Detection of Probing Attacks in SWAF

By

Sundas Hamid

2008-NUST-MS PhD-IT-42

Supervisor

Dr. Hafiz Farooq Ahmad

A thesis submitted in partial fulfillment of the requirements for the degree of

Masters of Science in Information Technology (MSIT)

In

NUST School of Electrical Engineering and Computer Science,

National University of Sciences and Technology (NUST), Islamabad, Pakistan

(November, 2011)

APPROVAL

It is certified that the contents and form of thesis entitled **“Detection of Probing Attacks in SWAF”** submitted by **Sundas Hamid** have been found satisfactory for the requirement of the degree.

Advisor: _Dr. Hafiz Farooq Ahmad_______

Signature: ___________________________

Date: ___________________________

Committee Member: _Dr. Khalid Latif_

Signature____________________________

Date:________________________________

Committee Member: Mr. Ammar Karim

Signature ____________________________

Date: ________________________________

Committee Member: _Ms. Sana Khalique_

Signature ____________________________

Date: ________________________________

TO

My Loving Parents,

Brothers

&

Husband

ABSTRACT

Probing is the major issue in web application security but there does not exit reasonable progress to detect probing before the actual attack is launched. The key challenge is to identify attacker's probing process for gathering information of vulnerabilities in Web application and take appropriate actions quickly before attackers exploit them. In this research work, we propose a methodology to detect probing; it is currently implemented as a part of SWAF (Semantic Based Web Applications Firewall) project. It assists SWAF to detect probing before an attacker is able to exploit vulnerabilities. Most of the vulnerabilities are discovered as a result of trial and error by the attacker. We make it possible to detect probing by using three techniques viz. XML rules, SWAF log and application profiling (together comes under threshold learning) and carrying out behavioral analysis of the attackers traffic to detect and block them. The proposed methodology increases the detection rate of SWAF and considerably decreases the attack ratio. As a part of this work we have also evaluated the performance of SWAF with probing detection technique using most popular scanners. Evaluation results confirm the effectiveness of proposed approach as it detects scanners with high detection rate.

CERTIFICATE OF ORIGINALITY

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person nor material which to a substantial extent has been accepted for the award of any degree or diploma at SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at SEECS or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Author Name: Sundas Hamid

Signature: ______________

ACKNOWLEDGEMENTS

First and foremost, I would like to extend my humble gratitude to Almighty Allah who always bestowed His blessings on me and gave me courage to accomplish this task. Darood-o- Salaam to Prophet Muhammad (P.B.U.H) chosen by Almighty Allah to guide the mankind to divine path.

I am truly indebted to my supervisor Dr. Hafiz Farooq for his support, guidance and unending tolerance. He patiently spelled out all new concepts and completely guided me in all technical directions. I own that without the inspiring guidance of Dr. Hafiz Farooq, this research would not have materialized. I extend my appreciation to my co-supervisor Dr. Khalid Latif. Whenever there was a problem he steered me through. I am thankful to all of my committee members, for their guidance throughout the research work of this thesis. I really have no words of thanks for my class fellows and research fellows for their co-operation. I would always cherish the moments spent with them.

I can never forget the contribution of my parents in providing their all out assistance to me. I owe all my achievements to my parents whose assistance and prayers enabled me to surpass all the hurdles in my life.

Sundas Hamid

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Usage of Internet is increasing phenomenally, people like to share their information on internet which would become accessible to each and every one. Use of internet and its applications has become a part of daily life, major uses of internet are: search engine, communication, job search, shopping, hobbies and research. Number of users of internet has reached to two billion [1]. Most of internet's traffic uses HTTP/HTTPS protocol where as the communication platform is World Wide Web (WWW) which is used by the companies to share business information with partners and customers.

The use of the World Wide Web or internet is a crucial part of modern life. We are just a click away from the rest of world and all this is possible due to the presence of the web applications. These applications are diverse in nature and present numerous services to the end users. Commonly provided services include chatting, video conferencing, email, online editing and other custom services. Each application is designed accordingly to the requirements of the service provided and to maximize and improve end user experience.

1.1 Web Application Security

An application is known as web application which can be access over a network i.e internet or an intranet. In today's e-business world, web application security has become the most important issue. The prime concerns of web sites and clients are the security issues. Highly confidential and critical information which is associated with web applications needs protection. Figure 1 shows the statistics [2] of different types of attacks in each type of organization.

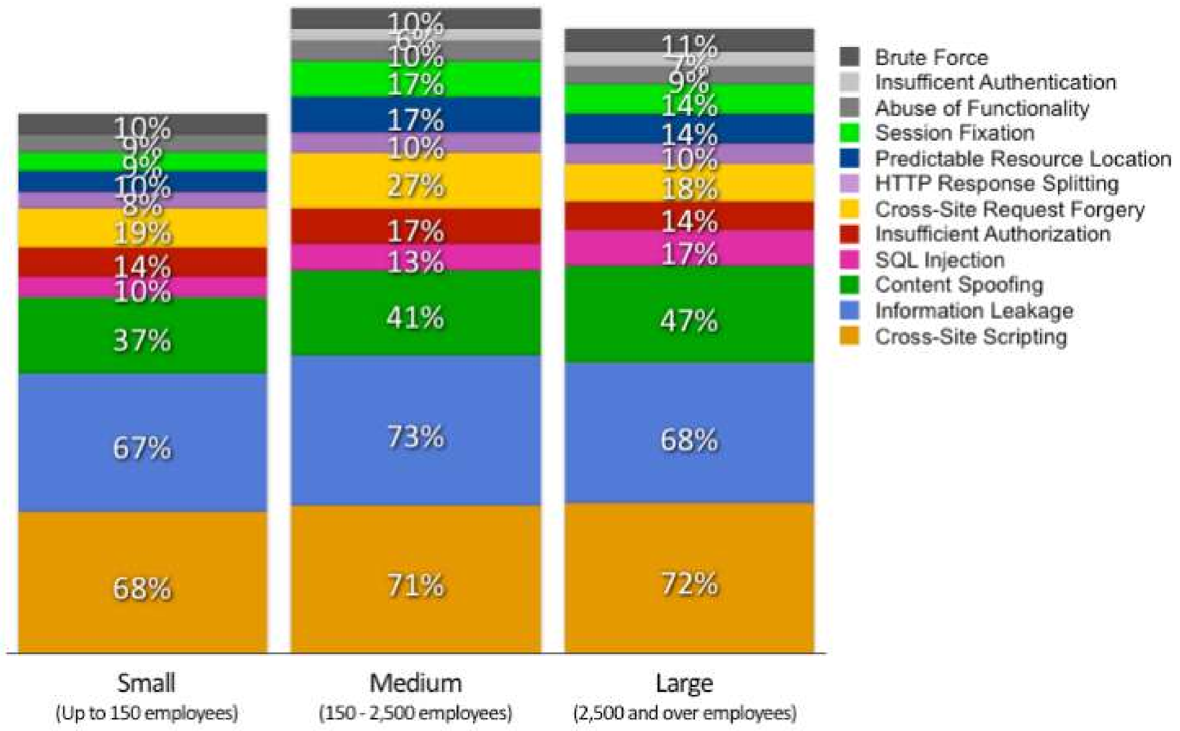

Figure 1: White Hat web site security statistics

Different types of security mechanisms have been deployed such as in the form of intrusion detection system and encryption devices but they have been proved quite insufficient. The system should be intelligent enough in order to mitigate application level attacks as well as system must be able to understand the context of the contents and able to filter that contents on the basis of its consequences onto the target applications. In the world of information, web applications and services have presented a new scope. Web was supposed to be a simple mechanism of document exchange but now it has become an essential part of government, corporate, exchange and educational information arena. Confidential data privacy and shift of critical information exchange needs to be protected. At different levels of OSI model, number of security technologies exists that provides protection against attacks. For the protection against web application, Web Application Firewalls (WAF) proves to be the best solution. Negative security model and positive security model are the two main approached used to build WAF.

SWAF (Semantic based Web Application Firewall) a semantic based solution to detect and prevent attacks against web applications is currently developed using a negative security model which uses semantic based reasoning ability to provide better detection rate [3].

1.2 Motivation

Detection rate is one of the key features of web application firewalls and requires improvement because most of web application firewalls suffer from low detection rate but SWAF does not suffer from above stated problem as it involves semantic based detection and validation techniques, still improvement is required in this area. Many strategies have been adopted to introduce improvement in this field but most fail to deliver the required results. Probing attack is one of the unnoticed aspects which need to be explored because it can help detect malicious activity and take a proactive action [4]. There are different component of SWAF, they all are important but the importance of Probing Detector cannot be denied, it is used to detect the vulnerabilities scanning and malicious activity of the users. Site scanning/probing is the initial phase of any attack on Web applications. During this phase, the attacker gathers information about the structure of the Web application. This can be very helpful for attacker to launch attacks. Probing Detector analyzes the user's requests, behavior and activities, and use this information to help the rule engine to enhance its detection capabilities. After capturing the required information, we need to apply the detection scheme to distinguish between malicious users who are invoking threats and normal users. By applying different probing attack detection techniques we can analyze the user request, their states and then generate the runtime threshold values for some attacks to detect the malicious activity and thus enhance the detection rate of SWAF. Process can be valuable for analyzing user's browsing behaviors on the web application. Once the malicious activity is detected and analyzed the system has precise information of the time and types of attacks which is beneficial in optimizing cost and time utilized in detection. It is also helpful to take an appropriate action like block the users. Figure 2 shows top level vulnerabilities of web applications, 80% attacks in web applications are shown here, root cause of every attack is the leakage of significant information which is the result of probing/scanning.

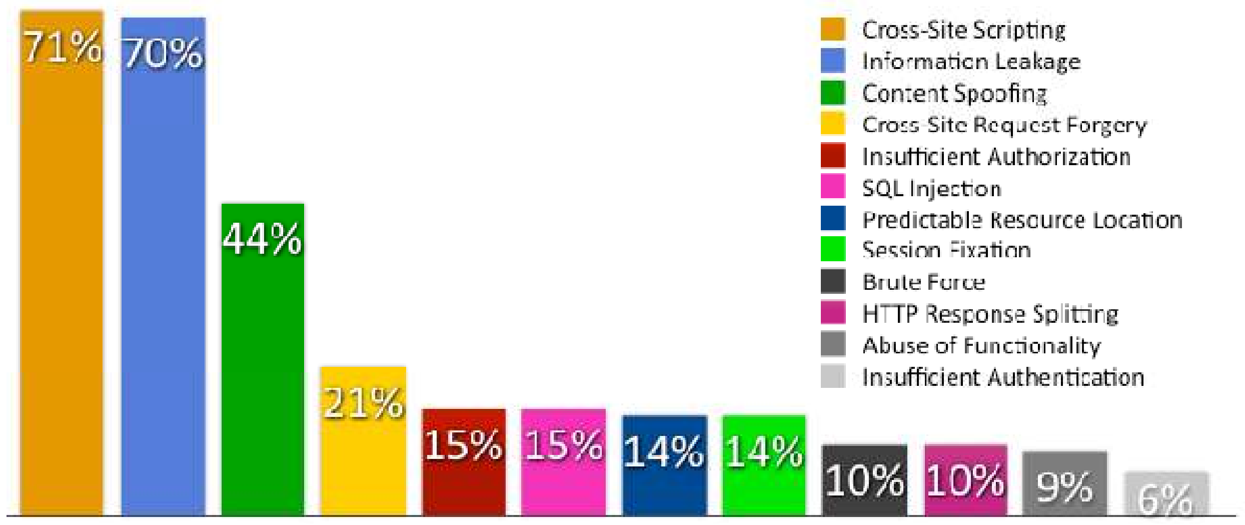

Figure 2: Overall Top Level Vulnerabilities

1.3 Thesis Objectives

The research is intended to obtain the following objectives:

1. To develop a probing detection module for SWAF to enable proactive defense in SWAF.
2. To design the module as an offline process to avoid overwhelming situation as well as ensure minimum resources utilization to overcome the limitations presented in the existing solutions.
3. To provide accurate and effective techniques for web application fortification for high performance rate.
4. To reduce attack surface by successfully blocking the site scanning/probing.

5. To obtain high performance use the log for the identification of probing.

1.4 Thesis Organization

Thesis is organized in the following way. Chapter 2 would present the literature survey. Methodology will be explained in chapter 3, the implementation and evaluation results will be describe in chapter 4 and chapter 5. In the end chapter 6 will presents the conclusion of thesis, future work and achievements.

1.5 Summary

The overview of research work is described in this chapter. It has presented the existing ratios and statistics of attacks in different types of organizations. Motivational factors and objectives have also mentioned here. In addition the explanation of thesis organization is illustrated which provides the detail of chapter of thesis.

Chapter 2

Literature Review

Research's background knowledge is described in this chapter. Web application's vulnerabilities, probing attacks and improved solutions are also discussed here.

2.1 Web Applications Security pressure

Cyber criminals take profit from identity theft, fraud and many other illegal activities, web applications are rapidly growing target for such criminals. On the target web application the impact of an attack is significant which can result in thwarting service interruption, stolen data and low productivity, to cater for from all these threats web applications need to be protected and better protection measures should require. No authentication and week applications are the major targets for unauthorized and malicious use and thus regarded as application layer vulnerability.

Probing is the main and unnoticed issue regarding web applications security, throughout this phase the attacker gathers as much information as possible and credentials about the target web applications structure, these websites are scanned for the purpose of known as well as unknown vulnerabilities. Entire web application can be scanned after the attacker examines the infrastructure. Probing/scanning presents a map of the whole site which includes all the parameters, pages, cookies. By using these credentials the attacker is able to understand the application's authentication, logic and other mechanisms [4].

2.2 Preventive Measures

Different preventive measures have been used to prevent from probing, three main approaches rule based, run time traffic analysis and API based are described in detail below:

2.2.1 Rule Based

Another technique that is used to detect the scanners, is rule based. To detect probing, they used set of rules which apply on the HTTP request to find out whether the request is normal or send via a automated tools. Example of rules is shown below

SecRule REQUEST_HEADER:User-Agent "webscarab" "deny,log,id: 330037, msg: 'webscarab detected'

This rule is used to detect the scanner named as webscarab, webscarab is an open source vulnerability scanner that is used to do probing. It sends webscarab as a string in each User-Agent header. To detect it whenever the User-Agent which is a request header is webscarab this rule fire and the scanner is detected.

ModSecurity, is an open source web application firewall is the example of this approach. The main disadvantage of this approach is whenever a new scanner has come or attackers modify the scanner name, the rules of ModSecurity fails.

2.2.2 Run time traffic Analysis

Run time traffic analysis is used to analyze the incoming traffic and inspect the following things:

- Generating errors using non existing URLs
- Providing long parameter values
- Accessing unauthorized parts of the application
- Adding and removing parameters

Each request is examined here and the state for each IP is maintained to find out the scanner traffic. If an IP is found to be indulged in suspected activity then, it is blocked for some time.

This traffic analysis is also used to detect existing scanners. Imperva is the example of run time traffic analysis tool. With all the above mentioned advantages there exist a few disadvantages, which includes excessive resources requirement because it checks and maintains the status of each user. Application profile is completely ignored by Imperva, there is no check for excessive file uploading. It also misses some checks which are needed for scanner traffic analysis.

Identifying and controlling automated clients [6] describe about the Insufficient Anti-automation which is to automate a process which was intended to be performed manually. Attackers or automated robots can repeatedly attempt to exploit the system, these automated tools have the potential to execute thousand of requests at a time as a result there is a potential loss or performance degradation. Ethically it is forbidden for an automated robot to be able to signup ten thousands new accounts in few minutes.

Here is a attacks technique which prevent the web site from serving normal user activity, this is called as Denial of Service (DoS) attack. Application layer's DoS attack can target each of these web application's components such as: web server, database server as well as authentication server.

Another automated process of trial and error which is to guess username, password as well as credit card number of a person, known as Brute Force attack. A single user name against many passwords is known as a normal brute force attacks where as to use many user names against one password is called as reverse brute force attack.

Another form of automated intellectual property theft is scraping attack. The intention of attacker to scrap a website by creating a valid account on the web application after logging attackers launched an automated tool which is called as robot or bot to extract information. When an attacker steal web based information under subscription to share free of charge, scraping becomes problematic.

Some products can accurately detect anomalous behavior that automatically learn the expected and correct use of a web application; i.e WebDefend firewall can detect automated attack tools. When large number of request originates from a single source in a short time period, then a condition arise which is an excessive access rate. WebDefend monitor each source whether it is a single source IP, a single user or a single session, as well as it also determines that the total number of requests which is generated in a specific time is above than a certain threshold.

Radware's Behavioral Server Cracking Protection [5] presents a tool that is developed to detect and prevent the scanning attacks. Author study different techniques that are currently used for scanning and they identify that the application layer scanning tools are more dangerous. Because they help the attacker to find the exact vulnerabilities that can be easily exploited. They divide the vulnerabilities scanner tools into three categories; these are generic scanner, dedicated scanners and exploitation tools. Exploitation tools are easy to detect because they launch the exact attack that can be detected using the signature. But the other two types of tools are much difficult to detect using the current signature based intrusion detection system. Because they do not launch the exact attack, instead they send the valid request that they used to analyze information that they used to identify the vulnerabilities. This paper shows that the behavioral based identification techniques are good then the signature based techniques, because behavioral based techniques are good to identify the malicious behavior that cannot possible to detect using the signatures. They used statistical and fuzzy techniques for this purpose. They also show the blocking mechanism that reduce the false positive and block the user if it is found as malicious, but one important aspect is that this tool also check the block user again and again, if it is proved after some request that user is not malicious the tool will immediately unblock it, if not then the blocking time will be increase.

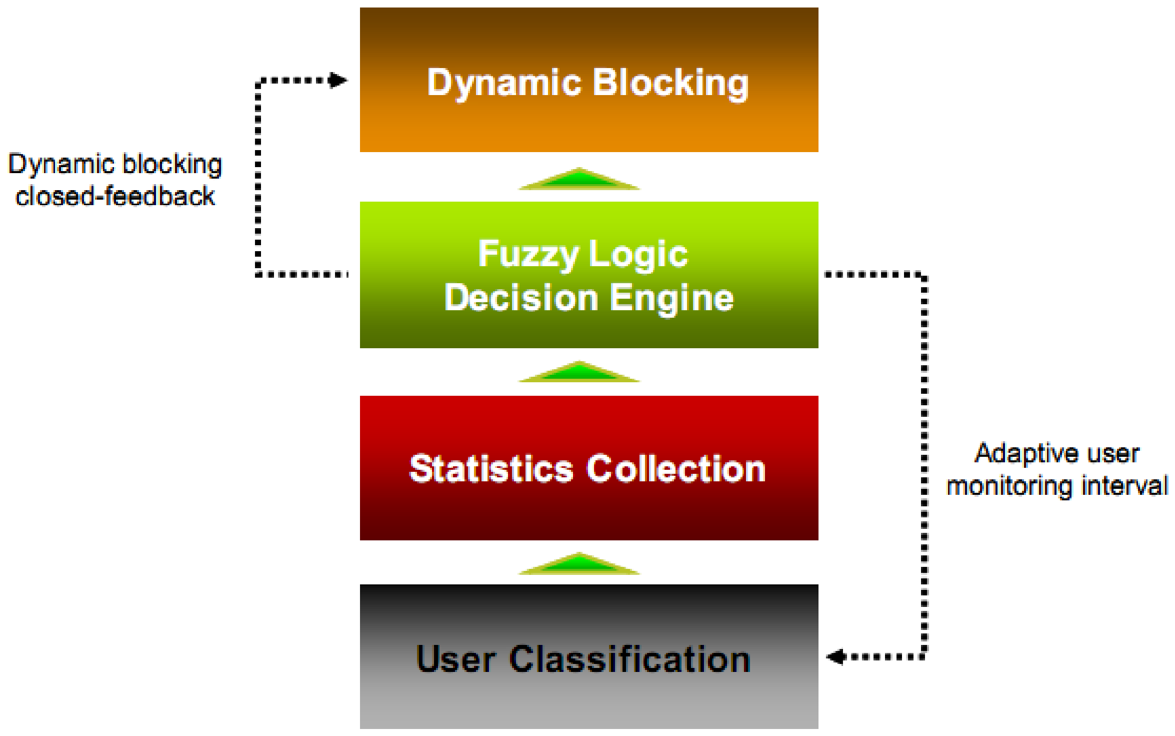
Dynamic Blocking
Dynamic blocking closed-feedback
Fuzzy Logic Decision Engine
Statistics Collection
Adaptive user monitoring interval
User Classification

Figure 3: Blocking Mechanism

This tool used the statistical techniques that can generate more false positive. But this tool can helps in blocking mechanism which is shown in the above figure and help to identify the available scanners that attacker used to launch attack and how to block them or work on them.

2.2.3 API Based

In case of API based technique, Probing Detection Module is integrated with the application. This module is responsible to detect probing on each and every part of the application which is vulnerable. API developed by OWASP name as AppSensor, used inside application for scanner detection as well as application worms detection. It detects scanners successfully because of complete knowledge of the application.

Main disadvantage is to integrate inside application so it becomes application specific. API is developed in JAVA, so currently available only for JAVA web applications.

There are two ways to provide the security to a system, one is reactive: the system will respond after attack has launched and the other approach is proactive: attack is not allowed to be launched [7]. The concept of AppSensor: a conceptual framework is to detect malevolent activity within an application, as well as it is able to identify the probing of attacker for possible vulnerabilities as a result it takes the responsive action urgently.

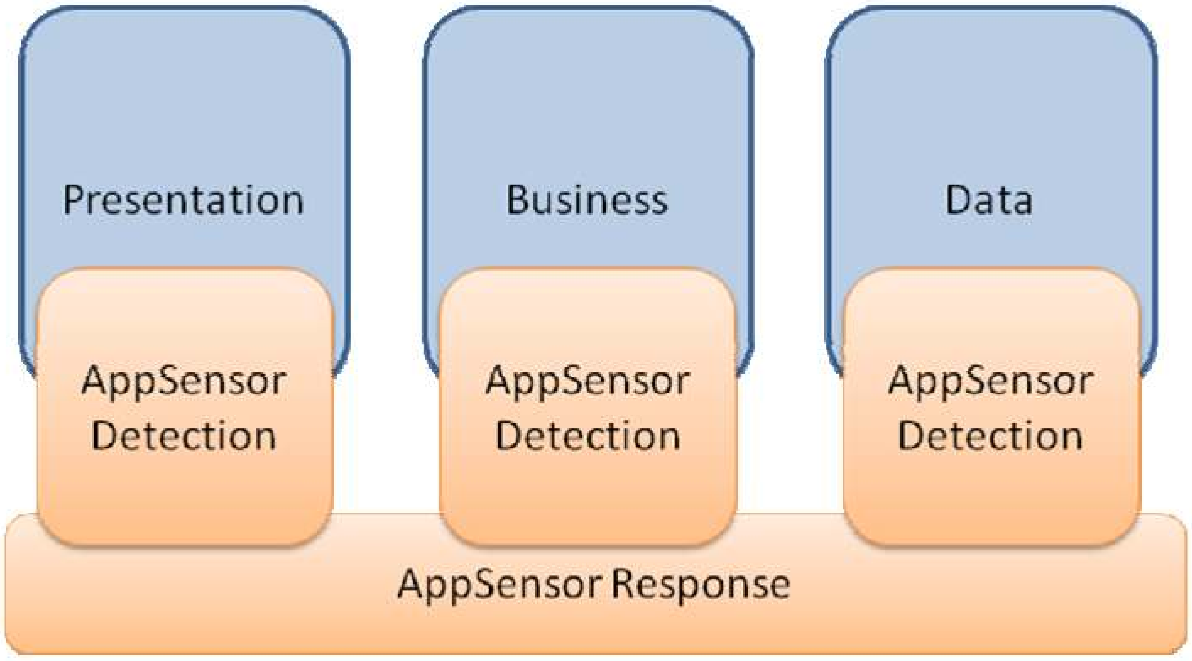

Figure 4: Architecture of AppSensor

Many uncover vulnerabilities are the result of hit and trial by the attacker within the application as shown in above mentioned figure. The concept of probing is as when an attacker tries to scan the web site to find the weaknesses inside the system and then by utilizing those weaknesses launch attacks. Two modules Detection unit and response unit are described in AppSensor, on the basis of defined policies the detection unit is responsible to identify malicious behavior, response unit get the reports activity from detection unit and then it take an appropriate action against the user in case if the action which is taken is an attack or a suspicious event. The system must differentiate while detecting malicious activity, two possible situations are: it is possible that the detected activity may have been caused unintentionally by the user or maybe it is the attacker's activity to hide the attacks attempts. In the first case that type of activity is called as suspect, but in the second case the action is clearly an intentionally a malicious activity in this case it is referred as Attack. For the classification of malicious activity following questions need to consider: whether the mentioned activity results from mistakenly key press by the user? To perform the identified activity does the user have to leave the normal flow of the application? Whether additional software or tools require performing the identified activity? For the justified action against the malicious user, it is necessary to accurately distinguish between suspected and malicious activity.

2.3 Attacker Activities

The two categories of attacker activities identified in this realm are as follows:

1. Malicious Activity
2. Suspicious Activity

2.3.1 Malicious Activity

This type of activity is referred as "Attack", it is clearly obvious that the malicious user is trying to perform an illegal operation on the system [7]. It can be detected using black list rules.

2.3.2 Suspicious Activity

This type of activity is not clearly an attack. May be the detected activity caused by unintentional user error or may be attacker tries to hide his attack attempts. Anyhow it is important not to disregard the objectionable system response [7]. Suspicious activity is difficult to detect. It is used to identify the vulnerabilities. Hybrid solution, blacklist and whitelist failed to detect such type of activities. Currently available scanners and attackers try to use suspicious activities to identify the vulnerabilities. Malicious activity is clearly an attack and can be detected easily using black list rules thus in a result attackers don't use it.

2.4 Summary

This chapter presented the existing solutions and the literature survey. Solutions are critically explained, which conforms that these solutions are insufficient and there is a need to continue this research work and arise with a new solution to conquer these inadequacy and weaknesses.

Chapter 3

Proposed Solution

This chapter describes the proposed solution and the methodology which is taken to implement the solution also presents here.

3.1 Aims and Objectives

The research based solution is set up to obtain following aims and objectives.

1. To provide precise and capable mechanism for web applications protection with high performance.
2. To reduce number of attacks by successfully blocking the site scanning/probing.
3. Usage of SWAF log analysis to identify probing to ensure less resource requirement and utilization.
4. Enable proactive approach in SWAF, to detect attack before it can damage the web application.

3.2 Methodology

Basic methodology used to format proposed solution is presented in this section. There are three different techniques which are used to detect and prevent probing attacks in SWAF. First of all there is the use of xml based detection rules to detect existing scanner which indulges in scanning/probing activity in the target web application. Secondly, SWAF log which comprises of infected log as well as access log is also utilized. Lastly, application profiling is employed which is adopted in the proposed solution.

3.2.1 Detection Rules

These are xml based rules which are used to detect existing scanners which do site scanning/probing. These scanners are used by the attackers to find the vulnerabilities in the target application, thus on the basis of these vulnerabilities, attacker can launch attacks on the target web application. Whenever a request has come from browser, request header User-Agent has the value of browser which contains the name of browser i.e IE, Mozila and Chrome. If scanner is used, it also has this header but its value is scanner name as like DirBuster, Netsparker etc. On the basis of these User-Agent header values the scanner is detected and blocked. The structure of the rule is as follows:

```
<rule>
<rulename>ProbingRule1</rulename>
 <ruledesc>Block DirBuster</ruledesc>
 <application>all</application>
 <location>RequestHeader:User-Agent</location>
 <pattern>DirBuster</pattern>
 <action>deny,log</action>
 </rule>
```

The name of the rule is ProbingRule1, rule description is to block DirBuster, a scanner which is used to scan hidden directories, have text files on the basis of these files it can brute force to find hidden directories i.e admin, pda, administrator. The text file checks each directory one by one whether it exists or not. This rule is valid for all the application, location is on User-Agent a request header, pattern is DirBuster and action is to deny and log it.

3.2.2 Use of SWAF Log

SWAF log consists of infected log and access log.

3.2.2.1 Infected Log

SWAF Infected log is used for probing attacks detection. Different scanners sends several malicious requests, log is analyzed to identify which user did probing on the targeted web application.

Procedure

The procedure is as like: infected log is group by IP, now check for each IP. Check infected log after each 2 minutes as a offline process, if the total number of user requests from that IP is very huge and exceed the threshold limit within the defined time period, set by the administrator of the system, then the IP is busy in doing probing and will fall in the category of suspicious activity, there is a need to block that IP for some time. State will manage for the IP and for other IPs same check will repeat. State includes the information of block IPs. Dynamic blocking concept is achieved as like whenever the same user has come again with probing attack, his/her blocking time will be double.

3.2.2.2 Access Log

Detection of probing attack is also achieved from access log. Infected log not store non malicious requests, whenever a request has come which is non malicious, access log store that type of requests. Behavioral analysis is based on access log. Three different techniques have been used to obtain behavioral analysis in SWAF which comprises on huge number of requests with 404 response status codes, excessive file uploads and excessive suspected activities.

Procedure

Access log is group by IP, check for each IP. First case is greater number of request with 404 response status code. Access log helps to find the suspicious behavior of the attacker i.e DirBuster a scanner which is used to find hidden directories and files. It is analyzed from log that this is suspicious activity of an attacker because it sends a lot of requests with response status code 404, in this case there is a chance of probing by doing behavioral analysis on that specific ip it should block. Information will add into state.

Second case is the larger number of file uploads. If an IP is busy in doing excessive file uploading, this is a suspicious activity and it will fall in the category of probing attack, system have to detect it. Both whitelist and blacklist can't detect such type of activity. There is a chance that the user is trying to launch DOS attack by doing excessive file uploading for the intention of lay down the site performance. The procedure to block such activity is: from the Log the system will check and block that IP. Requests will be count, file upload store as multipart request. The request header "content type" holds multi-part. If these types of requests exceed the threshold limit set by system administrator it is a probing attack, that specific IP must be block, information will add into state. For other IPs same check will repeat.

Third case is large number of suspected activities. Application profiling is used in this case. If there is large number of requests which are violating the application profiling rules then there are suspicious activities. If these suspected activities exceed the threshold limit then it will fall in the category of probing attack, IP will be block and information is added into state. For other IPs same checks will repeat.

3.2.3 Use of Application Profiling

The activity which is ignored by Black List could be suspicious and become an attack in future so there is a need to identify it. Application profiling is necessary to check these activities:

- Request the Resource with the method that it is not supportable
- Sending Request with additional parameters
- Sending Request with less parameters

Request the resource with unsupportable method

If a request has come with the method that is not acceptable then it is suspicious.

Example:

For an application i.e 127.0.0.1 acceptable methods are: GET or POST. If request is as:

HEAD 127.0.0.1:8888/WebGoat/attack

This request is not valid, thus by doing such behavioral analysis it can say that it is a suspicious activity. If there are number of requests which are violating the whitetlist rule and exceed the defined threshold limit then there is a need to block that IP.

Request with additional/less parameters

Quantity of parameters for the application is already mentioned in whitelist rules, but If a request violates this rule it is also suspicious.

Example:

A page Index.php takes 2 parameters, user sends 1 parameter or 3 parameters, so this type of activity is suspicious and can be probing, it should be identify and block.

If user's requests violating these activities then requests will fall in the category of suspected activities thus if these suspected activities exceed the threshold limit then these will fall in the category of probing attack. These activities check from access log, IP will be block and information is added into state.

3.3 Proposed Architecture

The detection of probing attacks is an offline process. Figure 5 shows the proposed architecture for probing attack detection module.

Following is detailed description of the proposed architecture.

3.3.1 Probing Filter

Probing filter is added in SWAF filters. Rule cache includes the detection rules of probing detection module. There are two ways to filter the traffic:

1. First technique is based on detection rules which is stored in rule cache
2. Second technique is based on state which is the input of probing filter

Each request has come in SWAF filters, on the basis of detection rules and state probing filter module will analyze whether it proceed this request to further modules or not, as it is already described that state stores the information of block IPs. The main purpose of this module is to filter that IPs which are busy in doing site scanning/probing. Probing filter based upon behavior analysis so better performance of SWAF is achieved. The working of remaining modules is described in whitelist.

Online request filtering and validation

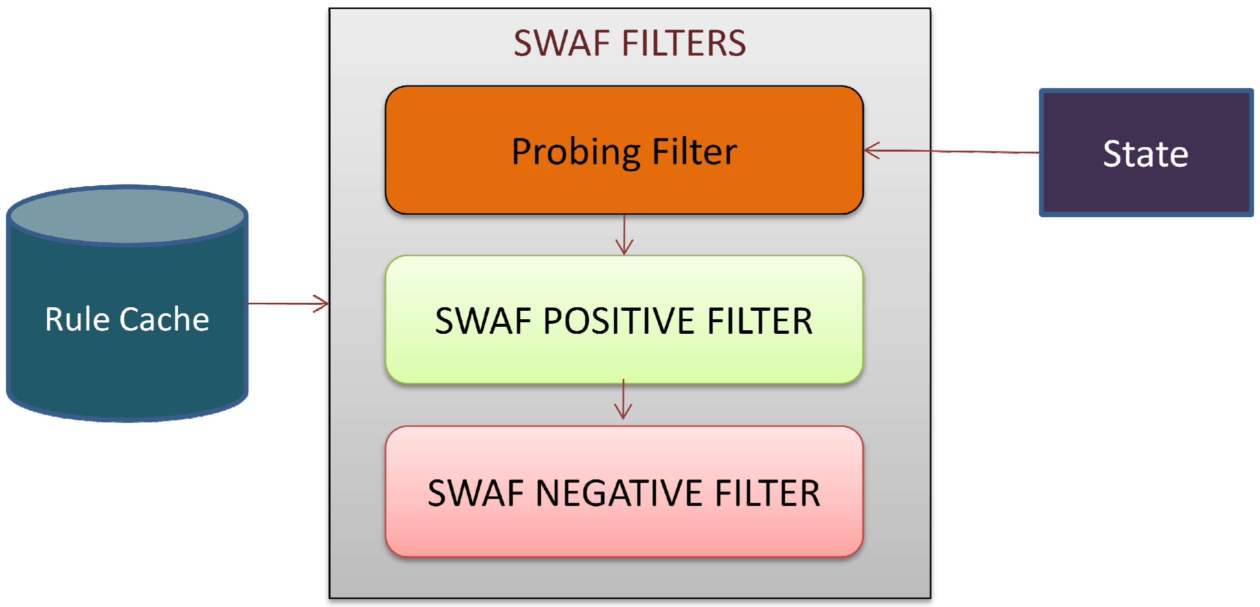

Figure 5: Online Request Filtering and Validation

This is online requests filtering and validation process the remaining working is done as an offline process.

3.4 Proposed architecture as an Offline Process

The above architecture is an online process which filters and validates request online. To avoid overwhelming situation the remaining process is done as an offline process, because by analyzing run time traffic more resources are required as well as it is necessary to use huge memory space to store enormous traffic information thus result in degradation of performance. The offline process is described in the figure below.

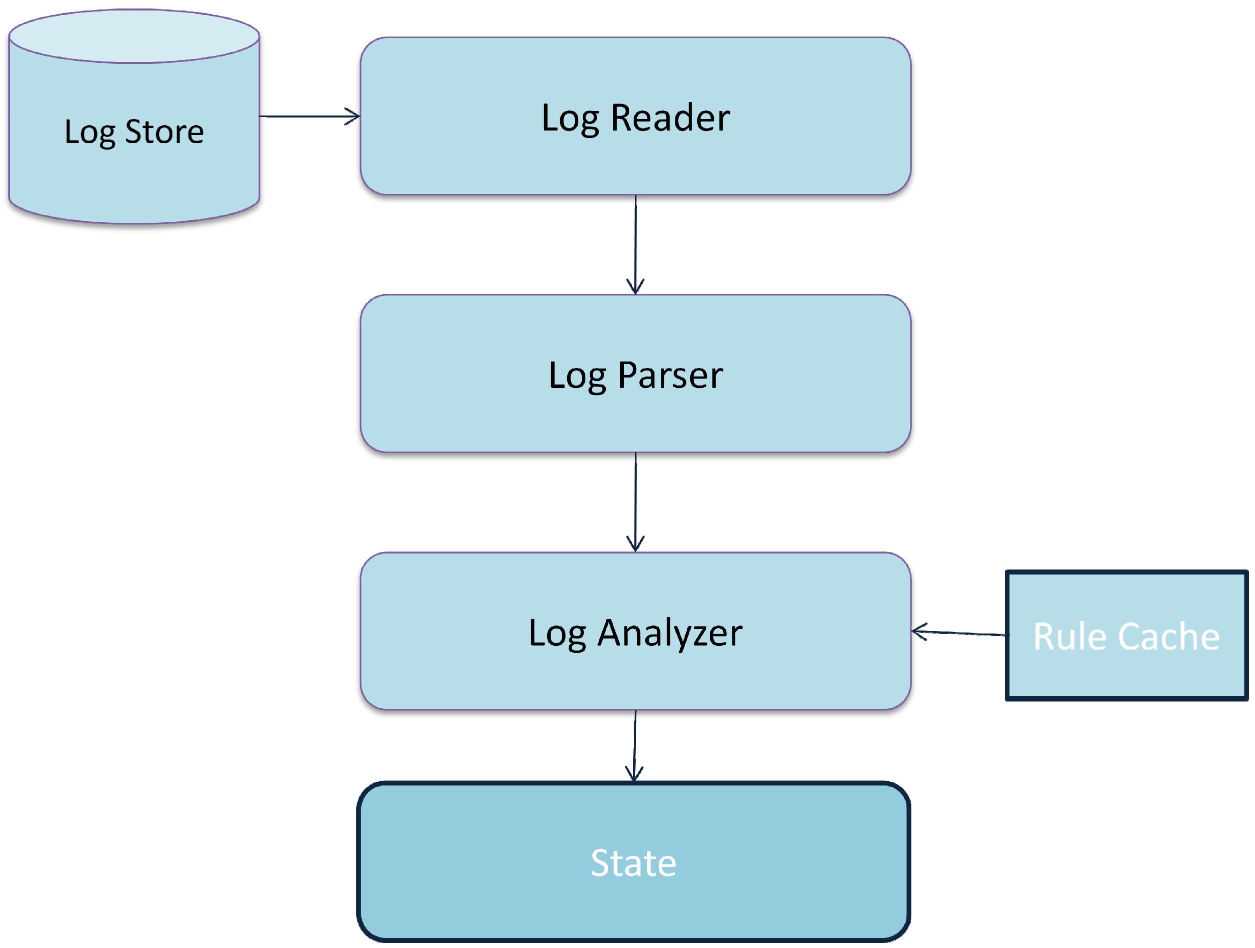

Figure 6: offline process

The description of this process is as: from log store module read the log, then parse it to form the précised and actual request, after that analyze it the input for log analyzer is rule cache which includes whitelist rules, in the end the information is added into state. Log analyzer includes all the processing of probing detection module which is describes above.

3.5 Summary

This chapter illustrated the detailed explanation of the proposed solution. The architecture of the proposed system is also presented. The techniques which are being used in the implementation have also described.

Chapter 4

Implementation of Proposed Solution

4.1 Introduction

The previous chapter described the proposed architecture. This chapter presents the implementation details and the overall architecture of the implemented system. Software which is used for the implementation is also illustrated here.

4.2 System Architecture

All the intercepted requests are placed in SWAF log. Log comprises of two sub categories, the access and the infected log. The existing structure of SWAF is improved to make it proactive. The filtering process is enhanced with the working of probing filter. After addition of probing Positive security system comes under hybrid security solution which overcomes the false positives problems. Negative security system has blacklist rules to detect attacks but it can't detect suspicious activities which become an attack in future, for this purpose the probing detection module will work. Rule cache also includes the detection rules where as the whitelist rules are already present over there, and state includes the information of block IPs, on the bases of these two modules the probing filter will take action. Whenever a request has come to probing filter it will analyze that request by consulting the rule cache as well as state, the validation's first level is to check detection rules in cache and the second check is to see state, after these two checks probing filter take an appropriate action that whether it will forward the request to other detection, the arrangement of SWAF architecture is shown in the figure below.

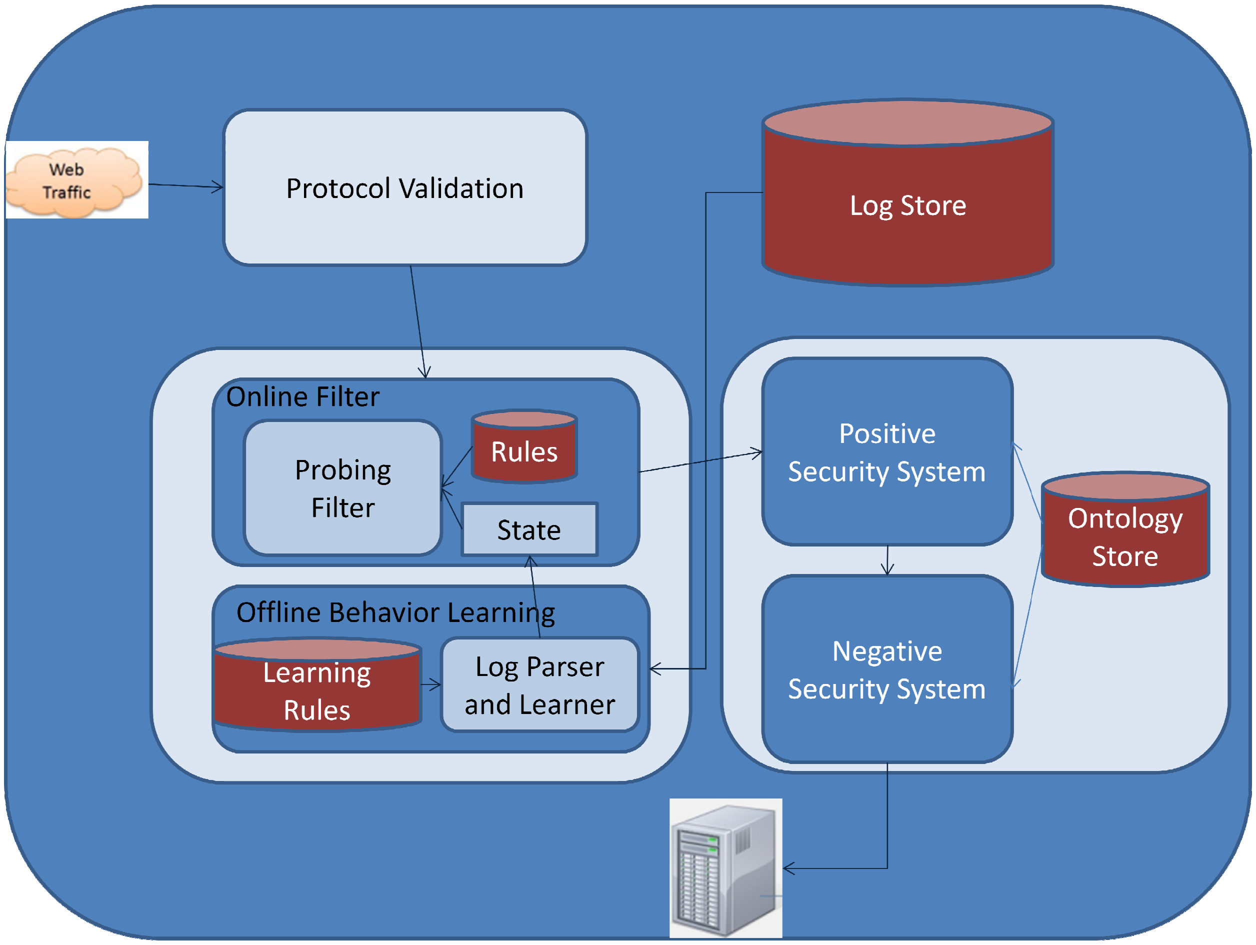

Figure 7: SWAF System Architecture

modules or not. This is the online process. SWAF performance is improved by doing the remaining process offline because there will be no overwhelming situation, and no overhead to analyze live traffic. Offline process includes all the working of Log, from log store all the requests are being parsed and learned after that all the processing has been done under the analyzer that includes the input of learning rules. Learning rules are the whitelist rules which use in application profiling, it is very much helpful to acquire the behavior analysis. The working of this offline process is described in chapter 3.

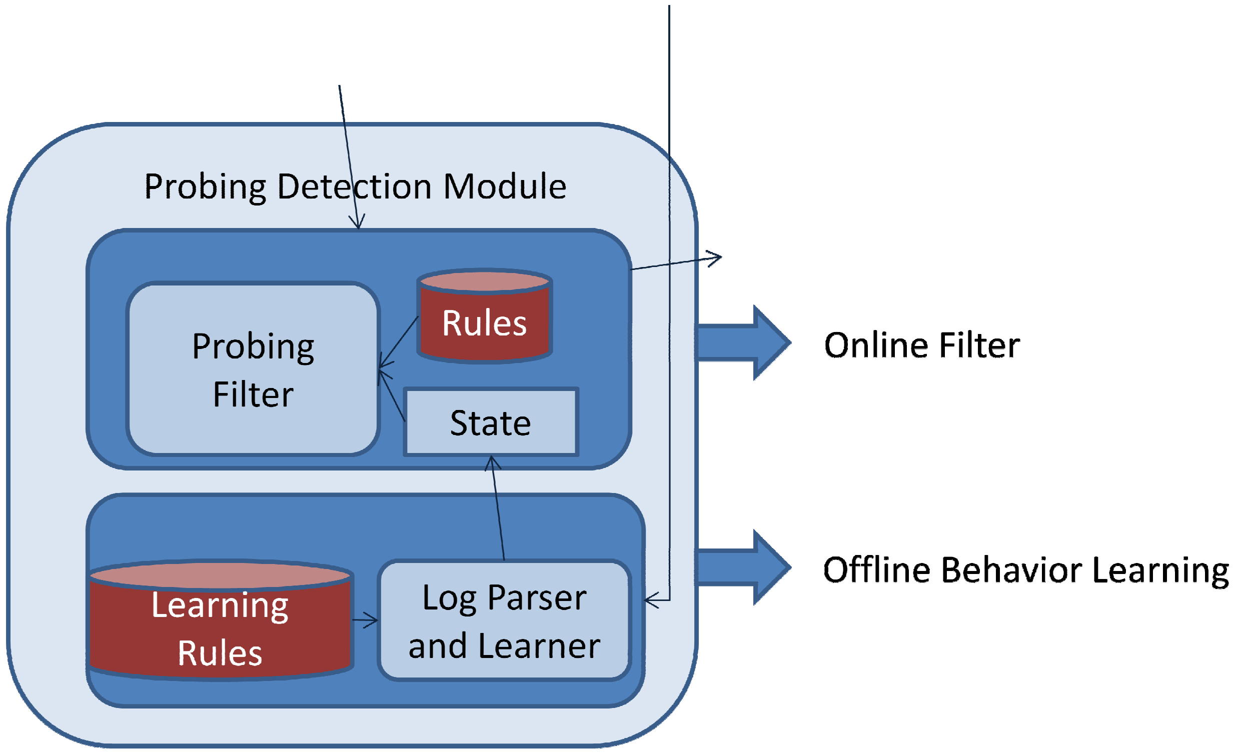

Figure 8: Implemented Probing Detection Module

The above figure shows the implemented approach which proves to be effective and better in performance as the behavior learning process is entirely an offline process which definitely reduces the online processing time.

4.3 Technology Used

The list of software components, database etc helpful for the accomplishment of given system are given below:

- For development IDE Netbeans 6.5 is used. SWAF system using JAVA implementation is already running and this research work is a part of it. Present SWAF system is facilitated by development and integration of given system.
- HTTP response message containing HTML code is parsed by HTML parser 2.0.
- XML based rules using java, user behavior is being learned from log, my SQL stores the log.

- The use of my SQL server 5.1 is to maintain the database of information in SWAF relevant to trace which is passes through SWAF. Proposed system is also using this same database.

4.4 Summary

In this chapter, the implementation of proposed system has been described. We have elaborated the architecture along with details. Furthermore the list of software tools which is used in the implementation is also given.

Chapter 5

Evaluation

5.1 System Evaluation

The system evaluation is carried out using different scanners and vulnerable applications. By the use of scanners, automated tools to test web applications for the security problem such as SQL injections, cross site scripting, insecure configuration, information leakage, directory traversal and remote command execution attacks [8]. Different probing attacks have been launched on the targeted web applications which are used for evaluation, these are vulnerable applications.

Evaluation process is based upon the following points:

1. Use of famous scanners
2. Use of vulnerable web applications

Evaluation steps are described in detail in this chapter.

5.1.1 Use of Famous Scanners

To scan potential vulnerabilities of web applications, tools identified as web application vulnerability scanners have been designed. These scanners automatically test the target web applications for the known vulnerabilities; by using these vulnerabilities attackers can launch different types of attacks on the targeted web application. Mostly scanners user by attacker to launch probing attacks such as Account Lockout Attack, Directory Finder, Forced Browsing, Argument Addition or Removal and Vulnerability Scanning. These tools crawl the web application and establish application layer limitations and vulnerabilities either by inspecting them for suspicious parameters or by changing HTTP messages. These scanners are available in large number both in open source and commercial [8]. Top level scanners which are mentioned in different journals and research papers have been used for evaluation criteria, these are:

- DirBuster
- Acunetix
- Netsparker
- ZAP proxy

5.1.1.1 DirBuster

It is an open source, multithreaded java application which is designed to scan hidden directories. DirBuster is intended to brute forces files and directories [9]. It is helpful for the attacker to learn site structure by scanning its hidden directories and files thus it is highly critical and dangerous.

5.1.1.2 Acunetix

Web vulnerability scanner is commercial based used to scan vulnerabilities like SQLi,XSS etc. For different types of vulnerabilities except SQL injection and cross site scripting, Acunetix web vulnerability scanner automatically check the targeted web application [10]. It has pioneered the web application defense scanning expertise.

5.1.1.3 Netsparker

It is the only free False-positive-free, commercial based web application security scanner which is used to scan vulnerabilities like SQLi, XSS etc. Netsparker automatically discovers flaws in the targeted web application that can leave towards dangerously exposed situation [11].

5.1.1.4 ZAP Proxy

Zed Attack Proxy (ZAP) is an open source scanner which is used to scan vulnerabilities like SQLi, XSS etc. It is a simple to use integrated penetration testing tool for capturing weaknesses in web applications [12]. It automatically crawl the web application for the potential vulnerabilities.

5.1.2 Use of Vulnerable Web Applications

The vulnerable web applications which are used for evaluation are:

1. WackoPicko
2. Acunetix Web

5.1.2.1 WackoPicko

This is the vulnerable web application used for testing, a PHP based technology which contains the vulnerabilities like SQL, XSS etc. it runs on apache web server [13].

5.1.2.2 Acunetix Web

This vulnerable web application is ASP based technology, used by acunetix scanner. It contains the vulnerabilities like SQLi, XSS etc. It runs on IIS web server [14].

5.2 Test bed Setup

Machine used for SWAF

- DELL Core I5, 2.5 GHZ
- RAM 2 GB
- OS: Windows 7

Machine used to Test Applications / Scanners

- DELL Core I5, 2.5 GHZ
- RAM 2 GB
- OS: Windows 7

5.3 Test Cases

There are four test cases which is used for evaluation criteria.

- Evaluation results using ModSecurity
- Evaluation results without SWAF
- Evaluation results using SWAF but no probing module
- Evaluation results using SWAF but with probing module

5.3.1 Evaluation results using ModSecurity

ModSecurity is an open source web application firewall. We have tested it for the mentioned web applications using above mentioned scanners. It contains different vulnerabilities which are shown in the following table.

Table 1: Evaluation with ModSecurity

Scanner	Application	Vulnerabilities	Comments
DirBuster	Wackopicko	5 directories	Access Allowed
DirBuster	Acunetix Web	1 directory	Access Allowed

Acunetix	Wackopicko	26	Access Allowed
Acunetix	Acunetix Web	12	Access Allowed
Netsparker	Wackopicko	27	Access Allowed
Netsparker	Acunetix Web	2	Access Allowed
ZAP Proxy	Wackopicko	7	Access Allowed
ZAP Proxy	Acunetix Web	1	Access Allowed

5.3.2 Evaluation results without SWAF

In this case SWAF is not involved; we have tested the mentioned applications using mentioned scanners for the potential vulnerabilities. The results are:

Table 2: Evaluation without SWAF

Scanner	Application	Vulnerabilities	Comments
DirBuster	Wackopicko	5 directories	Access Allowed
DirBuster	Acunetix Web	1 directory	Access Allowed
Acunetix	Wackopicko	32	Access Allowed

Acunetix	Acunetix Web	20	Access Allowed
Netsparker	Wackopicko	39	Access Allowed
Netsparker	Acunetix Web	13	Access Allowed
ZAP Proxy	Wackopicko	17	Access Allowed
ZAP Proxy	Acunetix Web	4	Access Allowed

5.3.3 Evaluation results using SWAF but no probing module

Here in this test case SWAF is used but the probing detection module is not on. We have tested the mentioned web applications using above described scanners. Results are as following:

Table 3: Evaluation with SWAF (No Probing Module)

Scanner	Application	Vulnerabilities	Comments
DirBuster	Wackopicko	5 directories	Access Allowed
DirBuster	Acunetix Web	1 directory	Access Allowed
Acunetix	Wackopicko	29	Access Allowed
Acunetix	Acunetix Web	16	Access Allowed

Netsparker	Wackopicko	28	Access Allowed
Netsparker	Acunetix Web	3	Access Allowed
ZAP Proxy	Wackopicko	9	Access Allowed
ZAP Proxy	Acunetix Web	1	Access Allowed

5.3.4 Evaluation results using SWAF but with probing module

This is the most important test case, here the working of the probing detection modules have been analyzed. Result shows the decrease number of vulnerabilities as compared to the previous test cases. These vulnerabilities have been shown because probing detection module analyses the user behavior, so in this step some vulnerabilities are there but the access is not allowed to use such weaknesses. Attacker can't use these vulnerabilities to launch the attack on the targeted web applications. After analyzing different user's behavior probing detection module take an action and start blocking the malicious users. Results which are achieved are as follows:

Table 4: Evaluation with SWAF (With Probing Module)

Scanner	Application	Vulnerabilities	Comments
DirBuster	Wackopicko	5 directories	Access Denied
DirBuster	Acunetix Web	1 directory	Access Denied

Acunetix	Wackopicko	18	Access Denied
Acunetix	Acunetix Web	11	Access Denied
Netsparker	Wackopicko	21	Access Denied
Netsparker	Acunetix Web	2	Access Denied
ZAP Proxy	Wackopicko	2	Access Denied
ZAP Proxy	Acunetix Web	0	Access Denied

5.4 Comparison

Different comparison scenarios have been described below. Comparison of ModSecurity and SWAF (with probing module) using wackopicko application. The comparison is shown using the following chart.

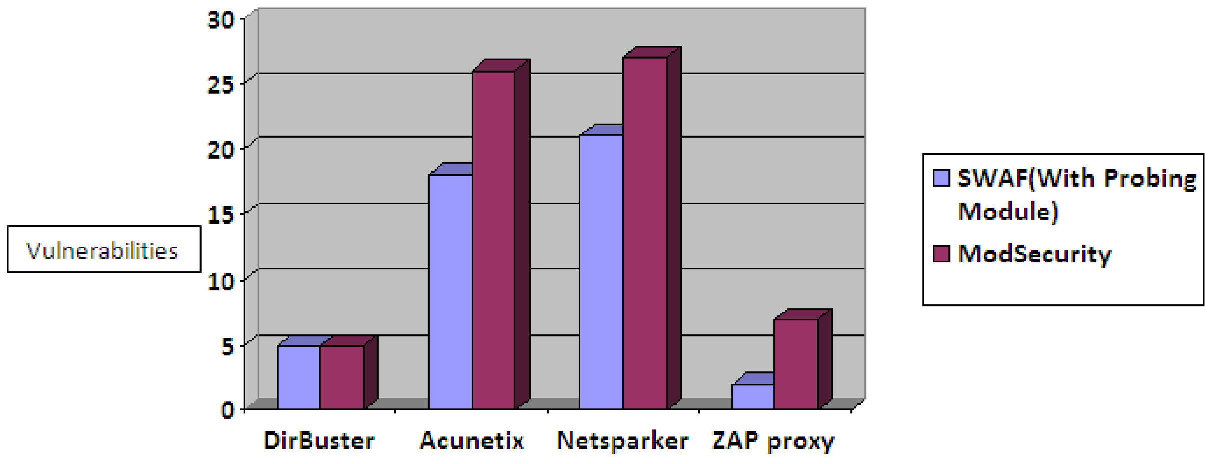

Figure 9: ModSecurity vs SWAF (With Probing Module) for Wackopicko

Comparison using acunetix web application is shown in the following chart.

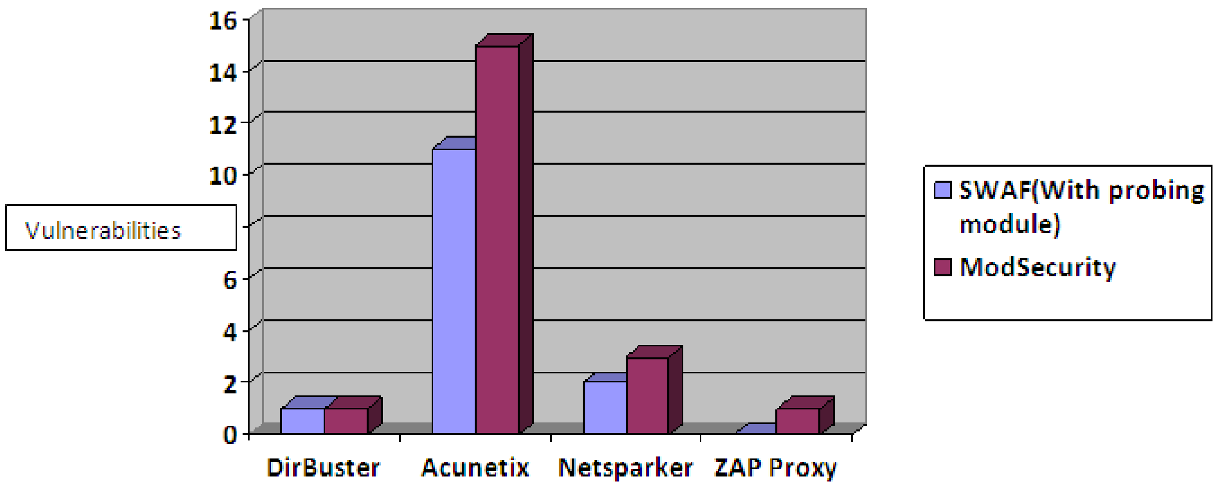

Figure 10: ModSecurity vs SWAF (With Probing Module) for acunetix web application

5.5 Summary

This chapter has illustrated the evaluation process and the results which are taken to access the performance of the system. The evaluation criteria is based upon two things, first is the use of scanners and the second is the use of vulnerable web applications. Evaluation results show the better performance of the proposed module.

Chapter 6

Conclusion and Future Work

Research contribution is enlisted in this chapter through conclusion of thesis work. Conceptual research work along with tasks supposed to helpful in future are also defined.

6.1 Research Contribution

These points are the main contribution drawn from this research.

- The research gives a detailed analysis of user behavior by using application profiling as well as the scanner traffic analysis is also done by application profiling.
- Scanner detection is done by using firewall log, because regular users and scanners are differentiated by log.
- Working in offline mode and scanner detection is done by log traffic analyzing so there is no overhead.
- Scanner behavior is successfully recognize by the use of following:
    - Log
        - Access Log
        - Infected Log
    - Application Profiling
- The use of XML based rules to detect existing scanners
- Traffic analysis check

6.2 Conclusion from Research Work

The need of the era is to secure the web applications. Proactive based approach to secure web applications is important. Existing solutions mainly based on the reactive approach which has its own advantages and disadvantages. The mentioned research work provides good defense against probing attacks which is the key reason of many attacks on web applications. This is the proactive based solution which is the need of the hour. SWAF integrated with this module will become proactive and can detect attacks before it can damage. The above mentioned approach enhances the overall protection and performance of SWAF. The limitations of the existing solutions have been overcome by the use of proposed solution. The intention of the proposed solution is to present an efficient and effective way to alleviate attacks and provides a better proactive based security.

6.3 Future Work

The system can be further enhances by:

- Automatically generate the rules which are statically implemented.
- Optimize the proposed module for live traffic so that its performance is more improved.

The abovementioned tasks are taken as future work.

References

[1]. http://www.physorg.com/news/2011-01-internet-users-worldwide-billion.html

[2] White hat website security statistics report [http://www.whitehatsec.com/home/resource/stats.html]

[3]. Semantic based Web Application Firewall [http://swaf.seecs.nust.edu.pk]

[4]. Imperva,Scannig/Site Probing [http://www.imperva.com/resources/glossary/site_probing.html]

[5] Radware's Behavioral Server Cracking Protection By Renaud Bidou

[6] Identifying and controlling automated clients by Ryan Barnett

[7] Detect and respond to attacks from within the application by OWASP, 2009

[8] Web Application Security Scanner Evaluation Criteria [http://projects.webappsec.org/w/page/13246986/Web%20Application%20Security%20Scanner%20Evaluation%20Criteria]

[9] DirBuster [http://sourceforge.net/projects/dirbuster/]

[10] Acunetix [http://www.acunetix.com/]

[11] Netsparker [http://www.mavitunasecurity.com/netsparker/]

[12] OWASP Zed Attack Proxy Project [https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project]

[13] wackopicko [http://www.darknet.org.uk/2010/12/wackopicko-vulnerable-website-for-learning-security-tool-evaluation/]

[14] Acunetix web [http://testasp.vulnweb.com]

[15] OWASP Account Lockout [https://www.owasp.org/index.php/Account_lockout_attack]

[16] OWASP Forced Browsing [https://www.owasp.org/index.php/Forced_browsing]

[17] OWASP SQLi [https://www.owasp.org/index.php/SQL_Injection]

[18] OWASP Session Prediction [https://www.owasp.org/index.php/Session_Prediction]

[19] OWASP Cross Site Tracing [https://www.owasp.org/index.php/Cross_Site_Tracing]

[20] T. Alexenko, M. Jenne, S. Deb Roy and W.Zeng,"Cross-Site Request Forgery: Attack and Defense", IEEE CCNC (2010)

[21] Directory Traversal [http://www.acunetix.com/websitesecurity/directory-traversal.htm]

[22] Vulnerability Scanning [http://netsecurity.about.com/cs/hackertools/a/aa030404.htm]

[23] OWASP XPATH injection[https://www.owasp.org/index.php/XPATH_Injection]

[24] OWASP Cross Site Scripting[https://www.owasp.org/index.php/Cross-site_Scripting]

[25] OWASP Denial of Service[https://www.owasp.org/index.php/Denial_of_Service]