitepapers/application/web-based- attacks_2053

[2] OWASP Accessed from - www.owasp.org/index.php/Guide

[3] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, S. Zhou, A. Tiwari and H. Yang, (2002) *Specification Based Anomaly Detection: A New Approach for Detecting Network Intrusions*, ACM CCS.

[4] C. Kruegel, and G. Vigna. (2003) *Anomaly Detection of Web-based Attacks*. In 10th ACM Conference on Computer and Communication Security (CCS-03) Washington, DC, USA, October 27-31, pp 251 – 261.

[5] Bolzoni D, Etalle S, Hartel P. (2006) *Poseidon: a 2-tier anomaly-based network intrusion detection system.* In: Information Assurance. IWIA 2006. Fourth IEEE International Workshop. Pp 10.

[6] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, L. Chang, (2003) *A novel anomaly detection scheme based on principal component classifier*, In Proceedings of the 3rd IEEE International Conference on Data Mining, pp. 172–179.

[7] C. Kruegel, G. Vigna, W. Robertson,(2005) *A multi-model approach to the detection of web-based attacks*, Computer Networks 48 (5) , pp 717–738

[8] C. Noble and D. Cook. (2003) *Graph-based anomaly detection*. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp 631–636.

[9] Maxion RA, Tan KMC (2000) *Benchmarking anomaly-based detection systems*. In: International Conference on Dependable Systems and Networks. IEEE Computer Society Press, Los Alamitos, pp 623–630.

[10] Este´vez-Tapiador J.M., Garcı´a-Teodoro P., Dı´az-Verdejo J.E. (2005) *Detection of web-based attacks through Markovian protocol parsing*. In: Proc. ISCC05; pp. 457–62

[11] J. Gomez, D. Dasgupta,(2001) *Evolving fuzzy classifiers for intrusion detection*, in: Proceedings of IEEE Workshop on Information Assurance, United State Military Academy, West Point, NY, 2001, pp. 68–75.

[12] J.M. Estévez-Tapiador, P. García-Teodoro, J.E. DíazVerdejo (2004), "*Measuring Normality in HTTP Traffic for Anomaly-Based Intrusion Detection*", in. Computer Networks, 45(2), pp 145-193.

[13] Tejinder Singh Mehta , Sanjay Jamwal, (2015) *Model To Prevent Websites From XSS Vulnerabilities*, (IJCSIT) International Journal of Computer Science and Information Technologies, 6 (2) , pp 1059-1067

[14] W. Robertson, G. Vigna, C. Kruegel, R.A. Kemmerer (2006), *Using generalization and characterization techniques in the anomaly-based detection of web attacks*, in: Proceedings of Network and Distributed System Security Symposium Conference, 2006, Internet Society

[15] Patcha, A. and Park, J.-M. (2007). *An overview of anomaly detection techniques: Existingsolutions and latest technological trends. Comput. Networks* , *51(*12) pp 3448-3470.

[16] V. Chandola, A. Banerjee, V. Kumar (2009), *Anomaly detection: a survey*, ACM Computing Surveys 41 (3) 1–58. ISSN: 0360-0300, doi : http://doi.acm.org/10.1145/1541880.1541882

[17] Desmond, Paul (2004). *All-out blitz against Web app Attacks Retrieved* December 30, 2006, from networkworld.com Web site: http://www.networkworld.com/techinsider/2004/0517techinsidermain.html

[18] Gartner (2005). *Improve IT Security with Vulnerability Management Retrieved* February 27, 2007, from Gartner.com Web site: http://www.gartner.com/DisplayDocument?doc_cd=127481

[19] Singh, N., Jain, A., Raw, R.S., Raman, R. (2014) *Detection of Web-Based Attacks by Analyzing Web Server Log Files*. In: Mohapatra, D.P., Patnaik, S. eds. Intelligent Computing, Networking, and Informatics. Springer, Heidelberg, pp. 101-109

[20] Joshila Grace, L.K., Maheswari, V., Nagamalai, D. (2011): *Analysis of Weblogs and Web user in Web mining*. Int. J. Netw. Secur. Appl. (IJNSA) **3**(1)

[21] Kolaczek, Grzegorz, and Tomasz Kuzemko.(2014) *Security Incident Detection Using Multidimensional Analysis of the Web Server Log Files* Computational Collective Intelligence. Technologies and Applications. Springer International Publishing. 663-672.