APPROVAL SHEET

Title of Thesis: Web Application Vulnerability Assessment Tools Analysis

Name of Candidate: Ajinkya Wakhale
Master of Science, 2018

Thesis and Abstract Approved: [illegible]
Dr. Charles Nicholas
Professor,
Department of Computer Science and
Electrical Engineering

Date Approved: July 24, 2018

ABSTRACT

Title of thesis:	WEB APPLICATION VLUNERABILITY ASSESSMENT TOOLS ANALYSIS
	AJINKYA WAKHALE, MASTER OF SCIENCE, 2018
Dissertation directed by:	DR. CHARLES NICHOLAS PROFESSOR,
	DEPARTMENT OF COMPUTER SCIENCE, AND ELECTRICAL ENGINEERING

In this era, with plethora of web applications and increasing amount of consumers using web applications for different purposes, it becomes very important to protect them from several web vulnerabilities present on the INTERNET. Web applications process large amount of data which they store it in a back-end database server where confidential data like username, password, credit-card information sits. Web applications usually interacts with customers and there is huge dependencies between customers and the server and this dependency introduces huge security holes which can be exploited by a hacker to steal the data [16].

The most common way to find vulnerability in the web application is to perform Vulnerability Assessment and Penetration testing (VAPT) on web application. According to OWASP [16],the most efficient way of securing web application is to manual code review. The drawback of doing manual review is that it requires expert skills and it is very time consuming and hence enterprises uses automated tools

to scan the systems and find vulnerabilities in them. Web application scanners are automated tools that scans the web application to detect unknown vulnerabilities in the application. This technique is usually referred as Dynamic Application Security Testing.There are several tools available in the market that does security testing on web applications and gives you detailed report on all the security loopholes present in the system [16]. It requires deep insight and understanding to deal with web application security not because of the many tools that are available, but because it is still in nascent stage. Hence, it becomes really important to find proper tools to scan the web applications and find vulnerabilities present in the system.

Most tools available in the market, both open source and paid commercial, confines themselves to the specific set of vulnerabilities in which they are expert.For example, some tools are best designed to find SQL injection in the system while some are good in finding cross-scripting or CSRF. Hence, it becomes important to find the right tools which takes into the consideration of development environment, needs and most importantly web application complexity.

This research propose a detailed report on some of the most commonly used tools in the market and their efficiency in finding out the vulnerabilities in the web application and the technique they used to find out the security loopholes present in the application. We discuss several efficient tools along with their advantages and disadvantages, techniques they use and most importantly, their efficiency to detect vulnerabilities in the application. It evaluates all the tools and give recommendation to the developer and user of the web application. It also analyzes whether the development and hosting environment of the application affects its security or not.

WEB APPLICATION VULNERABILITY ASSESSMENT TOOLS ANALYSIS

by

AJINKYA WAKHALE

Master's Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County in partial fulfillment
of the requirements for the degree of
Master of Science
2018

Advisor:
Professor Dr. Charles Nicholas

ACKNOWLEDGEMENTS

First and foremost I'd like to thank my advisor, Professor Charles Nicholas for giving me an invaluable opportunity to work with him for this research project He has always made himself available for help and advice whenever needed. It has been a pleasure to work with and learn from such an extraordinary individual.

I would also like to thank all my friends and people whom I met at UMBC for their encouragement and help whenever needed. Finally, to my parents and sister and all the family members for motivating me to pursue my dream of doing masters. I owe it all to them. It is impossible to remember all, and I apologize to those I've inadvertently left out.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Web Application

Web Application is a simple computer program that performs some function and uses web browser as a client. Customer uses web browser to access web application. Web browser in turn communicates with server to access the database which sits at the back-end and the server response the data back to the browser[13].

1.2 Client

In a client server environment, a client is referred to a program that the person uses to run the application. There could be multiple clients accessing the database at the same time and they use server to communicate with the database. It is a piece of program that provides a facility to the customer to use the services provided by the server[25].

1.3 Server

In web applications, there are two sides: One is client side and another one is server side. Both, client side and server side has programs that runs on them concurrently. Main purpose of a web server is to parse the requests from client-side

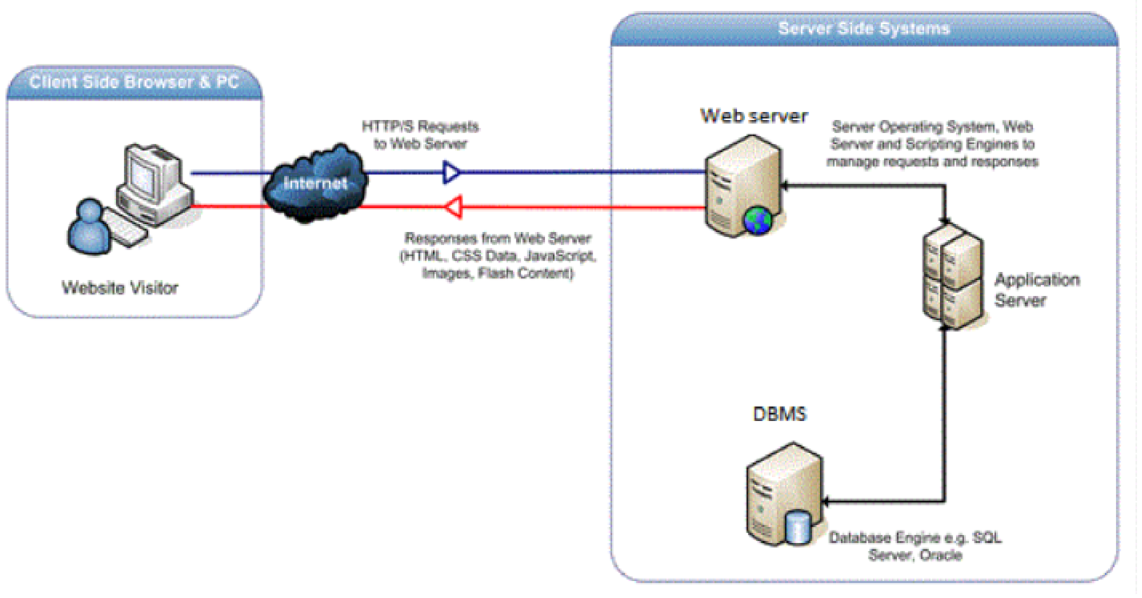

Client Side Browser & PC
Website Visitor
Internet
HTTP/S Requests to Web Server
Responses from Web Server (HTML, CSS Data, JavaScript, Images, Flash Content)
Server Side Systems
Web server
Server Operating System, Web Server and Scripting Engines to manage requests and responses
Application Server
DBMS
Database Engine e.g. SQL Server, Oracle

Figure 1.1: Web application Communication

and give appropriate response to the client. The requests follow HTTP protocol and are hidden from the common user of the application. Web server is responsible to store, process and deliver the web pages to the client[27].

Application server is the server which hosts the code at server side and exposes the business logic and processes to the client. Web server is responsible for delivering and showing the web pages to the client whereas application server is responsible for the logic and is responsible for the interaction with the system [22].

The figure 1.1 shows how a client interacts with web browser and application server to request the data from the database. Client first sends HTTP requests to the web server which in turn communicates with application server. Application server interacts with the database and gets the required data and sends it back to the web server. And then finally, the response is send to the client in the form of HTTP response [3].

Application server is the place where developer writes the business logic of the application. The main important attribute which server code shows are that it is hidden from the user. User agent or the user who is accessing the web application is not concerned with the intricacies of the business logic on the server.

1.4 Application Architecture

Software architecture of a program or a application depicts in understanding the behavior of the system. It serves as a blueprint for the system defining the work assignments that has to be carried out by each components. The architecture is primarily responsible for the qualities that system depicts like performance, re usability and the most important, Security. If the architecture of the system is sound and effective, it mitigates the risk early in the development process [12].

1.4.1 Web Application Architecture

To develop a robust and efficient web application, the developer needs to understand the intricacies of World Wide Web(WWW) and has to take into the account all the features as well as the problems associated with it. The main aim of the WWW project is to make arrangements to setup the information space so that people and machine can communicate [7]. As customers using web applications can be located around the world using different platforms and resources to access them. Also the data and content moving on the network differs in the format and types. Hence the major task for the people working on WWW project was to develop a system or a platform that can provide a universal standard to the information moving on the network. And the challenge for them was to minimize the interactions with the network [7]. Hence the biggest daunting task for a developer at the time when he starts to explore the best architecture pattern for his application is to select best type and the component model of the web application.

1.5 Web Application

1.5.1 Web Application Benefits

A web application alleviate the developer concern of building applications for specific platform, so that anyone can use them as long as they have Internet connection. Since the client runs in the web browser, user can have any operating system installed on his machine. Generally,it can be accessed through any web browser like Chrome or Internet Explorer, though some applications require specific browser [13]. It generally uses combination of server side script like (ASP, PHP etc) and client side script like JavaScript and HTML to develop the whole application. The client side is responsible for presenting the application to the end user while server side deals with all the business logic of the application. Server side is also responsible for accessing the database. According to Redrock software [1], some of the benefits of using web application are :

- No more updating issues
- Less resources means less money

- Platform Independent
- Quick Development Cycles
- Improved Security
- Flexibility of the Internet
- User Tracking

1.5.2 Web Application Security Issues

Several different and interacting technologies are used to develop a web application. Due to this reason, web application poses lots of security issues. With the popularity of INTERNET growing, many customers are going online for their daily work and uses several web applications. With these, lots of customer personal data like account number, credit card numbers are going online and hence they become vulnerable to fraud and other attacks [4].

1.5.3 What makes Web application so vulnerable?

Web applications can be complex programs which provides a service to the user to create, delete and modify the data. Generally, web applications are considered to be a script running on server side and the security risks and issues are only related to the scripts running on the server. However, design of the architecture also impacts security issues of web application [18].

- **Designed without Security Considerations:** Most often developers, managers and Software Engineers who are responsible for developing the web application do not take into the account a maximum number of security considerations. If proper security procedures and methods are not followed to at the time of designing the architecture, several security issues may be missed and it becomes very hard to fix the problem later.

- **Dependency on other Architecture Components:** Web application uses several interacting technologies to provide the service to the user. To do that, it acts as a gateway between client and the back-end server. Due to the reliance on the other components in the architecture, it becomes a requirement for a web application to assign some security measures to the other components. Since, these components may not always complement each other, separate interfaces needs to be developed to develop the communication channel for the components to interact with each other, leaving possibilities of many security loopholes[17].

- **Plethora of Programming Languages:** Due to abundance of the programming languages to develop the web application, it becomes difficult to design and define common security framework across the infrastructure [17].

1.5.4 Security Problems in Web Application

According to OWASP [15], the top ten security issues related to the web applications in 2017 are :

1. **Injection:** The input which is provided by the user is sent to the back-end for processing. If the input is not validated, they may contain special characters which has a special meaning[4]. Injection flaws occurs when an attacker can send hostile data to the interpreter. Injection flaws are very prevalent, especially in legacy systems. An attacker can find injection vulnerabilities in SQL, LDAP, XPath, OS commands, SMTP headers and ORM Queries.[15] Injection can result in the data loss, loss of confidential data and it can sometimes lead to complete host takeover.

2. **Broken Authentication:** For credentials stuffing, an attacker has access to hundreds of combinatorial username and passwords. It can also use them for finding administrative accounts and dictionary attack tools. These also leads to attacks in session management tools. These kinds of attacks are also very prevalent due to the design and implementation of identity and access controls [15]. Attackers uses manual approach to find broken authentication issues and then use automated tools to exploit them. With broken authentication, attacker can gain access to the few accounts and then can compromise the whole system using identity theft or disclose sensitive information.

3. **Sensitive Data Exposure:** To exploit sensitive data exposure vulnerability, a manual attack is required. Attacker steal keys, executes man-in-the-middle-attacks to steal the sensitive information. According to the OWASP [15], this has been the most impactful attack. An attacker can exploit weak encryption techniques used by the application to decrypt the sensitive information. Sen-

sitive data exposure leads to leak in sensitive and personal records of the user like health records, credit card numbers etc.

4. **XML External Entities:** In these kind of attacks, attacker can exploit weak XML processors by including hostile content in XML resulting in exploiting the vulnerable code, dependencies and integration[15]. SAST tools can discover the issue where XML processors allow the attacker to specify external entity which is a URI to that is evaluated during XML processing. These can be used to execute DOS attack and can be used to extract the data.

5. **Broken Access Control:** By using SAST and DAST tools, an attacker can find the absence of access control. Due to lack of automated detection and lack of effective functional testing,access control mechanism becomes weak. According to OWASP [15], the technical impact of this attack is that an attacker can acts as a authorized user or an administrator having privileged access and can create, delete or modify the data.

6. **Security Misconfiguration:** According to OWASP [15], attackers can attempt to exploit unpatched flaws or access default accounts, unused pages, unprotected files and dictionaries to gain unauthorized access to the system. These kind of attacks can happen at any level of the application stack. These kind of attacks can often lead to complete system compromise.

7. **Cross-site Scripting(XSS):** It is a kind of attack, in which an attacker can hijack user-sessions, redirect to malicious sites. A user can write content

into the HTML file through manipulation of input variables. After that, the attacker can trick the user of the web application to think that the content is real. More importantly, the attacker can craft a JavaScript cross site Scripting attack and can steal user cookie to launch session hijacking.

8. **Insecure De-serialization:** De-serialization attack is kind of difficult attack in which exploits rarely work but it does not change underlying code. Application is vulnerable to De-serialization attack if they deserialize hostile content provided by the attacker.According to OWASP [15], insecure De-serialization attack can result in two types of attacks:

   (a) Data structure attacks in which attacker modifies application logic or get access to remote method execution.

   (b) Access-controlled attacks in which content is changed.

   Hence, these kind of attacks can not be understated as it can lead to remote code execution.

9. **Using Components with Known Vulnerabilities:** These kind of attacks are very widespread as developer might be using many components without even understanding their security issues and loopholes. Some of the largest breaches to date have relied on exploiting known vulnerabilities in the components.

10. **Insufficient Logging and Monitoring:** According to OWASP [15], exploitation of insufficient logging and monitoring, is the bedrock of every ex-

ploitation. Due to lack of monitoring, attacker time their attack so that they are not caught. OWASP report for 2017 [15], in 2016, identifying a breach took an average of 191 days which is a plenty of time to cause damage to the application [19].

1.6 Web application Security

The process of building a secure web application need one or more guidelines and procedures which needs to be followed to make it a secure system. Gritzales and Spinelis [9] provide best practices for addressing security threats and issues in web application. There are very comprehensive guidelines available to make the web application more secure and it includes system, network and software security. According to [21], the best way to secure web application is to use penetration testing tools and the knowledge of good software practice right from the design phase which helps in developing a robust and secure application. Hence, the need for the requirements related to the security issues needs to be identified and included in the design phase of the application.

1.6.1 Basic Guidelines for Securing Web application

By using well-defined security related processes, developer can guard their applications against common threats like SQL injection and XSS. Applying basic guidelines specific to security, helps you to achieve maximum security and lower costs[21]. All of the developments in the web application have also attracted hackers who tries to exploit web application to access unauthorized data or to deny the service to the user. This has given birth to a new and young industry called Web Security [2].

1.6.2 Network Firewall

The biggest myth for the developer developing a web application is that the network firewall will protect the web applications and will not allow the malicious user to attack the system. Network security differs from web application security. Network security defenses such as firewalls prevents malicious users. As web applications are on-line all the time, it has to allow user to access the system and then hopes that the user will play by the rules [2]. Hence, analyzing all the traffic coming to the web server is not possible and vulnerabilities like SQL injection and XSS are difficult to catch. A Web application firewall analyzes all the HTTP and HTTPS traffic and can identify malicious users. It can block the user who is trying to exploit the vulnerability in the application. The problem with using Web firewall is that it detects only known vulnerabilities and as it is a user configurable software, therefore if it is not configured properly, it does not fully protect the web application [2]. It does not identify the loopholes present in the system.

1.6.3 Securing Web application

In order to develop a secure web application, it is required to identify all the possible vulnerabilities present in the system before the attacker tries to exploit them. Hence, it becomes very important to test the application at every stage of SDLC. Multiple approaches could be used to find out the loopholes in the system. One is to audit source code manually and other is to use automated tools. Both approach has their pros and cons and hence it becomes prerogative for the software managers to decide which approach works better for them. Different applications require different approaches to test the system. Hence, one has to chose efficient and effective solution that can realistically find bugs and threats present in the system. Generally, organization uses combination of both manual and automated tools.

1.7 Web Application Vulnerability Assessment Tools

Web application Vulnerability scanners are the tools that automatically scan the web application and detects vulnerability present in the system. In recent days, web application scanners became very popular as they automate lots of process and are very interactive in their use [2].

Several open source and paid web application scanners are available in the market. All the tools have their advantages and disadvantages. Some tools are very easy to use and are interactive and give proper reports and others are CLI based but are more efficient in finding vulnerabilities present in the system. Some tools are best designed to find SQL injection loophole, whereas some are expert in finding XSS and broken authentication attack.

In this research, we aim to analyze some of the most popular tools available in the market for web application scanning and give a detailed report on each tool comprising several parameters like usability, detecting vulnerability, reports, resources used and how efficient they are in finding vulnerabilities in the system. Also, we provide a detailed report on how the hosting and development environment of the application affects its security.

The remaining document is organized as follows: Chapter 2 describes the background and related work. It describes some of the popular tools available in the market and it gives information about some of the research that has already been achieved in this area. Chapter 3 describes the methodology which we used to test the tools and different applications. It describes the testing environment which we used to test the application. Chapter 4 summarizes our results and we conclude in Chapter 5.

Chapter 2

Related Work and Background

2.1 Background

2.1.1 Web Application Scanner

To prevent the scenario of a attack, developers are always encouraged to follow proper coding practices, review security measures and audit the application at every stage. However, as web applications are centered around users interaction with the system and satisfying user's requirements, developers often misses security aspects.Web vulnerability scanners are the most easy way to detect vulnerability in the system. There are many commercial and web application scanners available in the market which are used by enterprises to detect the security flaws in their application [8].

These tools normally include three stages.

- Configuration
- Crawling
- Scanning

Configuration involves defining the URL and setting up the parameters. In crawling stage, the scanner generates a map of the internal structure of the application. This stage is the most important stage in scanning a web application, as failing to discover a page will prevent their testing [8]. Scanner finds the first web page and then look for links. Each link is then requested and the same procedure is repeated until no more links or pages can be found.

In scanning stage, automated penetration testing is performed against the web application by simulating browser [8]. Thousand of test cases are executed and are analyzed to detect vulnerabilities in the application. Malicious requests are send to the server to check the response and error codes. Then these responses are validated and analyzed by putting a benchmark. Results are shown to the user which can be stored for more analysis. Some tools are GUI based whereas some are CLI tools. Web application scanners have predefined set of test rules which they use to find the loopholes in the system. They also have collection of signatures of known vulnerabilities of different servers, OS and network settings which are updated as new vulnerabilities are found [8].

2.2 Web Application Scanners

Many tools are available in the market which scans web application to detect vulnerabilities. Nowadays, most of the organizations implement web application scanning in early stages of development. In this, scanners find the weakness of the targeted system. By this, they find loopholes in the system before someone else does [6]. Vulnerability scanning can be done in two ways; using automated tools or by manual testing. Manual testing is very cumbersome and time consuming and as it is implemented by a human, there are chances that some weaknesses are not caught by the testing and the application becomes vulnerable. Hence, nowadays, organization rely heavily on automated tools to test the application.

2.3 Tools Background

2.3.1 OWASP Zed Attack Proxy

OWASP ZAP is one of the most popular and commonly used open source tool that is ideal for the developers and functional testers as well as it is useful to the experienced testers [11]. It can be run on both windows and Linux platform. The tool is written in Java programming language. Some of the features of ZAP are [24]:

- Intercepting Proxy: Analyzing requests and response.
- Scanner: Detecting Vulnerability
- Spider: Crawl a website
- Report Generation
- Brute Force: Perform Dictionary style attacks
- Fuzzing: Input of Random data strings in request headers and attacks
- Extensibility: Customized scripts to detect flaws

One of the biggest feature of ZAP is that it can be configured as proxy. Figure 2.1 the setup of ZAP acting as a proxy. This allows ZAP to record the requests and

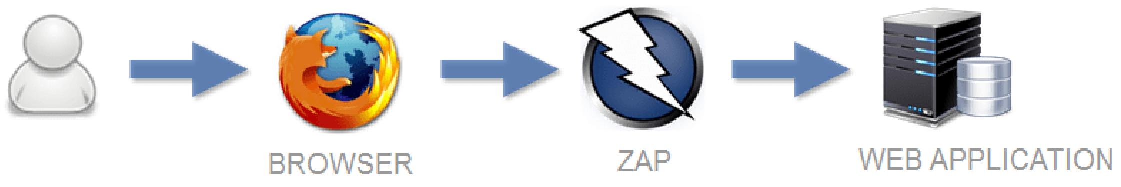
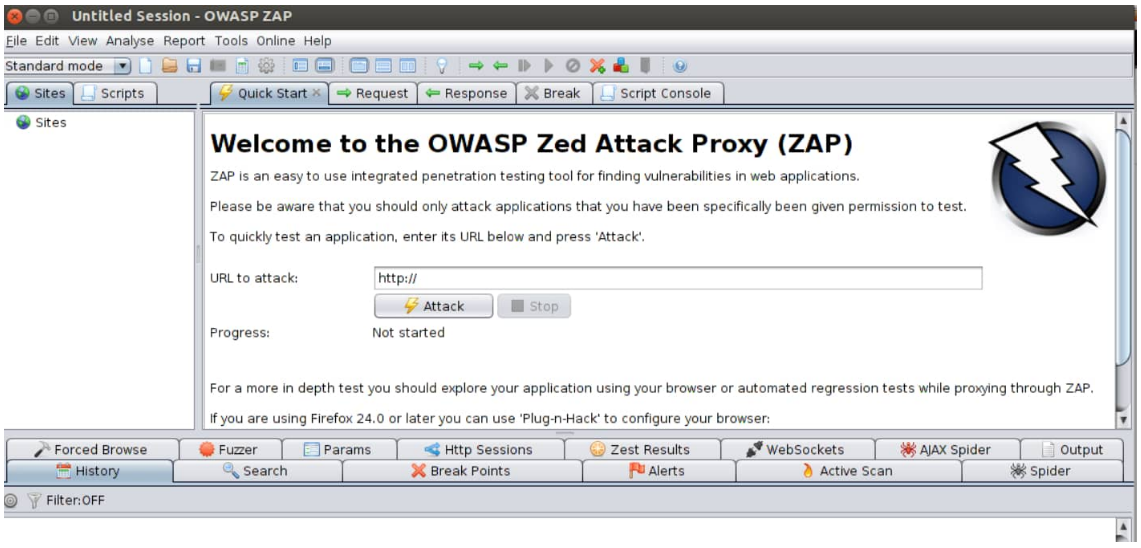


Figure 2.1: ZAP as Proxy



Welcome to the OWASP Zed Attack Proxy (ZAP)

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications.

Please be aware that you should only attack applications that you have been specifically been given permission to test.

To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://

Progress: Not started

For a more in depth test you should explore your application using your browser or automated regression tests while proxying through ZAP.

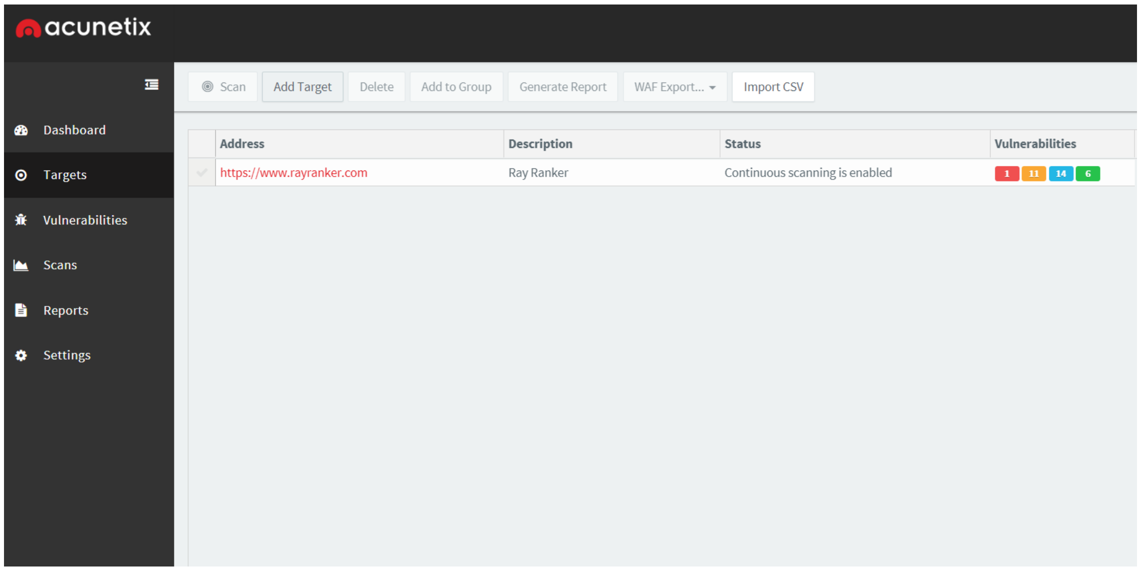
If you are using Firefox 24.0 or later you can use 'Plug-n-Hack' to configure your browser:

Figure 2.2: ZAP Interface

responses and then use them for a replay attack. ZAP is one of the most popular tool which is gained much support from over the past few years. It was ranked as the top security tool in 2013 [14]. More about OWASP ZAP is described in chapter 3 where actual applications are tested on ZAP.

2.3.2 Acunetix

Acunetix is a multi-threaded, lightning fast crawler that can crawl thousand of pages without interruptions. It is one of the most popular commercial tool used to scan web applications and has very high detection rate. It is very successful in finding SQL injection and XSS in web application. It has one of the lowest false



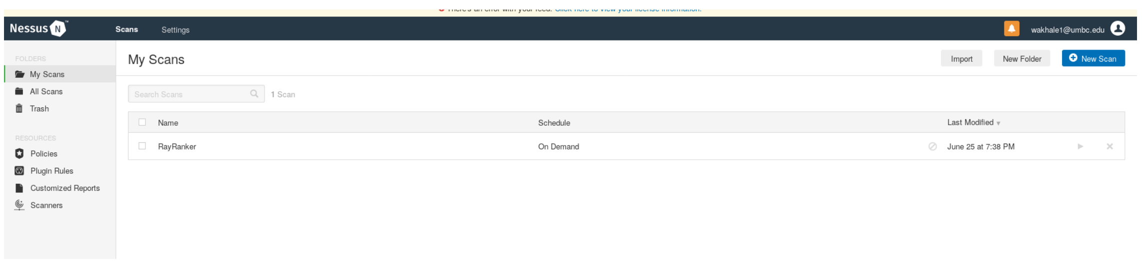
Address	Description	Status	Vulnerabilities
https://www.rayranker.com	Ray Ranker	Continuous scanning is enabled	1 11 14 6

Figure 2.3: Acunetix Web scanner Dashboard

positives as it combines black-box and white-box testing to enhance the detection rate. It has very high success rate when it is tested on applications which are developed in PHP and .NET platforms. The trial version which we used for this research comes for the windows platform and once successfully installed, starts a service which can be accessed through web browser. It is only developed for testing web applications and hence it is very popular among developers and organizations to test their product to find loopholes in their application.

2.3.3 Nessus Pro

Nessus Professional Vulnerability scanner is one of the most recognized and powerful commercial tool used to scan the web application. It is owned and developed by Tenable. According to the company Tenable [23], it has been used by more than one million users across the globe for vulnerability, configuration and compliance assessments. According to the company Tenable [23], some of the advantages



Name	Schedule	Last Modified
RayRanker	On Demand	June 25 at 7:38 PM

Figure 2.4: Homepage Of Nessus Pro

of using Nessus are:

- Easy to use: It just needs few clicks to scan the application.
- Comprehensive: It supports many technologies and identifies more vulnerabilities than its competitor.
- It has very high speed scanning with very low false positives.

Nessus Pro is a commercial tool. Hence, for this research, trial version is used. Trial version is available for seven days and Web application Scanning comes free with the trial version. Once, Nessus is fully installed, it starts a HTTP service which can be accessed through browsers and then applications can be scanned by configuring the requests.

2.4 Related Work

In this research, we aim to compare these tools and provide a detailed report on how these tools work and how effective they are to detect vulnerabilities present in the system. We do a comprehensive research on the tools in finding out the techniques they used, resources they need and how interactive and effective these tools are in finding out the vulnerabilities. Yuma Makino and Vitaly Klyuev [11],compared the tools like OWASP ZAP and Skipfish on Damn Vulnerable Web Application (DVWA) and analyzed the results. It lists down the features of each tool and number of vulnerabilities it is able to detect and their precision rate. In our research, we aim to test these tools on several application developed on different programming language like PHP, Java, and .NET. We test these tools on different environment, by hosting the application on different platform and then compare the tools based on the results in finding out whether the development environment affects the security of the application. Testing all tools on DVWA is a good approach but DVWA is a application developed for educational purpose and doesn't emulate real world application. Hence, testing tools on applications developed on different platforms gives good detailed analysis about the tools and helps us to analyze the tools effectively. Jose Fonseca and Marco Vieira [8] did a case study on several tools in which they injected realistic software faults and then compared the tools based on the result. They tested a PHP developed web application by using LAMP stack and then tested and evaluated all the tools. Our proposed methodology uses a platform Kali Linux in which all the tools are deployed. Then it scans the web application developed in different programming language and hosted on different platforms to detect the vulnerabilities present in the application. M.Curphey and R. Arawo [5], in their research also did a similar research in analyzing the tools which they use in their organization. Instead of testing the whole application, they used several tools for specific areas. For example, they compared tools which are source code analyzers and some tools which are Database Scanner. In our research, we aim to provide a detailed and comprehensive report on each tool comprising of several parameters and criteria to evaluate the scanners. Our objective is to evaluate each tool by doing a black box testing on all the applications and give a detailed report by comparing all the tools and analyze them. We compare them by using several parameters like ease to use, their effectiveness and the resources which they consume to test the application.

Chapter 3

Methodology

3.1 Introduction

To evaluate different web scanners and analyze their output, we installed all the tools on Kali Linux platform. Kali Linux is the world's most powerful and popular penetration testing platform which is used by security professionals in wide areas like vulnerability assessment, forensics and engineering. It is a flexible framework which can be customized by the professionals according to their needs [10]. Continuous refinement to the operating system and the platform makes it ideal for penetration testing framework which allows you to exploit many features of Linux and use information from open source community. It is just not collection of tools, but it is rather a framework which security testers and professional use to do penetration testing.

Kali Linux is security auditing Linux distribution based on Debian GNU/Linux. Kali Linux is derived from Debian testing and hence most of the packages are already available to it from Debian repository [10].

Kali Linux can be used by several types of users. Obviously, users can install them on their laptop to do penetration testing, but server administrators can also use it monitor network, forensics and monitoring. Once the Kali Linux system is installed ,all kinds of tasks and activities that are relevant to the testers are

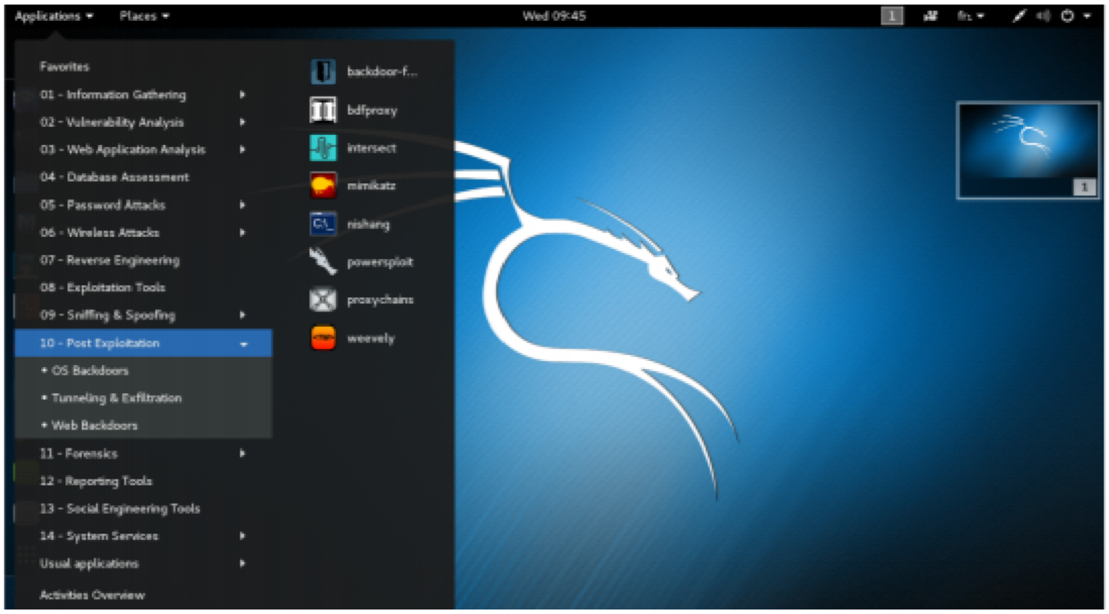


Figure 3.1: Kali Linux Application Menu

available. Figure 3.1 shows the application menu of Kali Linux system. These tasks and activities include [10]:

- Information Gathering
- Vulnerability Analysis
- Web Application Analysis
- Database assessment
- Password Attacks
- Reverse Engineering

3.1.1 Kali Linux Features

Kali Linux is the distribution which contains its own collection of tools tailored according to the users. It is pretty much like other Linux systems, but there are some features which differentiate Kali Linux with other systems [10].

- A Live system: ISO image of Kali Linux can also be used as a bootable live system.
- Forensic Mode: Kali Linux has forensic mode which can be enable from the boot menu.
- Complete Customizable: It is easy to customize Kali Linux based on our needs and preferences.
- Usable on Large number of ARM devices: It provides packages for the armel, armf and arm64 architectures.

As Kali Linux can be customized and is best suited for penetration testing. In this research, we used Kali Linux as an operating system in which all the tools were installed and then through the network, web applications are tested.

3.1.2 System Architecture

As Kali Linux is best suited for testing applications, it is used as a platform where all the tools are installed. As tools consumes a lot of computing power and resources when they test the application, a machine with higher memory and computing power was used.

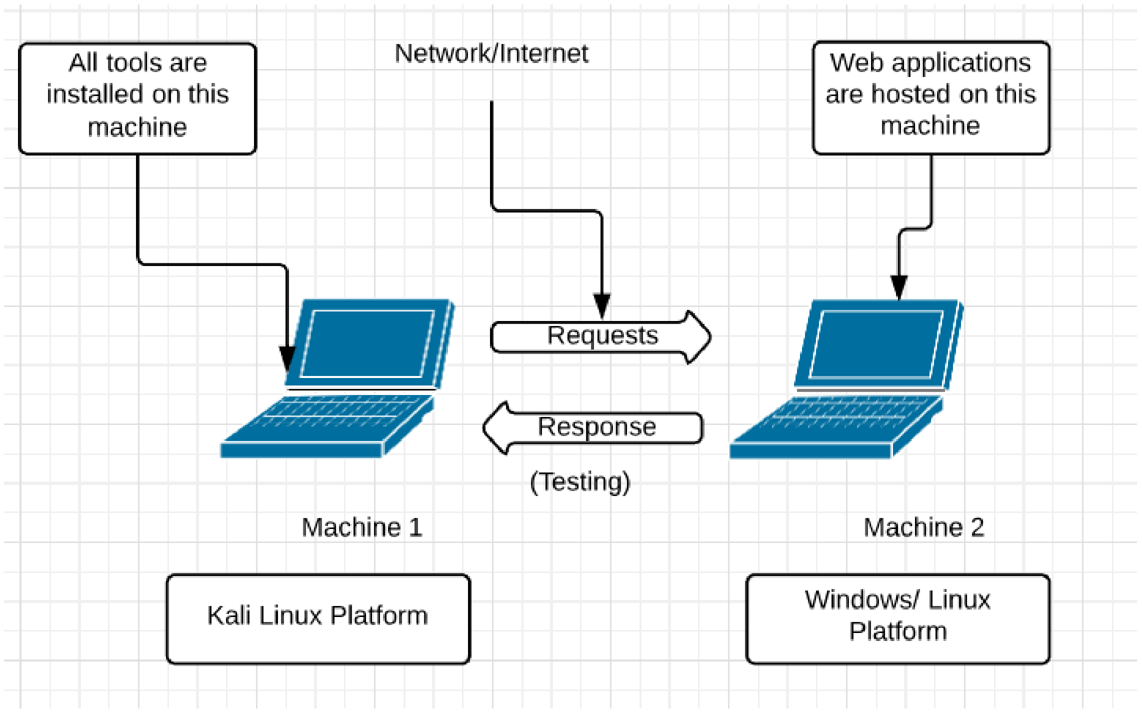


Figure 3.2: System architecture used for Testing

3.2 shows the architecture and methodology used for the testing purpose. Machine 1 in the 3.2 has following configuration:

- Operating System: Kali Linux:
- Installed Memory: 4 GB
- System Type: 64-bit

On machine 1, all the tools are installed. Each tools are available for the Linux platform. Some of the tools like W3af comes pre-installed with the Kali Linux system. To install all the packages, some of the packages needs to be installed.

To test different tools and analyze their results, web application scanners is run on both live websites and the applications which are hosted on machine 2 in 3.2. As legal issues needs to be resolved before testing live websites, this research focuses more on applications which can be hosted locally. Also, to analyze different applications, developed in different environment, several web applications developed on different platform are used to test these tools. These applications are hosted on Machine 2 which has following configuration.

- Intel core i7-8550U
- Operating System: To analyze the tools effectively and check their effectiveness in testing web applications and how the host machine operating system affects their capabilities, Machine 2 has both WINDOWS and Linux platform. And web applications which are to be tested are hosted on both Linux and WINDOWS platform.
- Installed Memory: 8.00 GB
- System Type: 64-bit

3.2 Web Applications used for testing

Web applications can be developed in several programming languages. It is the prerogative of web application developer to decide which environment he/she decide to develop the application. For this research, for testing purpose, five applications developed in five different environment are used. The development environment

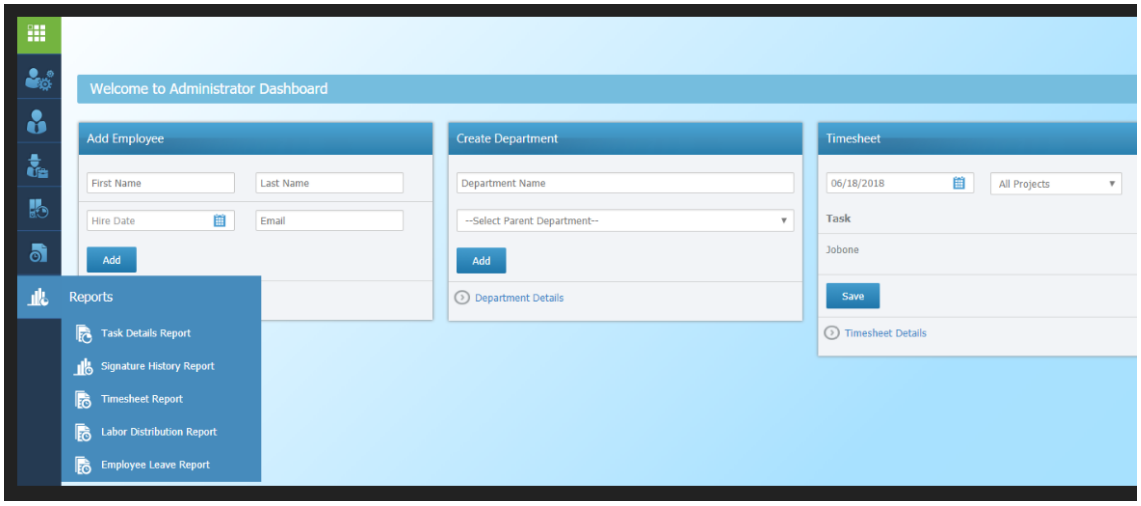
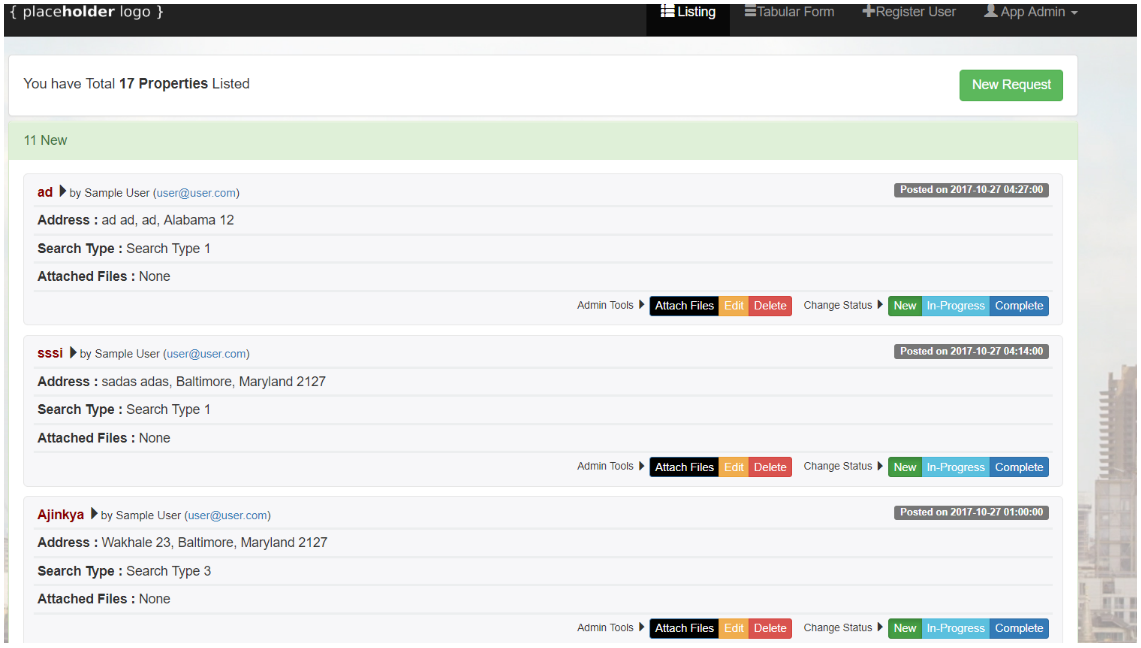


Figure 3.3: XPD Application

used in this research are :

- Java: A Java based application called XPDOffice, built on Spring/Hibernate framework, with JSP and AngularJS as front-end, Spring/Hibernate as back-end and PostgreSQL as database. It is a complex workforce management automation tool, which allows users and organizations to have a integrated solution for HR management, time and expense management, contract management and Customer Relationship management. Figure 3.3 shows the home page of XPDOffice. These application is used in this research as it is a very complex application and there are many APIs requests are generated through out the working of the application. As, the application deals with employees data and organization confidential information, it needs to be secured and robust. The application is hosted on machine 2 and requires Java 8 and a web server. It also needs a PostgreSQL database for the back-end.



You have Total **17 Properties** Listed

11 New

ad by Sample User (user@user.com) Posted on 2017-10-27 04:27:00

Address : ad ad, ad, Alabama 12

Search Type : Search Type 1

Attached Files : None

sssi by Sample User (user@user.com) Posted on 2017-10-27 04:14:00

Address : sadas adas, Baltimore, Maryland 2127

Search Type : Search Type 1

Attached Files : None

Ajinkya by Sample User (user@user.com) Posted on 2017-10-27 01:00:00

Address : Wakhale 23, Baltimore, Maryland 2127

Search Type : Search Type 3

Attached Files : None

Figure 3.4: Property Application

- PHP Application: This is a prototype PHP application developed on LAMP stack, which is used by the government agency to list and maintain the properties which are in the stage of foreclosure. Government agency employees use this application to store specific information about each houses and then update the process of foreclosure on the application. Figure 3.4 shows the homepage for the user "Admin". It has all the properties which are assigned to him and he has all the access to update the request to complete from New and attach files regarding to the property. As these application is very sensitive and it contains confidential data about the house owners, it needs to be secured and prone to attacks. Front-end is developed on HTML/CSS/Bootstrap, the middle layer is written in PHP and MYSQL is the database.

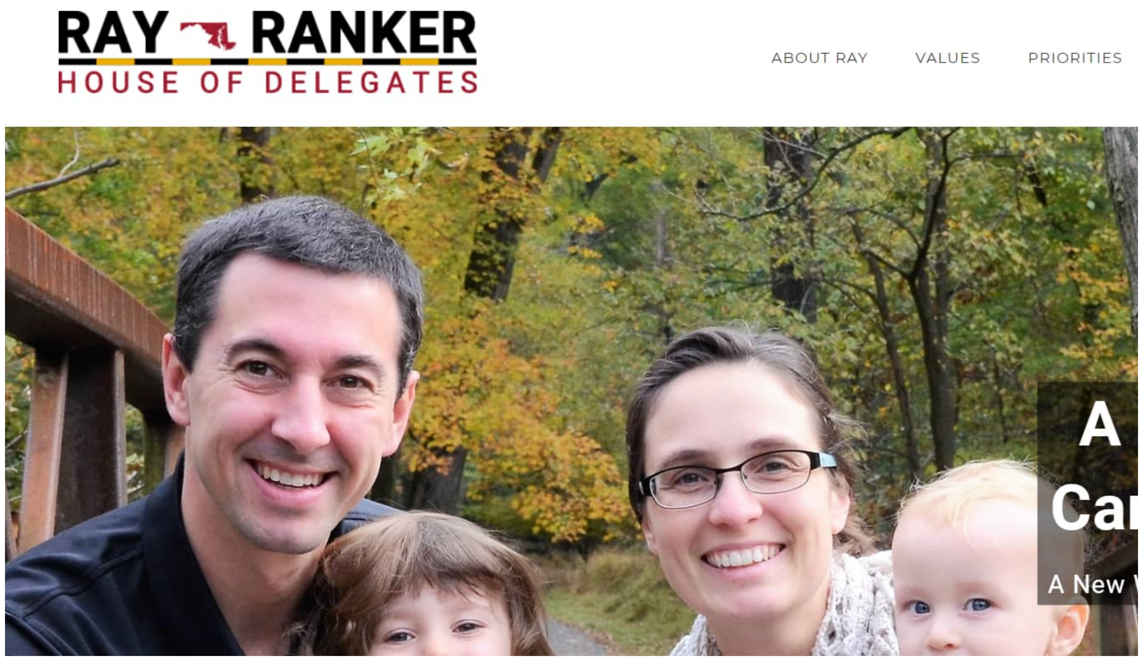


Figure 3.5: Homepage of website rayranker.com

- Static Application: A static website which contains normal forms and static pages. It contains information about a candidate called **Ray Ranker**, who is running for house of delegates in the state of Maryland [20]. For testing a static website, this research used a static website with address as https://rayranker.com. As it is a live website, permission has been taken from the concerned people. To test the robustness of the tools effectively, a static web application was used. Figure 3.5 shows the homepage of the website.

To analyze the tools effectively, wide range of applications are tested which differs in development environment, database setup, complexity and resources they use. Once all the applications are tested and scanned from various tools, results are analyzed and then a conclusion is made about all the tools and their efficiency.

3.3 Testing tools on Application

Each tools has a interface which allows you to enter URLs for the target application. All the applications mentioned in section 3.2 are hosted on Windows machine and uses several components like Web server and database to run smoothly. Once the application is hosted on machine 2 as shown in 3.2, the tools installed on machine 1 scans the web application and lists out the vulnerabilities present in the application by using the target URL. Detailed analysis and results of all the tools are presented in Chapter 4 for all the applications mentioned in section 3.2.

Chapter 4

Results

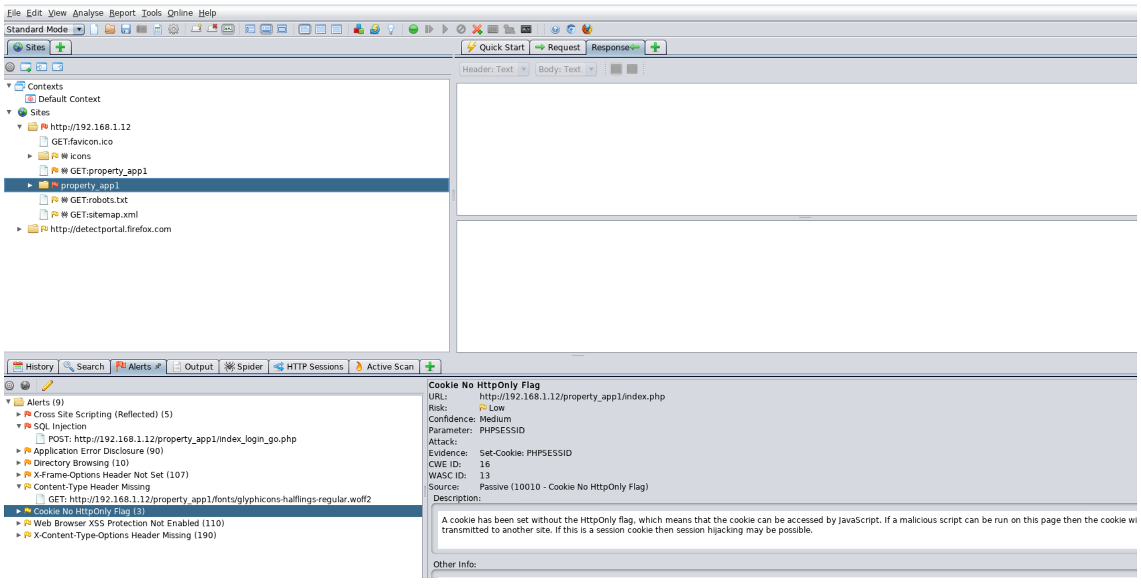
4.1 Results of Scanning Tools on Web Applications

4.1.1 Testing OWASP ZAP

OWASP ZAP has the capability to act as a proxy between web server and the browser. You can modify the settings of ZAP connection in tools section of ZAP interface. Once, it is configured to listen to the proxy, all the applications activity which are run on browser are all caught by ZAP. ZAP stores all the information about requests and responses made by the application. Once, all the manual activity is completed, ZAP has an option of Spider, which finds out all the pages which has not been accessed by the manual activity. And, after that active scan attacks all the possible scenarios and lists out all the vulnerabilities present in the application.

4.1.1.1 ZAP on PHP Application

The PHP based web application which is used for testing in this research is a complex and sensitive application. After testing OWASP ZAP on the IRS application, eleven vulnerabilities were found. Out of these nine alerts, four were high priority alerts, three were medium priority alerts and four were low priority alerts. High priority alerts were Cross-Site Scripting and SQL injection. It found five



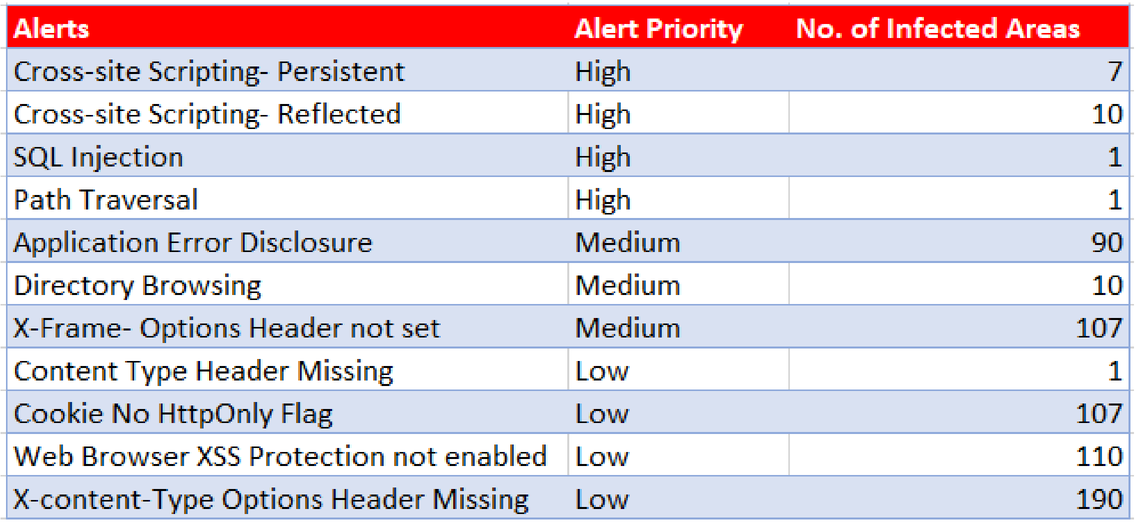
```
▼ Contexts
    Default Context
▼ Sites
  ▼ http://192.168.1.12
      GET:favicon.ico
    ▶ icons
      GET:property_app1
    ▶ property_app1
      GET:robots.txt
      GET:sitemap.xml
  ▶ http://detectportal.firefox.com
```

```
▼ Alerts (9)
  ▶ Cross Site Scripting (Reflected) (5)
  ▼ SQL Injection
      POST: http://192.168.1.12/property_app1/index_login_go.php
  ▶ Application Error Disclosure (90)
  ▶ Directory Browsing (10)
  ▶ X-Frame-Options Header Not Set (107)
  ▼ Content-Type Header Missing
      GET: http://192.168.1.12/property_app1/fonts/glyphicons-halflings-regular.woff2
  ▶ Cookie No HttpOnly Flag (3)
  ▶ Web Browser XSS Protection Not Enabled (110)
  ▶ X-Content-Type-Options Header Missing (190)
```

```
Cookie No HttpOnly Flag
URL:         http://192.168.1.12/property_app1/index.php
Risk:        Low
Confidence:  Medium
Parameter:   PHPSESSID
Attack:
Evidence:    Set-Cookie: PHPSESSID
CWE ID:      16
WASC ID:     13
Source:      Passive (10010 - Cookie No HttpOnly Flag)
Description:
  A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie wi
  transmitted to another site. If this is a session cookie then session hijacking may be possible.
Other Info:
```

Figure 4.1: Screenshot of Results of Scanning ZAP tool on PHP Application

areas in the code where cross side scripting can be infected and in one place where SQL injection is possible. This application is susceptible to one of the most popular attack that is SQL injection. Also, there are several instances where error/warning messages can disclose sensitive information and it is possible to view the directory listing, which can reveal hidden scripts, backup source files which can be accessed to read sensitive information. Figure 4.1 shows the screen-shot of the results of scanning IRS application through ZAP.



Alerts	Alert Priority	No. of Infected Areas
Cross-site Scripting- Persistent	High	7
Cross-site Scripting- Reflected	High	10
SQL Injection	High	1
Path Traversal	High	1
Application Error Disclosure	Medium	90
Directory Browsing	Medium	10
X-Frame- Options Header not set	Medium	107
Content Type Header Missing	Low	1
Cookie No HttpOnly Flag	Low	107
Web Browser XSS Protection not enabled	Low	110
X-content-Type Options Header Missing	Low	190

Figure 4.2: List of Vulnerabilities in IRS application found through ZAP

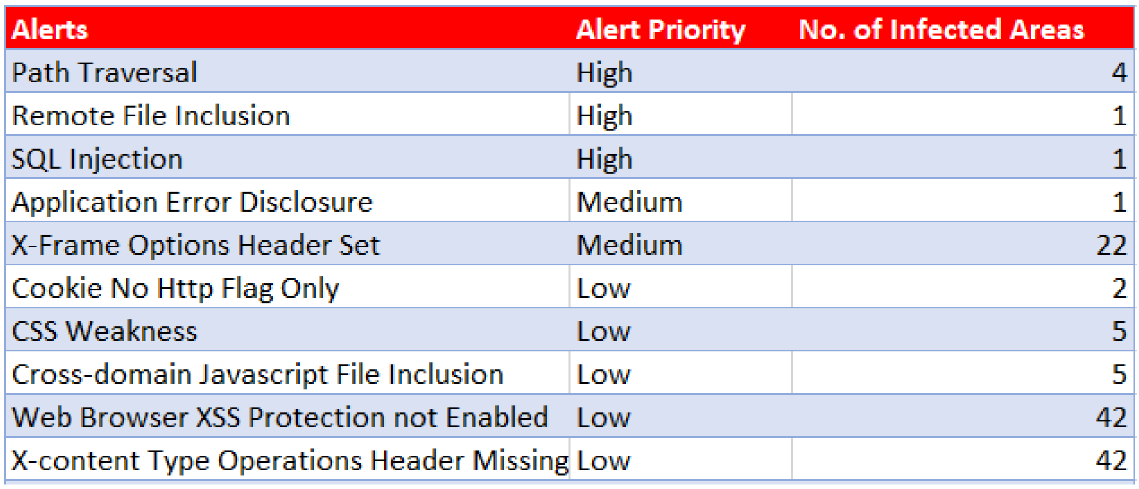
Figure 4.2 shows the vulnerabilities present in the PHP based applications. It also shows the number of instances where code is vulnerable to the attack. ZAP uses Spider to find out the hidden links and pages which are not accessed manually and then make crafted requests to the applications to find out the vulnerabilities. Following shows ZAP data scanning the IRS application.

- Total URLs Found: **27**
- Total Requests Made: **10518**
- Total Time taken for Scanning: **1 hour**
- Total Number of Alerts: **11**

4.1.1.2 ZAP on Java Application

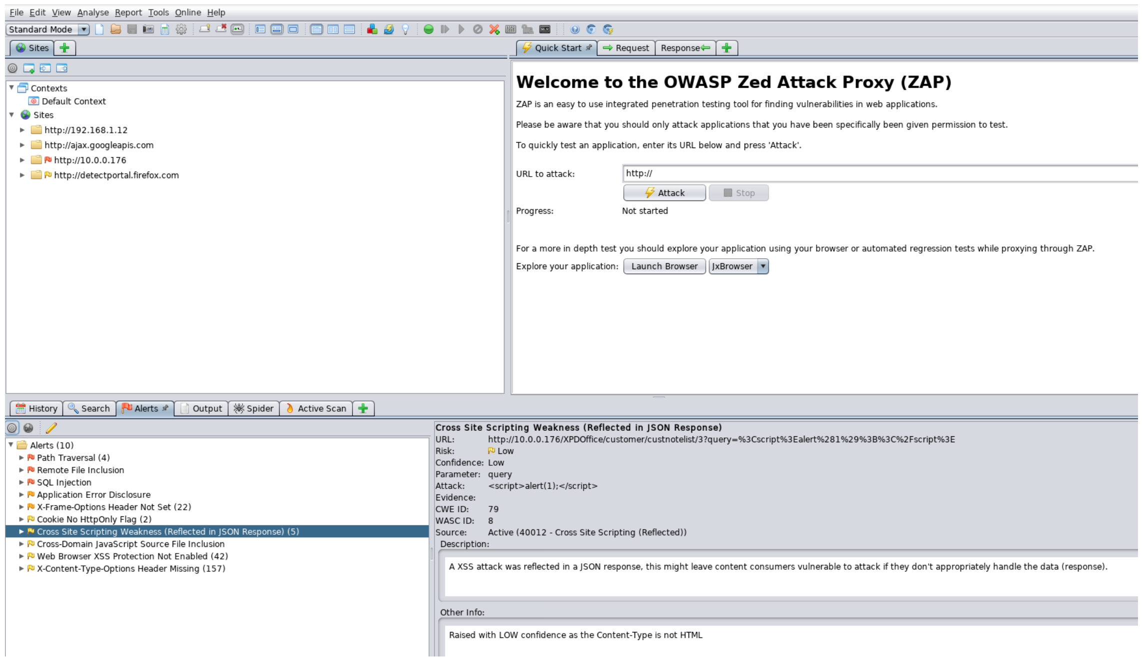
Testing ZAP on Java Application provided significant results which exposes vulnerabilities present in the application. We tested OWASP ZAP on our web application which was developed in Java. Total ten vulnerabilities were found by scanning XPD from ZAP. Out of ten, three were of High priority, two were medium and remaining five were low priority. It took almost one hour to scan the whole application. Following shows ZAP data on Scanning ZAP on XPDOffice and figure 4.1.1.2 shows the screenshot of the results.

- Total URLs Found: **457**
- Total Requests Made: **97394**
- Total Time taken for Scanning : **2 hour 10 minutes**
- Total Number of Alerts: **10**



Alerts	Alert Priority	No. of Infected Areas
Path Traversal	High	4
Remote File Inclusion	High	1
SQL Injection	High	1
Application Error Disclosure	Medium	1
X-Frame Options Header Set	Medium	22
Cookie No Http Flag Only	Low	2
CSS Weakness	Low	5
Cross-domain Javascript File Inclusion	Low	5
Web Browser XSS Protection not Enabled	Low	42
X-content Type Operations Header Missing	Low	42

Figure 4.3: List of Vulnerabilities in XPDOffice found through ZAP



Contexts
Default Context
Sites
http://192.168.1.12
http://ajax.googleapis.com
http://10.0.0.176
http://detectportal.firefox.com

Welcome to the OWASP Zed Attack Proxy (ZAP)

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications.

Please be aware that you should only attack applications that you have specifically been given permission to test.

To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://

Progress: Not started

For a more in depth test you should explore your application using your browser or automated regression tests while proxying through ZAP.

Explore your application:

Alerts (10)
- Path Traversal (4)
- Remote File Inclusion
- SQL Injection
- Application Error Disclosure
- X-Frame-Options Header Not Set (22)
- Cookie No HttpOnly Flag (2)
- Cross Site Scripting Weakness (Reflected in JSON Response) (5)
- Cross-Domain JavaScript Source File Inclusion
- Web Browser XSS Protection Not Enabled (42)
- X-Content-Type-Options Header Missing (157)

Cross Site Scripting Weakness (Reflected in JSON Response)

URL:	http://10.0.0.176/XPDOffice/customer/custnotelist/3?query=%3Cscript%3Ealert%281%29%3B%3C%2Fscript%3E
Risk:	Low
Confidence:	Low
Parameter:	query
Attack:	<script>alert(1);</script>
Evidence:	
CWE ID:	79
WASC ID:	8
Source:	Active (40012 - Cross Site Scripting (Reflected))

Description:

A XSS attack was reflected in a JSON response, this might leave content consumers vulnerable to attack if they don't appropriately handle the data (response).

Other Info:

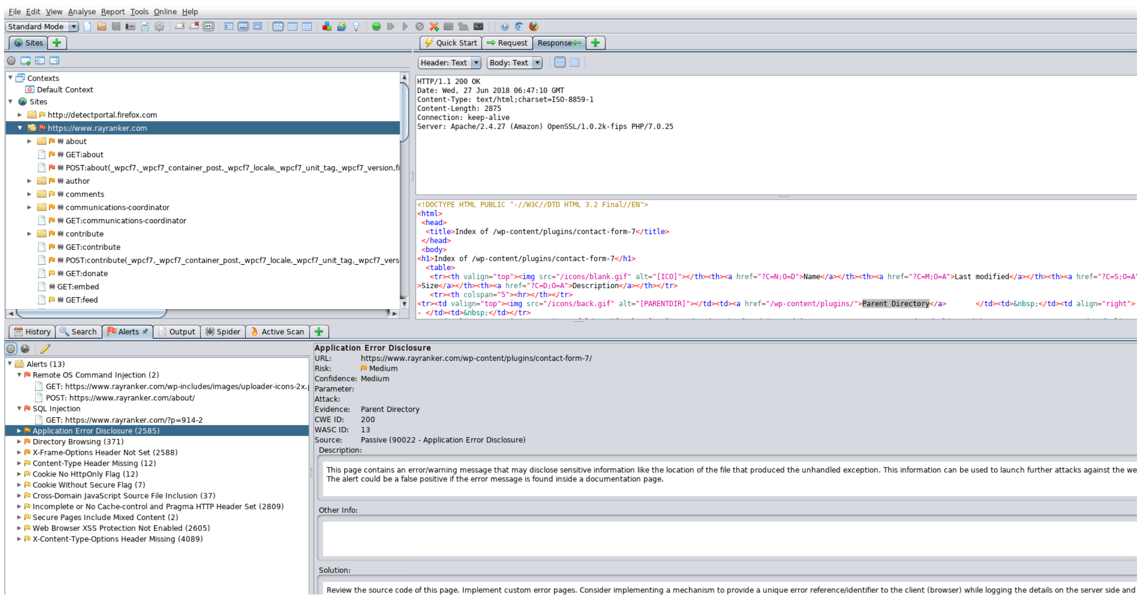
Raised with LOW confidence as the Content-Type is not HTML

Figure 4.4: Screenshot of Results of Scanning ZAP tool on XPDOffice

4.1.1.3 ZAP on Static Application

We tested OWASP ZAP on the static web application rayranker.com to find the security loopholes in the system. Again, ZAP is used as a proxy, which records all the activities occurring on the browser and then uses Spider to find out all the hidden pages and links and then uses Active scan to find out the vulnerabilities. We found total thirteen vulnerabilities in the website. Out of thirteen, two were of high priority, three were medium and remaining eight were of low priority alerts. It was able to found SQL injection in the application, which is considered to be the most common vulnerability found in the web applications and can lead to severe data loss. Figure 4.5 shows the screenshot of the results of scanning the website through ZAP.

It took almost six hours to test the whole application and the spider found total



```
Contexts
  Default Context
Sites
  http://detectportal.firefox.com
  https://www.rayranker.com
    about
    GET:about
    POST:about(_wpcf7,_wpcf7_container_post,_wpcf7_locale,_wpcf7_unit_tag,_wpcf7_version,fi
    author
    comments
    communications-coordinator
    GET:communications-coordinator
    contribute
    GET:contribute
    POST:contribute(_wpcf7,_wpcf7_container_post,_wpcf7_locale,_wpcf7_unit_tag,_wpcf7_vers
    GET:donate
    GET:embed
    GET:feed
```

```
HTTP/1.1 200 OK
Date: Wed, 27 Jun 2018 06:47:10 GMT
Content-Type: text/html;charset=ISO-8859-1
Content-Length: 2875
Connection: keep-alive
Server: Apache/2.4.27 (Amazon) OpenSSL/1.0.2k-fips PHP/7.0.25
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
 <head>
  <title>Index of /wp-content/plugins/contact-form-7</title>
 </head>
 <body>
<h1>Index of /wp-content/plugins/contact-form-7</h1>
  <table>
   <tr><th valign="top"><img src="/icons/blank.gif" alt="[ICO]"></th><th><a href="?C=N;O=D">Name</a></th><th><a href="?C=M;O=A">Last modified</a></th><th><a href="?C=S;O=A'
>Size</a></th><th><a href="?C=D;O=A">Description</a></th></tr>
   <tr><th colspan="5"><hr></th></tr>
<tr><td valign="top"><img src="/icons/back.gif" alt="[PARENTDIR]"></td><td><a href="/wp-content/plugins/">Parent Directory</a>       </td><td>&nbsp;</td><td align="right">
- </td><td>&nbsp;</td></tr>
```

```
Alerts (13)
  Remote OS Command Injection (2)
    GET: https://www.rayranker.com/wp-includes/images/uploader-icons-2x.
    POST: https://www.rayranker.com/about/
  SQL Injection
    GET: https://www.rayranker.com/?p=914-2
  Application Error Disclosure (2585)
  Directory Browsing (371)
  X-Frame-Options Header Not Set (2588)
  Content-Type Header Missing (12)
  Cookie No HttpOnly Flag (12)
  Cookie Without Secure Flag (7)
  Cross-Domain JavaScript Source File Inclusion (37)
  Incomplete or No Cache-control and Pragma HTTP Header Set (2809)
  Secure Pages Include Mixed Content (2)
  Web Browser XSS Protection Not Enabled (2605)
  X-Content-Type-Options Header Missing (4089)
```

Application Error Disclosure

URL:	https://www.rayranker.com/wp-content/plugins/contact-form-7/
Risk:	Medium
Confidence:	Medium
Parameter:	
Attack:	
Evidence:	Parent Directory
CWE ID:	200
WASC ID:	13
Source:	Passive (90022 - Application Error Disclosure)

Description:

This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be used to launch further attacks against the we
The alert could be a false positive if the error message is found inside a documentation page.

Other Info:

Solution:

Review the source code of this page. Implement custom error pages. Consider implementing a mechanism to provide a unique error reference/identifier to the client (browser) while logging the details on the server side and

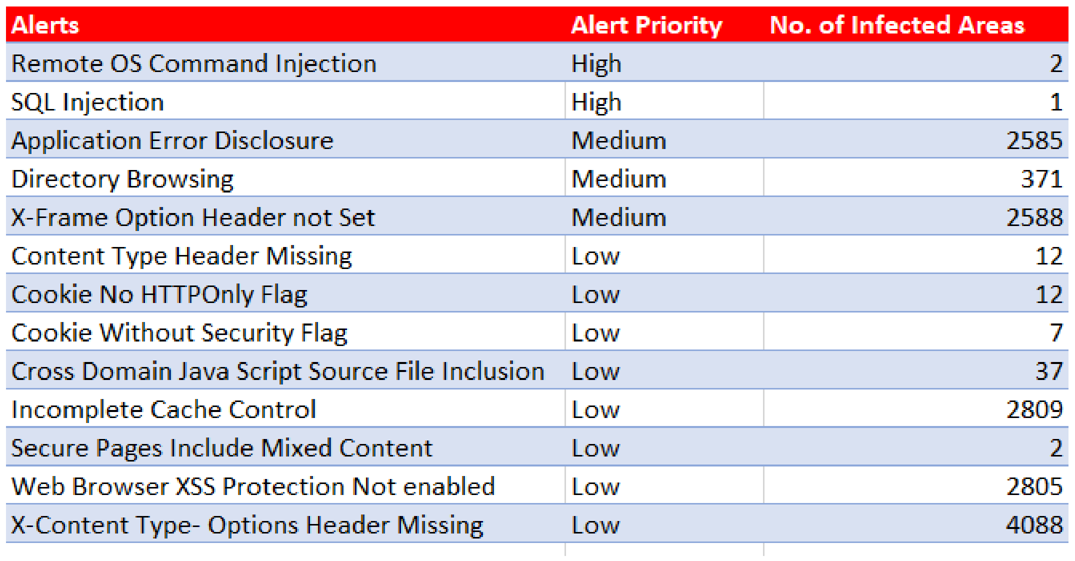
Figure 4.5: Results of Testing ZAP on Rayranker.com

5164 URLs to scan the whole application. Figure 4.6 shows the list of vulnerabilities present in the application and the number of infected areas for each alert. Following shows the ZAP data on scanning Rayranker.com through ZAP.

- Total URLs found: **5164**
- Total Time Taken in Hours: **6**
- Total Number of Alerts: **13**
- Total Requests Made: **437058**

4.1.2 Testing Nessus Pro

As Nessus Pro is a commercial tool, we used a trial version which is available for seven days. Web application scanning comes with the trial version and hence we were able to test Nessus Pro on our web application. Once Nessus Pro is installed on



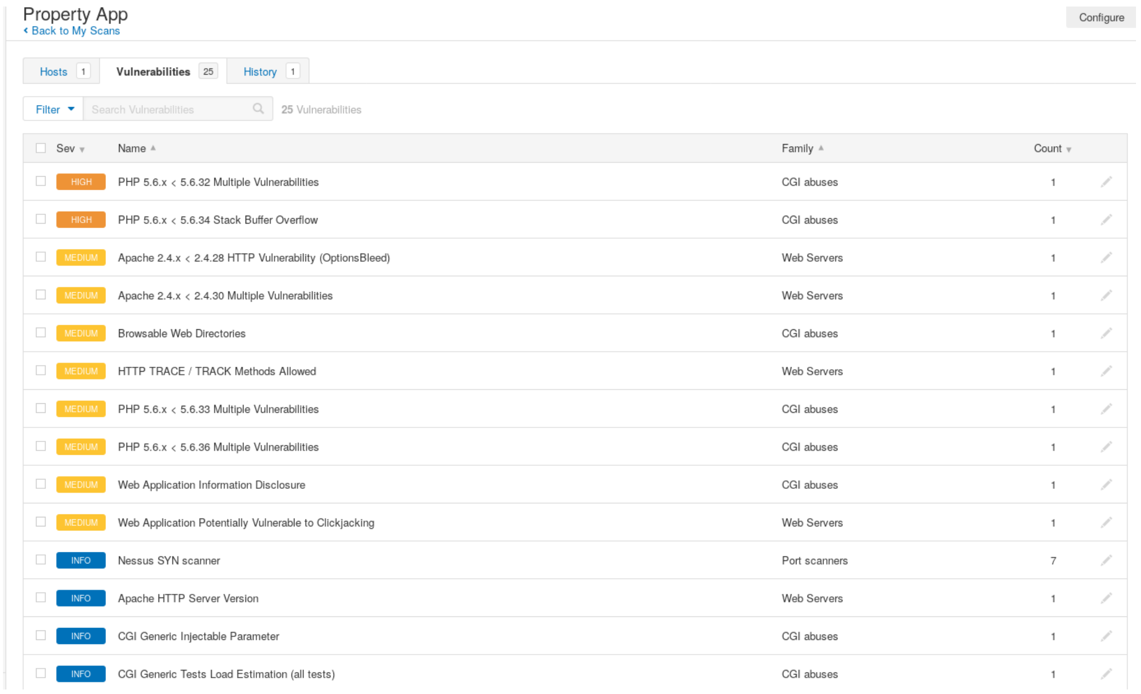
Alerts	Alert Priority	No. of Infected Areas
Remote OS Command Injection	High	2
SQL Injection	High	1
Application Error Disclosure	Medium	2585
Directory Browsing	Medium	371
X-Frame Option Header not Set	Medium	2588
Content Type Header Missing	Low	12
Cookie No HTTPOnly Flag	Low	12
Cookie Without Security Flag	Low	7
Cross Domain Java Script Source File Inclusion	Low	37
Incomplete Cache Control	Low	2809
Secure Pages Include Mixed Content	Low	2
Web Browser XSS Protection Not enabled	Low	2805
X-Content Type- Options Header Missing	Low	4088

Figure 4.6: List of Vulnerabilities in Rayranker.com found through ZAP

the machine, it works as a service which provides a interface to login on the browser. Once, you are logged in, it provides you the option for a new scan which can be configured. Once, scanning is complete, it lists down the vulnerabilities found by scanning the web application.

4.1.2.1 Nessus on PHP Application

We tested Nessus Pro on our PHP application and found 25 vulnerabilities. Out of twenty five, three were High priority alert and eight were of medium priority. It was able to find many possible vulnerabilities in the server as Nessus Pro scans the server first and then starts scanning the whole application. It took almost five hour to scan the whole application and Nesuss Pro trial version doest not reveals the requests it makes to scan the application. It was able to find potent vulnerabilities which makes the application very insecure especially through SQL injection and clickjacking.

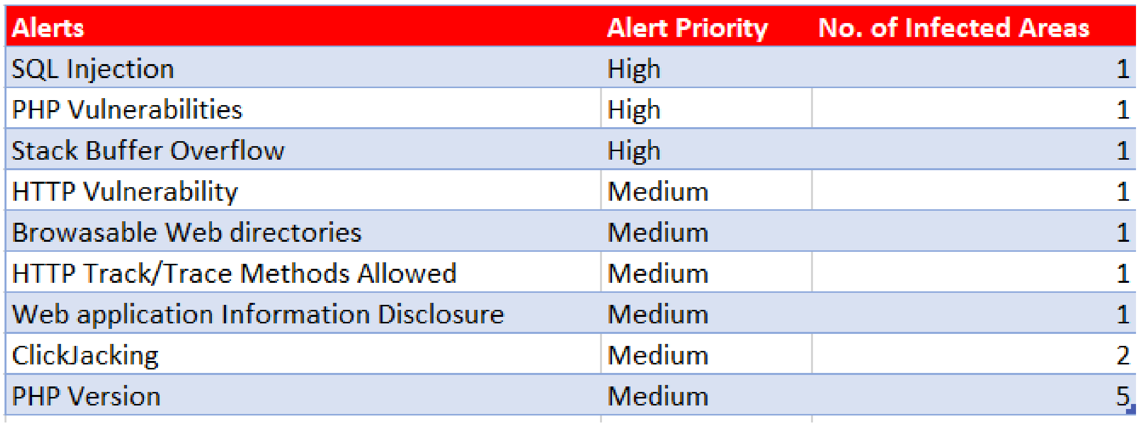


Sev	Name	Family	Count
HIGH	PHP 5.6.x < 5.6.32 Multiple Vulnerabilities	CGI abuses	1
HIGH	PHP 5.6.x < 5.6.34 Stack Buffer Overflow	CGI abuses	1
MEDIUM	Apache 2.4.x < 2.4.28 HTTP Vulnerability (OptionsBleed)	Web Servers	1
MEDIUM	Apache 2.4.x < 2.4.30 Multiple Vulnerabilities	Web Servers	1
MEDIUM	Browsable Web Directories	CGI abuses	1
MEDIUM	HTTP TRACE / TRACK Methods Allowed	Web Servers	1
MEDIUM	PHP 5.6.x < 5.6.33 Multiple Vulnerabilities	CGI abuses	1
MEDIUM	PHP 5.6.x < 5.6.36 Multiple Vulnerabilities	CGI abuses	1
MEDIUM	Web Application Information Disclosure	CGI abuses	1
MEDIUM	Web Application Potentially Vulnerable to Clickjacking	Web Servers	1
INFO	Nessus SYN scanner	Port scanners	7
INFO	Apache HTTP Server Version	Web Servers	1
INFO	CGI Generic Injectable Parameter	CGI abuses	1
INFO	CGI Generic Tests Load Estimation (all tests)	CGI abuses	1

Figure 4.7: Screenshot of Results of Scanning IRS application through Nessus Pro

4.1.2.2 Nessus on Java Application

XPD application which is developed in Java environment is scanned through Nessus to find out the vulnerabilities in the application. Total twenty seven vulnerabilities were found by scanning the XPD application. Out of twenty seven, two were high priority alerts, three were medium, one was low and remaining were info alerts. Total time to scan the application was around three hours. Figure 4.1.2.2 shows the screenshot of the results which were found by scanning XPD application through Nessus Pro. It was able to found big security loopholes like CSS and Path traversal which if exploited, can result in big data loss.



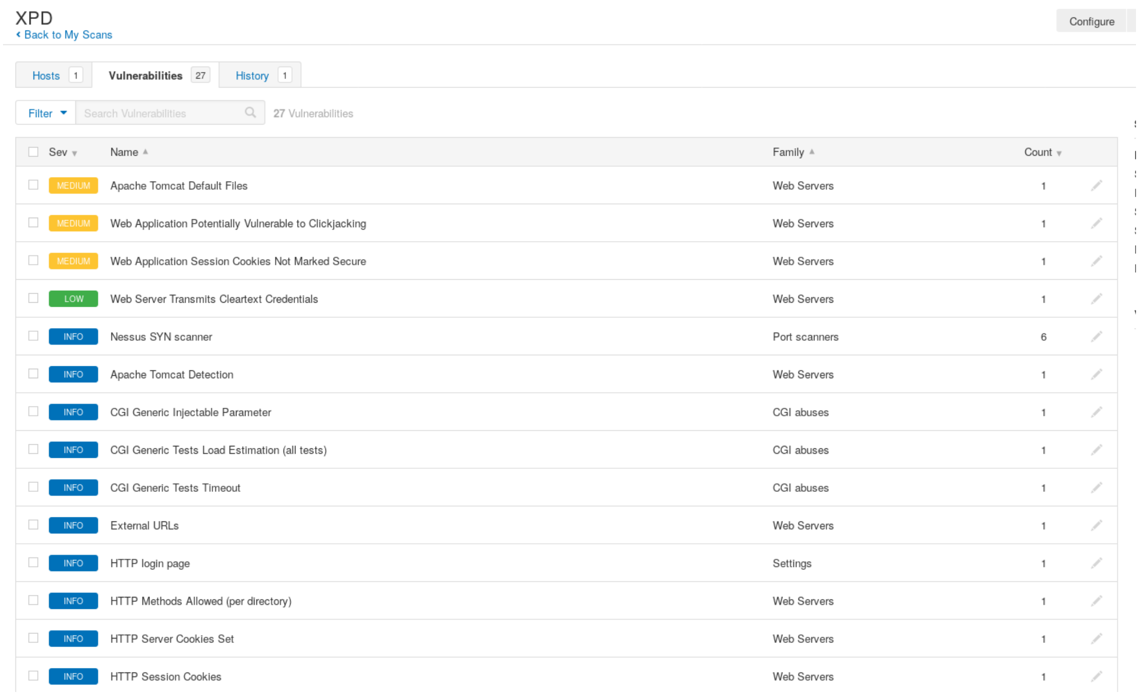
Alerts	Alert Priority	No. of Infected Areas
SQL Injection	High	1
PHP Vulnerabilities	High	1
Stack Buffer Overflow	High	1
HTTP Vulnerability	Medium	1
Browasable Web directories	Medium	1
HTTP Track/Trace Methods Allowed	Medium	1
Web application Information Disclosure	Medium	1
ClickJacking	Medium	2
PHP Version	Medium	5

Figure 4.8: Detailed Report on Scanning Nessus on IRS Application

4.1.2.3 Nessus on Static Application

We tested Nessus Pro on the static website which we used for our research. Total thirty vulnerabilities were found. Out of thirty, one were of high criticality, ten were medium and remaining were with low criticality and others were info warnings.

Total time taken to scan the whole application was around six hours and Nessus does not give detailed number of requests it uses to find out the vulnerabilities in the application. Figure 4.12shows the detailed number of vulnerabilities found and the number of infected areas. Total time taken to scan the whole application is around six hours. Nessus was able to find some big vulnerabilities like Stack Buffer flow which is a very big security issue and can lead to loss of confidentiality and integrity of the application. Figure 4.12 lists down the vulnerabilities found by scanning Rayranker.com through Nessus Pro.



XPD
‹ Back to My Scans

Hosts 1 | Vulnerabilities 27 | History 1

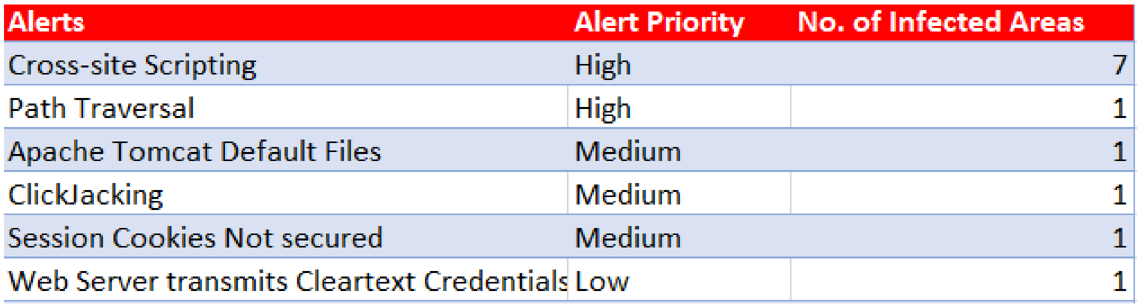
27 Vulnerabilities

Sev	Name	Family	Count
MEDIUM	Apache Tomcat Default Files	Web Servers	1
MEDIUM	Web Application Potentially Vulnerable to Clickjacking	Web Servers	1
MEDIUM	Web Application Session Cookies Not Marked Secure	Web Servers	1
LOW	Web Server Transmits Cleartext Credentials	Web Servers	1
INFO	Nessus SYN scanner	Port scanners	6
INFO	Apache Tomcat Detection	Web Servers	1
INFO	CGI Generic Injectable Parameter	CGI abuses	1
INFO	CGI Generic Tests Load Estimation (all tests)	CGI abuses	1
INFO	CGI Generic Tests Timeout	CGI abuses	1
INFO	External URLs	Web Servers	1
INFO	HTTP login page	Settings	1
INFO	HTTP Methods Allowed (per directory)	Web Servers	1
INFO	HTTP Server Cookies Set	Web Servers	1
INFO	HTTP Session Cookies	Web Servers	1

Figure 4.9: Screenshot of Results of Scanning XPD Application through Nessus Pro

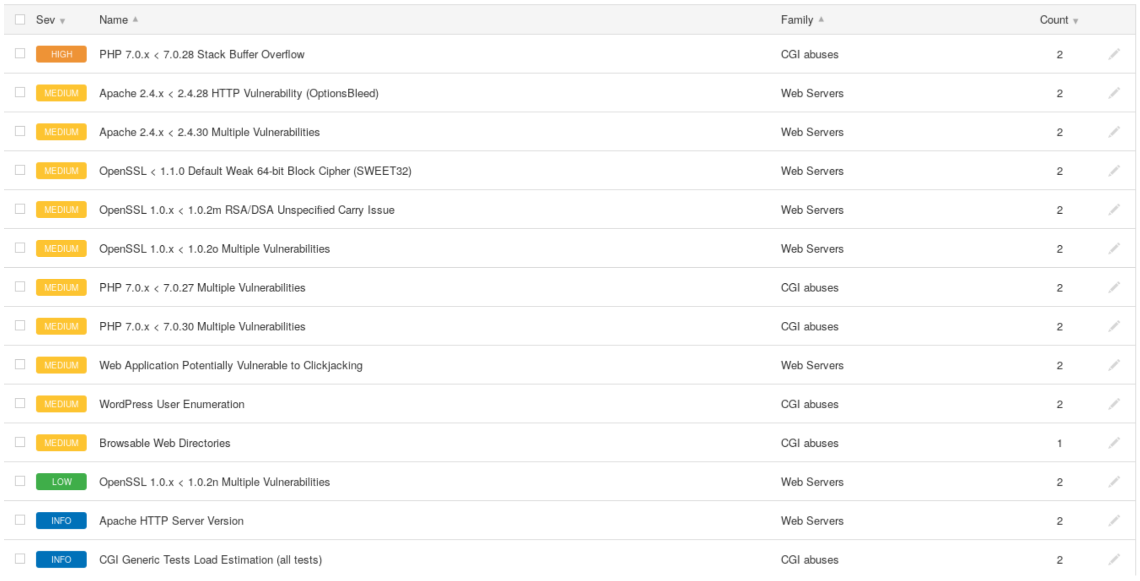
4.1.3 Testing Acunetix

Acunetix is a commercial tool and hence for this research we used a trial version which provides all the functionalities of the paid version free for fourteen days. Acunetix is a very popular web scanner because it specializes only in web application. Its interface is very interactive and less complex. Acunetix can automatically detect the technologies used by the web application and then scans the application using pre-defined settings.We scanned all our web applications through Acunetix and found many vulnerabilities.



Alerts	Alert Priority	No. of Infected Areas
Cross-site Scripting	High	7
Path Traversal	High	1
Apache Tomcat Default Files	Medium	1
ClickJacking	Medium	1
Session Cookies Not secured	Medium	1
Web Server transmits Cleartext Credentials	Low	1

Figure 4.10: Detailed Report on Scanning Nessus on XPD Application

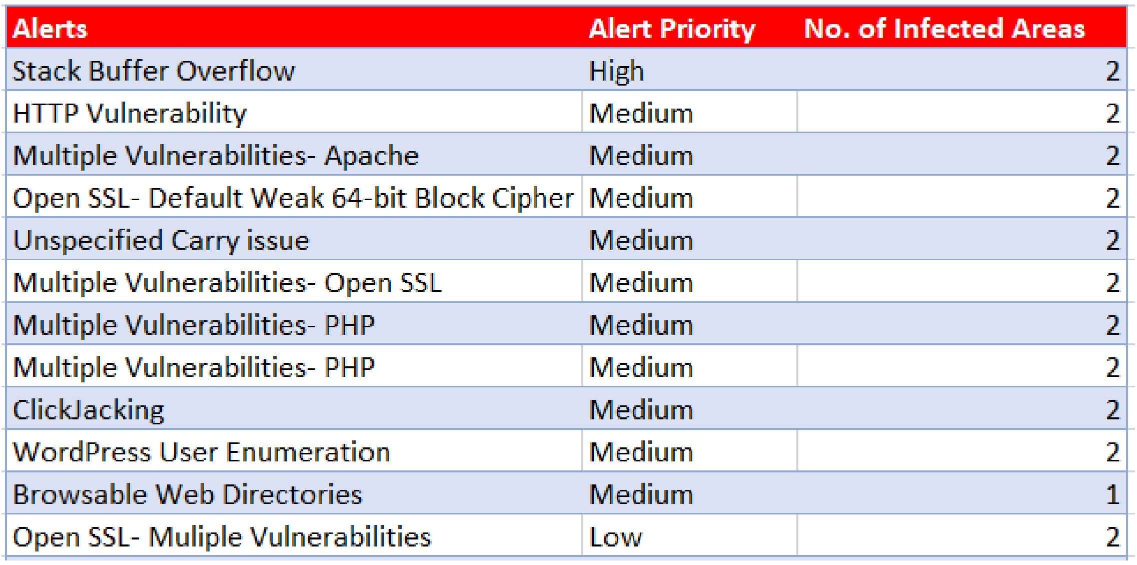


Sev	Name	Family	Count
HIGH	PHP 7.0.x < 7.0.28 Stack Buffer Overflow	CGI abuses	2
MEDIUM	Apache 2.4.x < 2.4.28 HTTP Vulnerability (OptionsBleed)	Web Servers	2
MEDIUM	Apache 2.4.x < 2.4.30 Multiple Vulnerabilities	Web Servers	2
MEDIUM	OpenSSL < 1.1.0 Default Weak 64-bit Block Cipher (SWEET32)	Web Servers	2
MEDIUM	OpenSSL 1.0.x < 1.0.2m RSA/DSA Unspecified Carry Issue	Web Servers	2
MEDIUM	OpenSSL 1.0.x < 1.0.2o Multiple Vulnerabilities	Web Servers	2
MEDIUM	PHP 7.0.x < 7.0.27 Multiple Vulnerabilities	CGI abuses	2
MEDIUM	PHP 7.0.x < 7.0.30 Multiple Vulnerabilities	CGI abuses	2
MEDIUM	Web Application Potentially Vulnerable to Clickjacking	Web Servers	2
MEDIUM	WordPress User Enumeration	CGI abuses	2
MEDIUM	Browsable Web Directories	CGI abuses	1
LOW	OpenSSL 1.0.x < 1.0.2n Multiple Vulnerabilities	Web Servers	2
INFO	Apache HTTP Server Version	Web Servers	2
INFO	CGI Generic Tests Load Estimation (all tests)	CGI abuses	2

Figure 4.11: Screenshot of Results of Scanning Rayranker.com through Nessus Pro

4.1.3.1 Acunetix on PHP Application

We scanned our IRS application which is developed in the PHP environment through Acunetix. As the application is developed in the PHP environment, we used PHP predefined settings. Acunetix provides this options when we configure our target for scanning. Total forty four vulnerabilities were found by scanning the application. Thirteen were of high severity, twenty one were of medium, six were low and four were informational. It was able to find vulnerabilities like Blind SQL injection, Cross-site scripting and Click-jacking.Following shows Acunetix data



Alerts	Alert Priority	No. of Infected Areas
Stack Buffer Overflow	High	2
HTTP Vulnerability	Medium	2
Multiple Vulnerabilities- Apache	Medium	2
Open SSL- Default Weak 64-bit Block Cipher	Medium	2
Unspecified Carry issue	Medium	2
Multiple Vulnerabilities- Open SSL	Medium	2
Multiple Vulnerabilities- PHP	Medium	2
Multiple Vulnerabilities- PHP	Medium	2
ClickJacking	Medium	2
WordPress User Enumeration	Medium	2
Browsable Web Directories	Medium	1
Open SSL- Muliple Vulnerabilities	Low	2

Figure 4.12: Detailed Report on Scanning Nessus on Rayranker.com

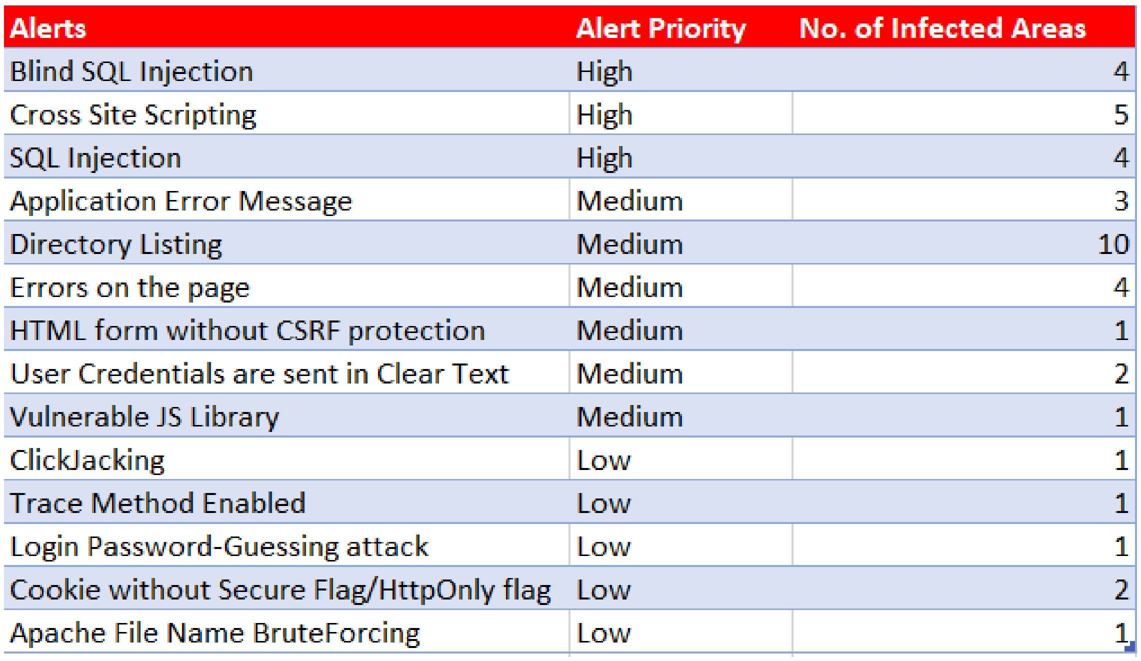
obtained by scanning the IRS application.

- Total time: 40 min
- Total Number of Requests: 17,643
- Total Number of Alerts: 44

4.1.3.2 Acunetix on Java Application

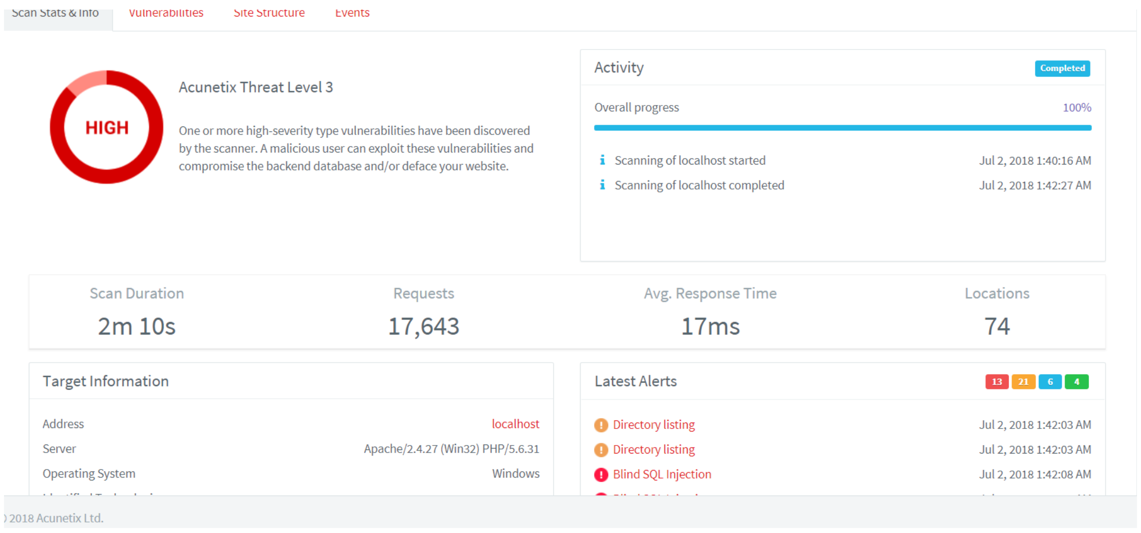
We tested Acunetix on our web application XPDOffice which is developed on the Java platform. At the time of configuring, we used Acunetix predefined settings for Java environment so that it can effectively scan the application. We found total twenty one vulnerabilities in the XPD Application. Out of twenty, six were of high severity, seven were of medium, three were low and remaining were informational. Following shows Acunetix data obtained by scanning the XPDOffice application.

- Total Time taken: 40 mins



Alerts	Alert Priority	No. of Infected Areas
Blind SQL Injection	High	4
Cross Site Scripting	High	5
SQL Injection	High	4
Application Error Message	Medium	3
Directory Listing	Medium	10
Errors on the page	Medium	4
HTML form without CSRF protection	Medium	1
User Credentials are sent in Clear Text	Medium	2
Vulnerable JS Library	Medium	1
ClickJacking	Low	1
Trace Method Enabled	Low	1
Login Password-Guessing attack	Low	1
Cookie without Secure Flag/HttpOnly flag	Low	2
Apache File Name BruteForcing	Low	1

Figure 4.13: List of Vulnerabilities in IRS application found through Acunetix



Acunetix Threat Level 3

One or more high-severity type vulnerabilities have been discovered by the scanner. A malicious user can exploit these vulnerabilities and compromise the backend database and/or deface your website.

Activity — Completed

Overall progress 100%

Scanning of localhost started Jul 2, 2018 1:40:16 AM
Scanning of localhost completed Jul 2, 2018 1:42:27 AM

Scan Duration	Requests	Avg. Response Time	Locations
2m 10s	17,643	17ms	74

Target Information

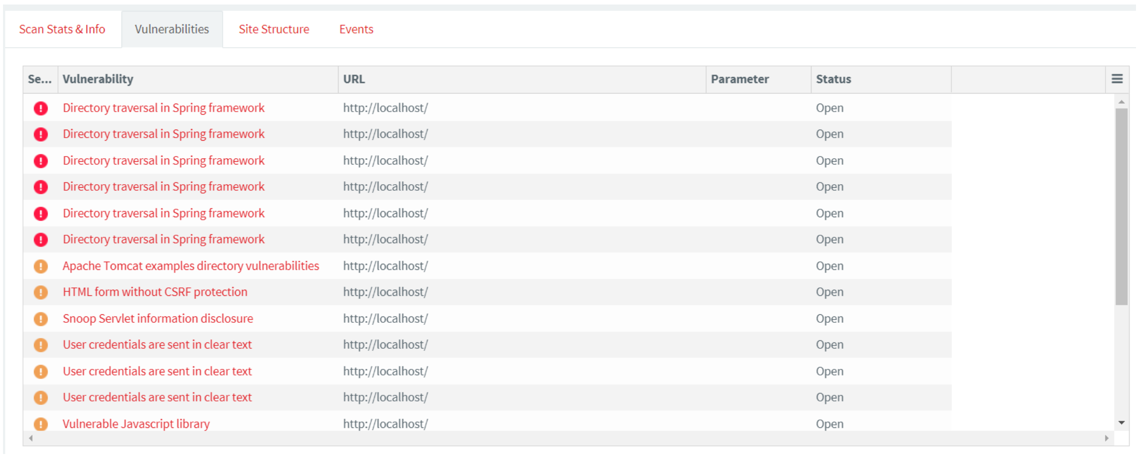
Address	localhost
Server	Apache/2.4.27 (Win32) PHP/5.6.31
Operating System	Windows

Latest Alerts 13 21 6 4

Directory listing	Jul 2, 2018 1:42:03 AM
Directory listing	Jul 2, 2018 1:42:03 AM
Blind SQL Injection	Jul 2, 2018 1:42:08 AM

2018 Acunetix Ltd.

Figure 4.14: Screenshot of the Results of Scanning IRS through Acunetix



Scan Stats & Info | Vulnerabilities | Site Structure | Events

Se...	Vulnerability	URL	Parameter	Status
●	Directory traversal in Spring framework	http://localhost/		Open
●	Directory traversal in Spring framework	http://localhost/		Open
●	Directory traversal in Spring framework	http://localhost/		Open
●	Directory traversal in Spring framework	http://localhost/		Open
●	Directory traversal in Spring framework	http://localhost/		Open
●	Directory traversal in Spring framework	http://localhost/		Open
●	Apache Tomcat examples directory vulnerabilities	http://localhost/		Open
●	HTML form without CSRF protection	http://localhost/		Open
●	Snoop Servlet information disclosure	http://localhost/		Open
●	User credentials are sent in clear text	http://localhost/		Open
●	User credentials are sent in clear text	http://localhost/		Open
●	User credentials are sent in clear text	http://localhost/		Open
●	Vulnerable Javascript library	http://localhost/		Open

Figure 4.15: Screenshot of the Results of Scanning XPD through Acunetix

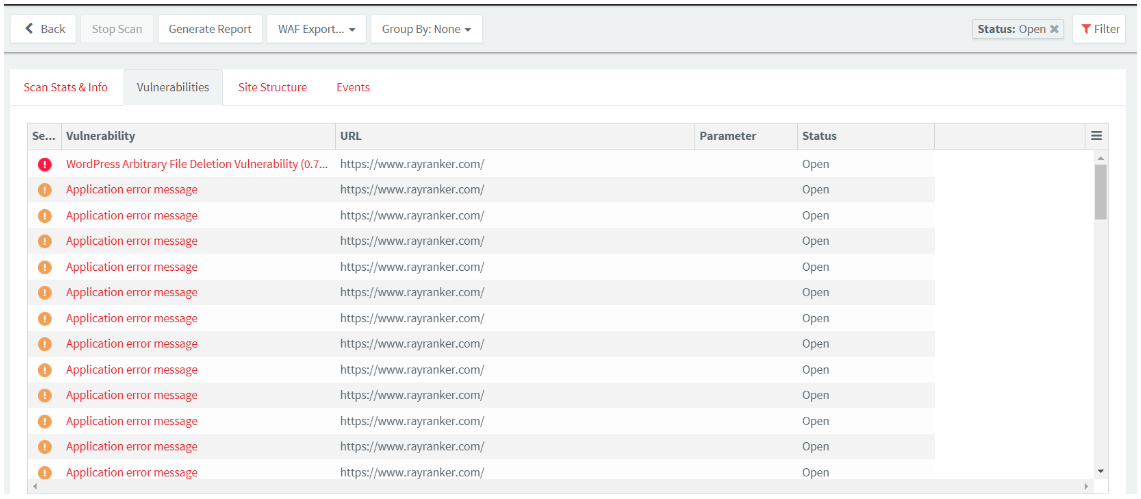
Alerts	Alert Priority	No. of Infected Areas
Directory Traversal in Spring Framework	High	6
Vulnerable JavaScript Library	Medium	1
User Credentials are sent in clear text	Medium	2
Snoop Servlet Information Disclosure	Medium	1
HTML form without CSRF protection	Medium	1
Apache Tomcat examples directory vulnerabilities	Medium	1
ClickJacking	Low	1
Cookie without HttpOnly flag set	Low	1
Cookie without Secure flag set	Low	1

- Total Number of Requests made: 2974

- Total Number of alerts: 21

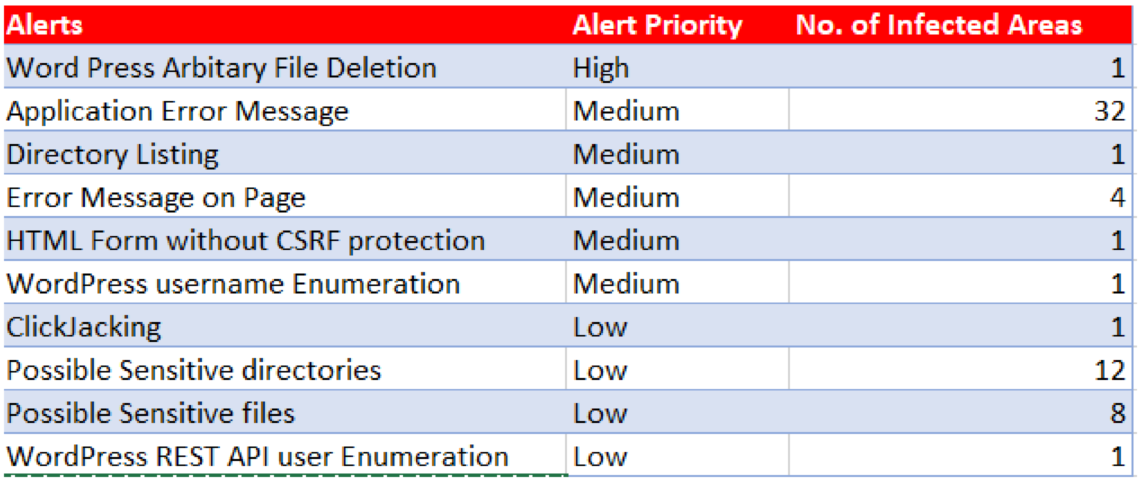
4.1.3.3 Acunetix on Static Application

We tested Acunetix on our static application called Rayranker.com. As it is a live website, it can be easily configured on the Acunetix interface. We found total sixty four vulnerabilities through Acunetix. Out of sixty four, one was of high severity, thirty nine of medium, twenty four of low and remaining were informational. Following shows Acunetix data obtained by scanning the website rayranker.com.



Se...	Vulnerability	URL	Parameter	Status
●	WordPress Arbitrary File Deletion Vulnerability (0.7...	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open
●	Application error message	https://www.rayranker.com/		Open

Figure 4.16: Screenshot of the Results of Scanning Rayranker.com through Acunetix



Alerts	Alert Priority	No. of Infected Areas
Word Press Arbitary File Deletion	High	1
Application Error Message	Medium	32
Directory Listing	Medium	1
Error Message on Page	Medium	4
HTML Form without CSRF protection	Medium	1
WordPress username Enumeration	Medium	1
ClickJacking	Low	1
Possible Sensitive directories	Low	12
Possible Sensitive files	Low	8
WordPress REST API user Enumeration	Low	1

Figure 4.17: List of Vulnerabilities in Rayranker.com found through Acunetix

- Total time taken: 5h40m31s
- Total Number of Requests: 55,902
- Total Number of Alerts: 70

We were able to detect some of the known vulnerabilities like Directory listing and Click-jacking by scanning the website through Acunetix.

This chapter discusses all the results this research got after testing the applications through all the tools. Chapter 5 analyzes those results and compare different tools based on different criteria and gives a conclusion about the tools.

Chapter 5

Analysis and Conclusion

5.1 Analysis

We tested our all three tools on our three web applications and found many vulnerabilities that are present in those applications. We aim to compare these tools and analyze them with different parameters. We search if there is something new or special in their approaches and how they scan the web applications. Also, we were interested in dissecting the procedures which they use to scan the application, their effectiveness in finding vulnerabilities and the resources they use. Lastly, how much expertise and knowledge is needed to understand the tool. In our research, we focused mainly on black-box testing of the web application i.e Enterprise focused tools. Our main aim was to analyze the tools comprehensively. We have done black-box testing on all the web applications. It does not know the internals of the application and uses crawling to find more links and fuzzing techniques over the web Http requests. We use two kinds of parameters to compare the tools. One is from the results which we got from scanning the web application and other is performance and functionalities which the tools provide. For the results, the parameters are :

IRS Application Data Comparison(PHP)			
Scanner Name	Total Alerts	Total Scan Time	Total Number of Requests
OWASP ZAP	11	197 s	10518
Acunetix	44	130s	17,643
Nessus Pro	25	5h	N/A

- Number of requests
- Number of Alerts
- Total Time taken

We compared the data which we got by scanning all the applications from different tools and then we analyzed those results.

5.1.1 PHP/IRS Application

If we analyze the results, we can see that Acunetix was able to find more vulnerabilities than other tools. Acunetix provides a functionality to adopt a predefined settings for the development environment in which the web application is developed. All the three tools were able to find SQL injection, which comes in top 10 vulnerabilities according to OWASP. Acunetix was the only tool to detect cross-site scripting problem in the application which can cause huge damage to the application. Moreover, all the tools were able to find the vulnerabilities, but Acunetix were able to detect more vulnerabilities deeply. It first scanned the web server first and gives you the loopholes in the web server. Acunetix gives you the option of customizing your target on the basis of the development environment and then it uses its predefined scripts to scan the application. If we analyze all the results, Acunetix

XPD Application Data Comparison(Java)			
Scanner Name	Total Alerts	Total Scan Time	Total Number of Requests
OWASP ZAP	10	1 h	97394
Acunetix	21	210s	2974
Nessus Pro	27	5h	N/A

was able to find more vulnerabilities at the application level. Nessus Pro is more expert in finding the network security issues, while ZAP was able to detect more or less same vulnerabilities as Nessus Pro.

5.1.2 Java Application/XPD

We tested all the tools on our Java application and found some interesting results. Both ZAP and Nessus were able to find vulnerabilities like Cross-site scripting and path traversal. Nessus is very fast but most of its scanning process involves scanning the network and server. Other tools apart from ZAP might have detected more number of alerts, but the issues were mostly at the server side. ZAP was able to find more vulnerabilities at the application level. It is well known for Java applications and the results which we got proves that. ZAP provides an option in which it acts as a proxy and records all the pages that browser visits and then uses Spider and Active scan to find more vulnerabilities. Acunetix was also able to find some of the vulnerabilities, but it doesn't cover the whole application comprehensively.

Static Website Data Comparison			
Scanner Name	Total Alerts	Total Scan Time	Total Number of Requests
OWASP ZAP	13	6 h	4,37,058
Acunetix	64	5h	55,902
Nessus Pro	30	6h	N/A

5.1.3 Static Website(Rayranker.com)

All three tools scanned the live website rayranker.com and found many vulnerabilities at the application level and at the server level. All the tools found medium severity alerts like Directory traversal and Open SSL flag set. But ZAP was able to find out the vulnerabilities like Cross-site scripting and SQL injection. Both these vulnerabilities comes in top ten vulnerabilities according to OWASP. As ZAP acts as a proxy, it gives you the option to record all the activity at the browser and then uses Spider to find more links. It works best for the static application as the pages are fixed. Acunetix was also able to find out many vulnerabilities but those were mostly informational and of medium severity. Hence, ZAP works best for the websites which are static.

5.2 Manual testing

For our two web applications XPDOffice and Property Application, we sent the vulnerabilities which we found from our tool to the development team of the organization who is responsible for developing it. Development team verified the vulnerabilities by doing manual testing on the applications and reported false positives,false negatives and True positives for each vulnerabilities detected by the tools.

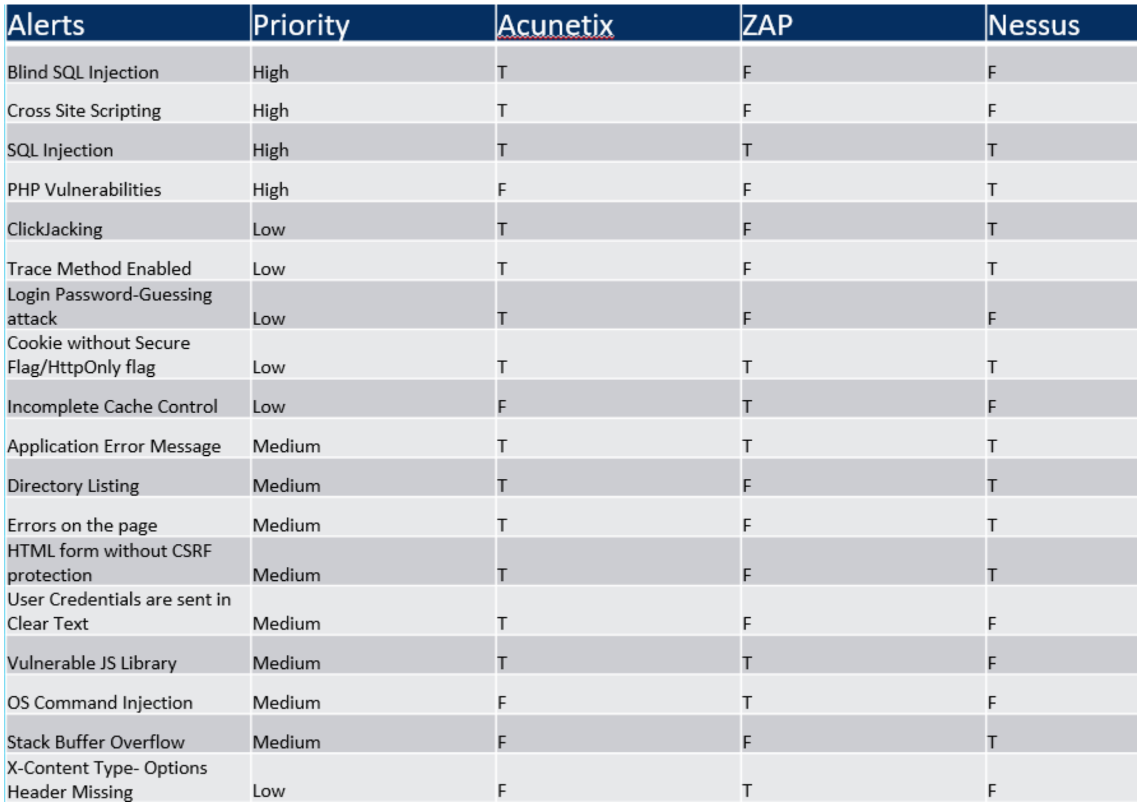
As all the tools generated report giving proper attack details about each vulnerabilities found, the testing team was able to test them and report about each tool.

5.2.1 Vulnerability Correctness Score

We used the data which we got from the testing team and generated a vulnerability correctness score for each tool by dividing the number of vulnerabilities which it was able to correctly detect to total number of vulnerabilities verified by the testing team. This score gives us a good parameter to analyze the tools precision rate and also evaluate them on the basis of false positives and false negatives.

$$VulnerabilityCorrectness = (NoOfmatchedAlerts/TotalAlertsVerified)$$

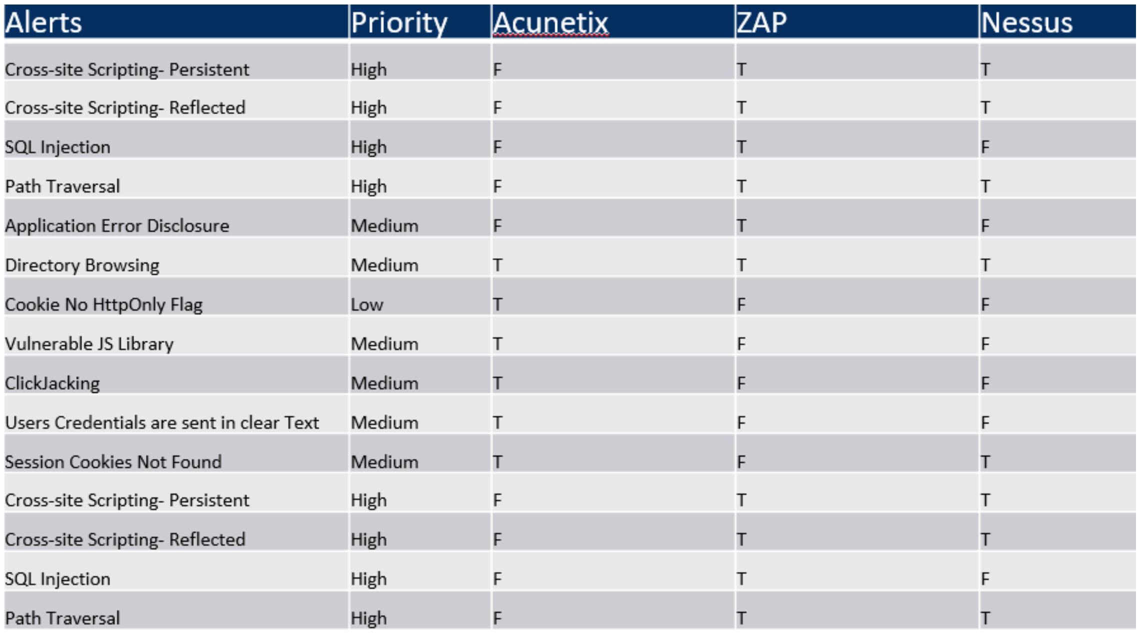
Figure 5.1 shows the vulnerability correctness for each tool results that is verified from manual testing of our Property app Application. It shows all the vulnerabilities which was verified manually by the testing team and is presented in a matrix comparing them with the results from tools. For example, if we see the figure 5.1, vulnerability blind sql injection was detected by Acunetix but not by ZAP and Nessus Pro. We use this data to calculate vulnerability correctness for each tool and then use this percentage score to evaluate these tools.



Alerts	Priority	Acunetix	ZAP	Nessus
Blind SQL Injection	High	T	F	F
Cross Site Scripting	High	T	F	F
SQL Injection	High	T	T	T
PHP Vulnerabilities	High	F	F	T
ClickJacking	Low	T	F	T
Trace Method Enabled	Low	T	F	T
Login Password-Guessing attack	Low	T	F	F
Cookie without Secure Flag/HttpOnly flag	Low	T	T	T
Incomplete Cache Control	Low	F	T	F
Application Error Message	Medium	T	T	T
Directory Listing	Medium	T	F	T
Errors on the page	Medium	T	F	T
HTML form without CSRF protection	Medium	T	F	T
User Credentials are sent in Clear Text	Medium	T	F	F
Vulnerable JS Library	Medium	T	T	F
OS Command Injection	Medium	F	T	F
Stack Buffer Overflow	Medium	F	F	T
X-Content Type- Options Header Missing	Low	F	T	F

Figure 5.1: Vulnerability Correctness for IRS Application

Similarly, Figure 5.2 shows the vulnerability correctness for each tool results for our XPDOffice Application. It also shows the vulnerabilities which was verified from manual testing and each tool performance for the specific vulnerabilities. Table 5.1 shows vulnerability correctness score for each tool against the tested applications. The score is calculated in percentage and shows the correctness of the tool.



Alerts	Priority	Acunetix	ZAP	Nessus
Cross-site Scripting- Persistent	High	F	T	T
Cross-site Scripting- Reflected	High	F	T	T
SQL Injection	High	F	T	F
Path Traversal	High	F	T	T
Application Error Disclosure	Medium	F	T	F
Directory Browsing	Medium	T	T	T
Cookie No HttpOnly Flag	Low	T	F	F
Vulnerable JS Library	Medium	T	F	F
ClickJacking	Medium	T	F	F
Users Credentials are sent in clear Text	Medium	T	F	F
Session Cookies Not Found	Medium	T	F	T
Cross-site Scripting- Persistent	High	F	T	T
Cross-site Scripting- Reflected	High	F	T	T
SQL Injection	High	F	T	F
Path Traversal	High	F	T	T

Figure 5.2: Vulnerability Correctness for XPD Application

Vulnerability Correctness				
Tool	Application	Total alerts(manual testing)	No. Of Vulnerabilities Matched	Vulnerability Correctness(percentage)
Acunetix	IRS	18	13	72.2
Nessus Pro	IRS	18	10	55.55
ZAP	IRS	18	7	38.88
Acunetix	XPDOffice	11	6	54.54
Nessus Pro	XPDOffice	11	5	45.45
ZAP	XPD	11	6	54.54

Table 5.1: Vulnerability Correctness for each tool

5.2.2 Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a model or a testing on the test data. It has four important terminologies.

- True Positives: What is predicted is actually present
- True Negatives: What is not predicted and is not present
- False Positives: What is predicted but not actually present
- False Negatives: What is not predicted but present

We can use a confusion matrix to generate two scores Precision and Recall for our tools. As, we have set of vulnerabilities which was verified by the manual testing. We can use this data to generate two more scores to effectively evaluate our tools.

In our case, true positives would be vulnerabilities which were found by the tools and which was verified by manual testing. Similarly, false positives will be one which was detected by tool but was not present in the application. Also, False Negative are the vulnerabilities which were there in the application but was not detected by the tools. So, if we create confusion matrix for each tool results when compared with actual results, we can calculate Precision and Recall.

		Prediction outcome		
		p	**n**	**total**
actual value	**p′**	True Positive	False Negative	P′
	n′	False Positive	True Negative	N′
	total	P	N	

Table 5.2: Confusion Matrix

- Precision: It is fraction of relative instances among the retrieved instances. Also called positive predictive value. Precision could be defined as measure of quality.

- Recall: It is fraction of relative instances that have been retrieved over the total amount of relevant instances. It can also be termed as relevance and can be defined as measurement of completeness.[26].

Both these values can be calculated from confusion matrix which can be used to evaluate our tools.

$$\begin{aligned} Precision &= (TotalPositive/(TotalPositive + FalsePositive)) \\ Recall &= (TotalPositive/(TotalPositive + FalseNegative)) \end{aligned}$$

We created a confusion matrix for all the results which we got by testing our tools on IRS application and XPDOffice. Based on their false positive and total positive, we calculated Precision and Recall for each tools and then used them to evaluate the

Tools Test Results				
Tools	Application	True Positive	False Positive	False Negative
Acunetix	XPDOffice	6	3	5
Nessus Pro	XPDOffice	5	1	6
ZAP	XPDOffice	6	2	5
Acunetix	IRS	13	1	5
NessusPro	IRS	10	1	8
ZAP	IRS	7	3	11

Table 5.3: Tools Testing Results

tools. Table 5.2.2 shows the test results for each tool on IRS and XPDOffice. Table shows True positives and true negative for each application found by comparing the vulnerabilities with the one shown in figure 5.1 and 5.2. Using these data, we can create confusion matrix for each tool and then calculate their Precision and Recall.

Acunetix on IRS	
13(tp)	5(fn)
1(fp)	0(tn)

Table 5.4: Confusion Matrix for Acunetix on IRS

We created confusion matrix for each tools on the basis of the data in Table 5.3 and used the data to calculate Precision and Recall. Table 5.10 shows the Precision and Recall for all the results.

ZAP on IRS	
7(tp)	11(fn)
3(fp)	0(tn)

Table 5.5: Confusion Matrix for ZAP on IRS

Nessus on IRS	
10(tp)	8(fn)
1(fp)	0(tn)

Table 5.6: Confusion Matrix for Nessus on IRS

Acunetix on XPD	
6(tp)	5(fn)
3(fp)	0(tn)

Table 5.7: Confusion Matrix for Acunetix on XPDOffice

5.3 Tools Functional Evaluation

We used several parameters to evaluate functional side of the tools and how easy it is to configure them and scan the web applications. Each tool has its own interface and provides different functionalities to scan the web application. Some tools like Acunetix also provides the functionality to customize the scanning template for specific development environment. Some tools like Nessus and Acunetix gives you the option to upload a sequence which can be tested to find vulnerabilities.PFollowing discusses each tools with different parameters.

- Speed: This is the total amount of time taken to scan the web application. Number of requests generated by the tool is directly proportional to the scan time. If we compare the scanning time of all the tools tested on all three web applications. Nessus Pro is a suite with many scanners available to scan

ZAP on XPD	
6(tp)	5(fn)
2(fp)	0(tn)

Table 5.8: Confusion Matrix for ZAP on XPDOffice

Nessus Pro on XPD	
5(tp)	6(fn)
1(fp)	0(tn)

Table 5.9: Confusion Matrix for ZAP on XPDOffice

Tools Test Results			
Tools	Application	Precision	Recall
Acunetix	XPDOffice	66.66	54.54
Nessus Pro	XPDOffice	83.3	54.54
ZAP	XPDOffice	75	54.54
Acunetix	IRS	92.8	72.2
NessusPro	IRS	90.9	55.5
ZAP	IRS	70	38.88

Table 5.10: Tools Testing Results

the application. Acunetix is very fast compared to other tools as it is multi-threaded and scans the target very fast.

- Prerequisite Knowledge and Expertise: Both ZAP and Nessus need prior knowledge about HTTP requests and authentication. To use these two tools effectively, prior knowledge of web authentication is required. Acunetix directly gives you the option to scan the application and hence it can be used by new developers to scan their application. Nessus Pro configuration is very complex and you need to understand the site map of the application before scanning it. It needs login page details, landing page and form data to configure the scan, which sometimes becomes difficult. ZAP is a open-source tool and hence there are many tutorials which can be found on the web and can be used by the tester and developers.

- Interface-Ease to use: Acunetix has the best interface, which provides you all the options to configure the target and scan the application. Its configuration is very easy and doesn't require deep understanding of the application. ZAP's interface is also complex, but it gives the option where it acts as a proxy and can record all browser activities. After that, Spider and active scan can detect vulnerabilities in the application. Nessus Pro's interface is very complex and needs deep understanding of the tools before using it.

- Specialization and Scan Template: All three tools provides a default template which is used for scanning the applications. Some tools like ZAP gives you the option to customize the scan template according to the vulnerabilities. Nessus Pro uses a predefined template to scan the web applications. It first scans the server and then starts scanning the web application. As said earlier, it becomes difficult to configure the template in Nessus. Acuentix also has a default template, but it has several other scan types like High risk vulnerabilities, SQL injection, and cross-site scripting. According to the need and design of the web application, scans can be configured in Acunetix.

- Predefined Settings for development environment: Only Acunetix provides the functionality to configure the scan according to the development environment of the web application. It has predefined scripts and settings for the applications developed in Java, PHP, ASP, Node,js and many more.

- Report: All the tools provides an interface to generate the report for the scans. Both Nessus Pro and Acunetix provides very deep and understandable report which can be used by the developers and tester to detect the critical areas in the web application. ZAP also provides you the option to generate report, but its report lacks depth. It also provides complete URLs which it requested to the web application to detect the vulnerabilities.

- Cost: OWASP ZAP is a open source tool, but its vulnerability correctness percentage is highly appreciable compared to other tools. Acunetix is very expensive compared to Nessus Pro. Its cost is around 8000 dollars for ten targets whereas Nessus Pro comes at 2190 dollars per year. Nessus Pro, not only provides a web application scanner, but also provides a whole suite of scanners which can be used to scan other components. Acunetix focuses only on Web application scanning.

- Vulnerability Correctness: Table 5.1 shows the vulnerability correctness for each test that the tool performed on two applications. It can be seen that Accunetix and Nessus Pro has high vulnerability correctness percentage for all the tests. ZAP's performance and detection rate is highly appreciable for a open source tool. If we compare all the vulnerability correctness values, Nessus Pro and Acunetix has high correctness value compared to ZAP.

- Precision and Recall: We created confusion matrix for all the tests and calculated precision and recall for the same. A high precision means that the test returned substantially more relevant results, while high recall means that the test returned most of the relevant results. Precision and Recall values for the tests performed by the tools are shown in Table 5.10. As you can see, Acunetix and Nessus Pro have high precision which means that they were able to find most of the vulnerabilities present in the application. ZAP's precision is also very high and is appreciable if we consider that ZAP is a open source tool. All tools have almost same Recall value which means they found most of the correct vulnerabilities and number of false positives were less for all the tools.

5.4 Conclusion

If we evaluate all the results and analyze them, all tools were effective to find some of the vulnerabilities present in our web applications. Both commercial tools Nessus Pro and Acunetix has ability to do a comprehensive scan of the application. They were able to detect some of the known vulnerabilities like SQL injection and Cross-site scripting, which exposes serious security loopholes in the application. If we consider the development environment of the web applications, only Acunetix has the functionality to customize the scan according to the environment. Acunetix's interface and ease to use stands among all the tools as it provides a easy interface and the response time is also fast. ZAP acts as a proxy between browser and web application and hence was able to test all urls in the application. It also uses Spider to find the links which browser has not visited and then do complete scan to find out the vulnerabilities. Acunetix generates a site map of the application which it is scanning and then uses its scripts to test the application. For a complete black-box testing of a web application, this research recommends Acunetix commercial version which can do a comprehensive scan on the web application and can present security loopholes to the developer and testers. Its interface also is easy to use and can be configured easily. The report generation in Acunetix is also deep and well explained and shows the attack details. It also provides functionality to generate reports using different templates according to the user of the application. Like executive,developer, OWASP Top 10 and many more. Hence, for a comprehensive scan of a web application, which can detect all known and unknown vulnerabilities, Nessus Pro is recommended. It is a complete package to both developers and testers. Nessus pro does a comprehensive scan of the network and web servers but lacks in overall testing of web application. It doesn't create site map as Acunetix does and hence sometimes misses some part of the web application which might have vulnerabilities. Acunetix creates a site map of the web application and crawls through all the links to find vulnerabilities and hence is best suited for a web application scanner among these popular web scanners.

If we consider vulnerability correctness value for ZAP, its performance is highly appreciable if we consider that ZAP is an open source tool. So, an organization who does not want to spend money on buying commercial tools for scanning their web applications, ZAP is recommended.

Acunetix is suited for organizations who would like to use it on a continuous basis and wants to focus only on Web application scanning. If we compare the results of both the commercial tools which we used in our research, their performance is almost similar. Hence, it depends on application structure and web application components to select a tool which is best suited for their security purpose.

Chapter 5

Future Work

5.1 Overview

In the recent years, a lot of web applications have been released in the world. People all around the world uses web applications for their daily work like banking, social media and many more. This increase also brings lots of security issues in the web application which can be exploited by the hackers. Hence, it becomes imperative and significant for developers and enterprise to secure their web applications before hackers can exploit them. However, testing for vulnerabilities manually is very tedious and time consuming and sometimes human hand misses some of the known vulnerabilities while testing the application. Hence, there is a need of web application scanner. Companies around the world uses web application scanners in their Software Development Life cycle to detect vulnerabilities and remove them. Hence, the market for web scanners has increased a lot. There are many tools in the market which provides the option to testers and developers to scan their application and check for security loopholes. In this research, we evaluated top three tools on three web applications; developed in three different development environment and evaluated the tools. We used Kali Linux to host the tools and a window machine to host the applications. We scanned all the applications from different tools and analyzed their results. We compared them on different parameters like Ease to use, interface, response time, number of alerts and total number of requests. Then, we came up with a conclusion for a tool which would be well suited for a black box testing of a web application.

In future, we plan to include more tools for our evaluation so as to compare them more effectively. Also, we tested our tools on applications which have unknown vulnerabilities. But, in the future, we can scan our tools on applications which have known vulnerabilities and then compare them with others. This result along with our research can evaluate the tools efficiently.

Two web applications out of three which we used in this research were hosted on local machine and one was a live website. In the future, we can use applications which are hosted on cloud server like Amazon and then scan those applications to evaluate these tools. Once, the applications are hosted on QA system which emulates production system, it becomes more efficient to test them as it will be able to detect real-world vulnerabilities and can help developer and testers to design their system free of security loopholes.

References

[1] 10 benefits of web based applications and systems, Sep 2015.

[2] Robert Abela. Getting started with web application security, Dec 2017.

[3] Durai Amuthan.h. What is the difference between application server and web server?, Jun 2014.

[4] Lauri Auronen. Tool-based approach to assessing web application security. *Helsinki University of Technology*, 11:12–13, 2002.

[5] Mark Curphey and Rudolph Arawo. Web application security assessment tools. *IEEE Security & Privacy*, 4(4):32–41, 2006.

[6] Nor Izyani Daud, Khairul Azmi Abu Bakar, and Mohd Shafeq Md Hasan. A case study on web application vulnerability scanning tools. In *Science and Information Conference (SAI), 2014*, pages 595–600. IEEE, 2014.

[7] Roy T Fielding and Richard N Taylor. Principled design of the modern web architecture. *ACM Transactions on Internet Technology (TOIT)*, 2(2):115–150, 2002.

[8] Jose Fonesca. Testing and comparing web vulnerability scanning tools for sql injection and xss attacks. page 1.

[9] Stefanos Gritzalis. Addressing threats and security issues in world wide web technology, 1997.

[10] Raphael Hertzog, Jim OGorman, and Mati Aharoni. *Kali Linux Revealed: mastering the penetration testing distribution*. Offsec Press, 2017.

[11] Yuma Makino and Vitaly Klyuev. Evaluation of web vulnerability scanners. In *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 2015 IEEE 8th International Conference on*, volume 1, pages 399–402. IEEE, 2015.

[12] J.D Meir, Alex Homer, and David Hill. Web application architecture guide. *Web Application Architecture Guide*, page 618, 2008.

[13] Daniel Nations. What exactly is a web application?, Feb 2018.

[14] NJ Ouch. 2013 top security tools as voted by toolswatch.org readers, Dec 2013.

[15] Top OWASP. Application security risks-2017, open web application security project (owasp), 10.

[16] Andrey Petukhov and Dmitry Kozlov. Detecting security vulnerabilities in web applications using dynamic analysis with penetration testing. *Detecting Security Vulnerabilities in Web Applications Using Dynamic Analysis with Penetration Testing*, page 116.

[17] Kristof Phillipsen. Web application vulnerability assessment, Jul 2015.

[18] Sajjad Rafique, Mamoona Humayun, Zartasha Gul, Ansar Abbas, and Hasan Javed. Systematic review of web application security vulnerabilities detection methods. *Journal of Computer and Communications*, 3(09):28, 2015.

[19] Abdul Rahman. Analysis of web application security. *IEEE*, page 35, May 2017.

[20] Ray Ranker. Home.

[21] Khairul Anwar Sedek, Osman Norlis, Mohd Osman, and Kamaruzaman Jusoff. Developing a secure web application using owasp guidelines. 2, 10 2009.

[22] Angela Stringfellow. What is web application architecture? how it works, trends, best practices and more, Sep 2017.

[23] Tenable. Tenable network security — usa 2016.

[24] Usman Waheed. Security regression testing framework for web application development. Master's thesis, 2014.

[25] Wikipedia. Client (computing), Jun 2018.

[26] Wikipedia. Precision and recall, Jul 2018.

[27] Wikipedia. Web server, May 2018.