A Novel Design for Smart App Firewall Framework

Hesham F. Ali
Computers and Systems Department, Electronics Research Institute,
Cairo, Egypt.
hesham@eri.sci.eg

Bassem A. Abdullah
Computer and Systems Engineering Department, Ain Shams University,
Cairo, Egypt.
babdullah@eng.asu.edu.eg

Atheer Mostafa
CIO, Starware
Cairo, Egypt
Atheer@startware.net

ABSTRACT

The big and continuously developing web sites, tens of thousands of them redeveloped their code considering secure coding with more interest although this still was not still enough because a single critical vulnerability in a single line in a single page could lead to full website hack and all information disclosure.

In this paper we propose a system aiming to design application for firewall, depending on using the learning based whitelist rules in addition to the classical method of using black list rules. The concept was evaluated by generating these white rules manually and showed excellent performance when tested in websites of STARWARE company clients who involve high traffic daily like "Youm7" Press. In this paper, we target applying an artificial intelligent learning engine such as Artificial Neural Network (ANN), Support Vector Machines (SVM), etc to automate updating the rules of the whitelist extending the concept proved in our commercial platform.

The proposed application is targeting enhancing the system performance by minimizing the traffic with percent more than 90% from internet traffic and web page download time will be reduced by more than 50%. The filters to be build is targeting better detection with more than 50% with respect to the running today filters applying on more 10K rules.

Keywords: Blacklist, Whitelist, Neural network, DNS, Firewall, Varnish, Reverse Proxy.

1. INTRODUCTION

Almost all -even big and famous- websites and services till now have been hacked several times. Recently, website hackers have begun to develop attacks that target vulnerabilities in the business logic, rather than in the code itself. Business logic attacks are often not looked upon as security risks but hold serious business implications for website owners because they generally remain undetected [1].

The well-known brands firewalls and expensive network security devices are not able to cover this aspect because all these vulnerabilities are standing in the application layer; the seventh layer in Open Systems Interconnection (OSI) model, and are very hard to detect without fully understanding the logic of the developer while being able to control successfully up to the fifth layer in the network OSI model.

The most common example of this is comment spam. This is where hackers insert automatically generated comments into a blog or online forum, directing people to bogus sites that promote bogus pharmaceuticals when it's actually malware. The implications of such attacks can range from degradation in your company's search engine rankings to being blacklisted and completely removed from search results. So, all data used by the website (from users, other servers, other websites and internal systems) must be validated for type (e.g. numeric, date, string), length (e.g. 200 characters maximum, or a positive integer) and syntax (e.g. product codes begin with 2 letters and are followed by 5 digits) and business rules (e.g. televisions can only cost between 2000 EGP and 8000 EGP, an order can contain at most 20 items, daily credit limit must not be exceeded). All data written as output

(displayed) needs to be safe to view in a browser, email client or other software and the integrity of any data that is returned must be checked [2].

Our solution is based on the growing and widely used caching platform "Varnish". It's well tested and open source for extension and development. We can implement the system in matrix of any clustered servers to provide a cloud based matrix and could be distributed geographically. This will make the solution great stable and performance efficient to perform in high volume traffic with least costs. With the use of one of a famous artificial intelligent engine such as neural network, it will receive the output of the "Varnish" and generate whitelist rules.

In this paper, Section 2 will give an overview for the history and state of the art of the web site security models, Section 3 will represent the proposed system and section 4 will give the conclusion.

2. HISTORY AND STATE-OF-THE-ART

Most of the critical hacking techniques are based on sending commands within data parameters to the web page considering that data inputs will be concatenated later with developer commands to be executed, this is very clear in SQL Injection, Cross Site Scripting, File Inclusion, Buffer Over Flow, Command Execution … etc

On 2003 we considered building a library through an intelligent security code review application to fix the SQL Injection vulnerability specially and replace all input reading inside the ASP code with a security developed library to filter all parameters based on their log of behaviour. The first mode of this security library is logging mode, in which the parameters values are logged and saved in a specific database for some time of using the site pages all features. Then a filter generator will be used to analyze the log database and output specific filtration rules in Visual Basic syntax to be added into the security library and compiled as a DLL. The last step is to bind this DLL with the ASP website, and run the security library in filtration mode using the DLL filters for all parameters that learned, and apply a basic security for the parameters that are in new pages to be logged and added for next use of the security hardening process to add the new parameters rules. Basically the rules were considering parameters data types validation to protect numeric parameters from SQL Injection, and also protect string parameters being escaped by a single quote for SQL Injection too. Simple rules for file inclusion are applied manually to the rules DLL.

On 2011 STARWARE (as a company) started implementing Cloud Cache servers to provide performance and basic security; that is based on Reverse Proxy technology on external system/servers to provide simple static and dynamic content caching rules written in VCL language to be used in Varnish system as a reverse proxy to the original servers. Varnish – as an open source reverse proxy - converts VCL rules to C++ to be compiled as binaries on the fly for very fast execution in memory that enables processing 10,000 to 40,000 requests per second. The main need to this solution was for performance, stability and basic security including a Blacklist parameter filtering in VCL to the Varnish configurations to protect and block the direct and common cases of SQL Injection specially used by tools and vulnerability scanners.

The blacklist model is to exclude and filter the keywords and patterns used commonly in exploiting a security vulnerability, this requires including all keywords and functions and special function in all version used in the database management system. As example, there are hundreds of MySQL functions, other hundreds in SQL Server, others for Oracle, and PostgreSQL …etc. There are usually hundreds of Filter Evasion and Filter Bypassing techniques and tricks on the web; that's why blacklisting is not efficient enough for security against hackers (this can prevent only script kiddies hackers that are using tools without security and development knowledge).[3-6]

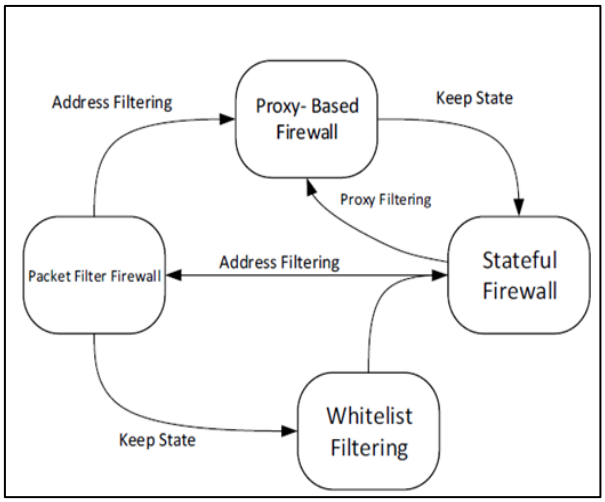

Fig. 1. Diagram for the whitelist Firewall

The Whitelist model is more efficient and this is the model that already implemented while configuring Network Firewalls [7-8], but implementing it in IDS (Intrusion Detection Systems) or IPS (Intrusion prevention system) is so difficult to implement because it requires very detailed and professional rules writing in Regular Expressions which, usually requires a skilled professional with development background to write these rules with cooperation with web application developer. This is not flexible and requires a lot of time and efforts and requires dedication of the developer with the security professional for this task which is not practical and sometimes impossible after handing over of the development company, also it is impossible in compiled and closed code sites to make this kind of security enhancements by whitelisting in most cases

and will result in several issues of incorrect configurations that are hard to debug and troubleshoot. Fig. 1 gives a brief view about whitelist firewall.

3. THE PROPOSED SYSTEM

The proposed system is targeting to automate the rules writing, which requires well knowledge of the parameters (including cookies) in many tracks; the data types, the data length, output structure, output format and output size. In actual web site usage of learning is similar to the basic idea that we implemented simply 10 years ago in the ASP and DLL case, but in more advanced and professional engineering technology.

We can basically write parameters logging to "Varnish" in the learning mode. In this mode we test all site services with normal traffic usage, and then apply this log over a modern machine learning technique to define page specific and parameter specific rules and write their related VCL rules file. Next, implement this file as a security configuration for this web site. The parameters will be validated and filtered in the ordinary "Varnish" fast processing without effect on the original application code or without performance impact. On the other hand, we will gain great performance and high availability features.

Fig 2. Represents a simple comparison between the classical flow between any internet user with any web site with respect to the process between any internet user and any website but within the control from the smart app firewall system

Fig 2: System layout with Smart app firewall versus Classical firewall

The proposed system is composed mainly from 3 phases:
Phase 1: User request analysis.
Phase 2: Controller and Reverse proxy.
Phase 3: Rules generation and Learning.

The system workflow is as follows:
Phase 1: User request analysis

1. User request will be received by smart app firewall.
2. The request will be redirected through Geo-DNS to the cloud network.

Phase 2: Controller and Reverse proxy

3. Cloud network will route the request to the "Varnish" reverse proxy.
4. "Varnish" controller will apply the Whitelist and Black list rules:
    a. Approved request : go to the requested web server
    b. Not approved: block and send to web interface as alert of hack attempt to be displayed to the security monitoring specialist.
5. If the request is from trusted IP, the controller will send it to the learning module.

Phase 3: Rules generation and Learning.

6. The learning module will analyze the request using machine learning techniques.
7. The learning module will generate whitelist rules.
8. The resulted rules will update the application firewall whitelist rules.

Fig. 3 gives a summary to the workflow for a user request as described above.

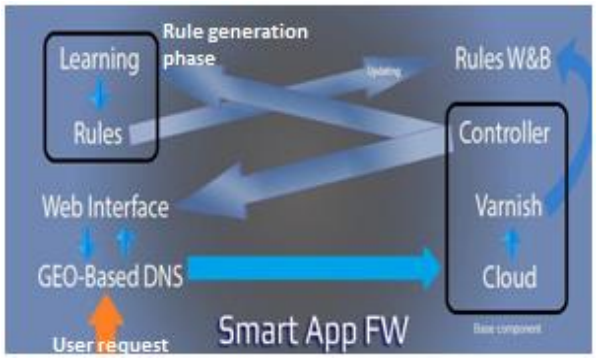

Fig 3: Smart App FW flow

For the whitelist rule engine, the neural network or the SVM used will learn from a trusted user behavior to build a whitelist that will be the source of the rules while the blacklist will remain to be the same used by traditional app firewall. In case of using SVM, one of kernels commonly used like Radial Basis Function (RBF) might be used benefiting from its ability of modeling non-linear models [13].

The proposed research in learning is based on:

- Using Neural Networks [9-11], Support Vector Machines (SVM) [12-14] to build a whitelist rules learning model and other machine learning techniques.
- Comparing between the performances of the machine learning techniques and select the appropriate technique.
- Propose a new learning model if needed that fits the learning of the rule generation phase.

4. CONCLUSION

Implementing machine learning technique for the user usage analysis will be more effective. Being easy to implement external of the web server and even external of the datacenter will enable it to be provided as a cloud service for customers. Cloud Caching & Security is already implemented and used to decrease the traffic usage; this is provided as cloud service for web site owners as CloudFlare or Incapsul and their services are relatively expensive. These systems don't provide learning security rules writing and implement very primitive blacklists for their application firewalls. This is also used by ISPs to reduce the external traffic usage by caching the commonly used sites static content (knowing that static content is about 90% or more of the web page).

The proposed smart application firewall is enhancing the system performance by minimizing the traffic with more than 90% from internet traffic and web page download time is reduced by more than 50%. The filters get better detection with more than 50% with respect to running today filters applying on more 10K rules.

5. REFERENCES

[1] "**9 Things Businesses Need to Know About Web Security**", http://mashable.com/

[2] "**Top 10 website security issues**", Watson Hall Ltd, 2009.

[3] http://www.bbc.co.uk/arabic/worldnews/2013/09/130929_uk_cyber_unit.shtml

[4] Ahmed Amin, H. Farouk,Ahmed Mahmoud, Heba Aslan, "**A Design for an FPGA Implementation of RIJNDAEL CIPHER**", Journal of Programmable Devices, Circuits, and Systems, International Congress for Global Science and Technology (ICGST), 2009.

[5] H. Farouk, A. Tobal, "**An Efficient Detection and Classification Method for Landmine Types Based on IR Images Using Neural Network**", International Journal of Geology, pp. 91-95, Issue 1, Volume 4, 2010.

[6] B. Abdullah, A. Younis and N. John, "**Multi-Sectional Views Textural Based SVM for MS Lesion Segmentation in Multi-Channels MRIs**," The Open Biomedical Engineering Journal, vol.6, 2012, pp. 56-72. doi:10.2174/1874230001206010056

[7] https://www.owasp.org/index.php/Web_Application_Firewall

[8] https://www.owasp.org/index.php/Category:OWASP_Best_Practices:_Use_of_Web_Application_Firewalls

[9] B. Abdullah, A. Younis P.M. Pattany, and E. Saraf-Lavi, "**Textural based SVM for MS Lesion Segmentation in FLAIR MRIs**," Open Journal of Medical Imaging, vol.1, pp. 16-42. doi: 10.4236/ojmi.2011.12005, 2011.

[10] Bassem A. Abdullah, A.A. Younis (UM-ECE_team) in Workshop of "**MS Lesion Segmentation Challenge 08**". Results accepted to appear on the ranked online results on Oct. 10, 2011.

[11] Oliver Coleman, O.J., Blair A.D., "**Evolving Plastic Neural Networks for Online Learning: Review and Future Directions**"., Proceedings of the 25th Australasian Joint Conference on Artificial Intelligence (AJCAI'12), Sydney, Australia, 2012.

[12] Guoqiang Peter Zhang "**Neural Networks for Classification: A Survey**", IEEE transactions on systems, Man and Cybernetics part C: Applications and Reviews, VOL. 30, NO. 4, NOVEMBER 2000.

[13] Chih-Chung Chang and Chih-Jen Lin, "**LIBSVM: A library for support vector machines**", ACM Transactions on Intelligent Systems and Technology, vol. 2, no. 3, pp. 1-27, 2011, Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[14] A Pozdnukhov and M Kanevski, "**Monitoring Network Optimisation for Spatial Data Classification Using Support Vector Machines**", International Journal of Environment and Pollution, vol. 28, 2006.