Rational Unified Process based Model to Fetch the Vulnerabilities in Effective Way

Deven Gol[1] Priyank Bhojak[2] Dhaval Patel[3]
[1,2,3]Assistant Professor
[1,2,3]Department of Information Technology
[1]GCET College, GTU V. V. Nagar, Gujarat 388120, India [2]BVM College, GTU V. V. Nagar, Gujarat 388120, India [3]INDUS University, Ahmedabad, Gujarat 382115, India

Abstract— We are in the era where a person needs to interact with Web applications day to day life. As the use of internet growing like online shopping, online transaction, and so on. So that we need to think about the Security of Web Application due to vulnerabilities generally found. There are number of tools available in market but still some of the tools can't identify all attacks properly. Because of these Vulnerabilities, an attacker can easily enter into the system by unauthorized access, malicious activity; damage the system which can have an effect on the system. Vulnerability Scanner is a tool that will help to identify the weak hole of the system which makes entry point of Hacker. If we can identify suck kind of vulnerabilities it will help user from an access of sensitive data. Most of the web application security vulnerabilities occur just because of nonspecific input validation problems. Even if a large number of web application vulnerabilities are unproblematic to distinguish and to avoid, a lot of web developers are unfortunately not security-aware that make easy access for attackers. As a effect, here be present a huge amount of susceptible web applications and websites on the web which is common for all people. Security vulnerabilities are the significance of violating security properties which harm the systems direct or indirect way.

Key words: SQL Injection, Broken Authentication, HTTP Banner Disclosure, XSS, Crawler, Web Application Vulnerability (WAV), Rational Unified Process (RUP)

I. Introduction

As Internet or surfing has become one of the mainly common communication medium in the digital era. Many number of users connect to different way for web-based applications to find the particular information, they exchange messages, even interact with each one, and several more. As the era is all about web technology, the web application is flattering additional and admired and has turn out to be an imperative component of our daily lives. It is crucial function of web application as it has becoming more dangerous now days. Ensuring privacy, reliability the accessibility of the data and services presented through these portals rely on the security measures built in to the web applications. Whenever security is violating, it implies either huge financial, public relation or social impact for the organizations involved. Our approach to this project is to develop our own web application tool rules out the possibility that the vulnerability scanners are trained for or otherwise already know of the specific implementation of vulnerabilities available. Since we are spotlight on automated penetration testing tool such as a framework for web application, we will avoid doing to teach the tools their way around the application. Instead we will let them rely on their incorporated crawling engines, which previous research have established are reasonably consistent. The natures in some of these tools require more communication than others, but we use the automated crawling Approach which will help to save all pages and scanning functionality.

II. Types of Attacks

A. SQL Injection Attack

SQL injection is attack kind of attack which aligned with a database-driven web application. SQL injection attacks are applied throughout performing unacceptable input characters, special symbols or strings which is interested in the database of the system that is used to transform it for their intentional exploit [11]. It could be promising if a web application does not endow with appropriately provided filtering for the different user inputs. SQL injection can also be achievable through inserting novel keywords, Characters, dissimilar arrangement of operators or special symbols and strings into the original Structure query language. If the attacker is successfully enter in to system then the attacker will unswervingly bypass an original SQL statement and get chance to perform attack within code which will provide access to the back-end of database of the vulnerable i.e. harmed application. Sometimes it may be happened that conditions may be interconnect or correlate with the whole file system of full database with full of opportunity and even he can perform different system calls which act as a valid or legitimate user. It can be possible user can inject a vulnerable or malicious input string into filed of user's SQL statements. This is later on used to compile the SQL based query. A web application is also a vulnerable or malicious to an SQL based injection input to perform SQLiA attack which can be provide access to an attacker. This is very much capable to run or compile the SQL based statements within an accessible database or tables of the web based applications.

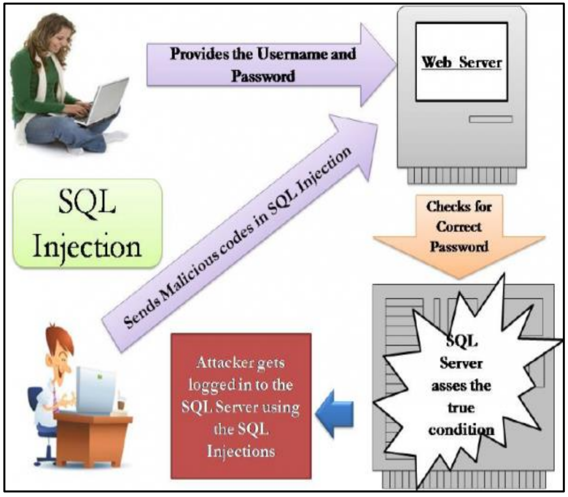

Provides the Username and Password
Web Server
SQL Injection
Sends Malicious codes in SQL Injection
Checks for Correct Password
SQL Server asses the true condition
Attacker gets logged in to the SQL Server using the SQL Injections

Fig. 1: Scenario of SQLIA Attack [19]

B. Broken Authentication & Session Management

There is an application which are interrelated to authentication and session management frequently not implemented appropriately, It may be provide access to attackers to compromise with the password, secret keys, or session Ids or tokens, or to utilize other accomplishment flaws to presume additional users' identities. Broken Authentication and Session Management are exceptionally common in web applications, as reported by OWASP in 2010, reaching the second place in 2013, when it was considered the largest part of common programming error in web based applications.

C. Cross Site Scripting (XSS)

XSS attacks, as they are known, allow attackers to implement malicious Java Script code, pretending that the application is transferring the code to the web user. When a website is vulnerable to cross site scripting XSS then an attacker is able to execute any malicious scripts within the victim's browser which will be capable of used to session hijack to get full access to user's sessions, even attacker can spoil web sites or possibly introduce worms, Trojans, backdoors among others. It was considered the most common programming error second place in web applications in 2010, and the third most common in 2013.

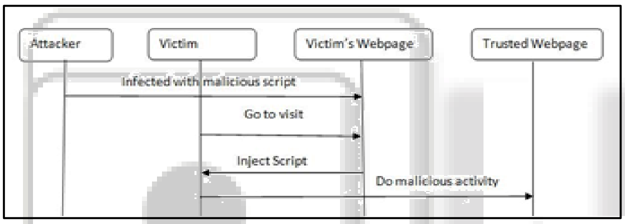


Fig. 2: Sequence View of XSS Attack [1]

III. VULNERABILITY DETECTION

A. Penetration Tests

A Penetration Test, frequently referred as pen test, estimate the protection of applications by simulating malicious attacks.

It is different from other types of auditing in the intelligence that penetration tester takes the attackers point of view, by means of only incomplete knowledge regarding the internal workings of an applications[1].

This is different than, for example, a code review where the auditor has access to all aspects of the application.

Penetration tests frequently involve a important amount of human intervention, but often assisted by different types of tools.

B. Penetration Testing Tools

These are software developed penetration testing tools to assist professionals during a penetration test.

This could either represent tools that accomplished unambiguous automated tasks or completely mechanized "point- and-shoot" solutions, without individual involvement, crawls the functionality in an application sooner than trying to identify vulnerabilities in web[1].

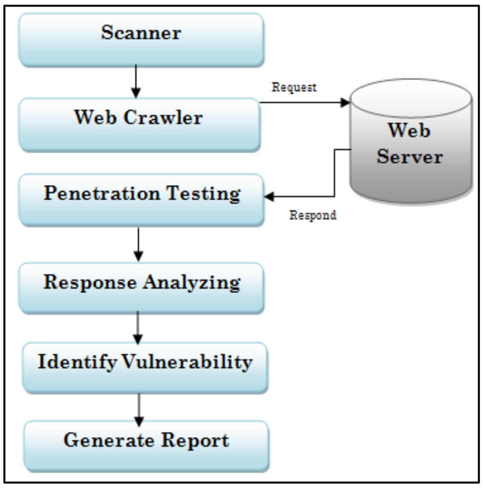


Fig. 3: Phase of Proposed Model

Revealing engine is generally used to afford the web demand all the way through various individual attacking based code otherwise different kinds of scripts.

Revealing engine also hangs around for the reaction receiving commencing the web server that can be analyzed later on and if it will discover the particular or specific pattern of vulnerable script or malicious script in the responded or result of data, the malicious vulnerability is acknowledged fruitfully.

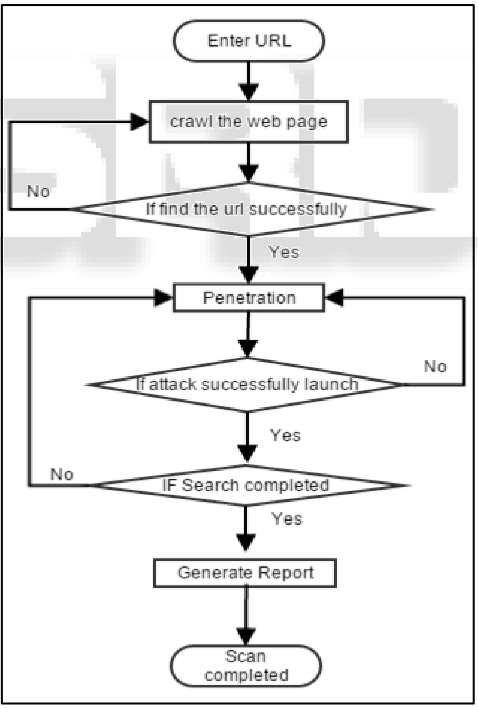


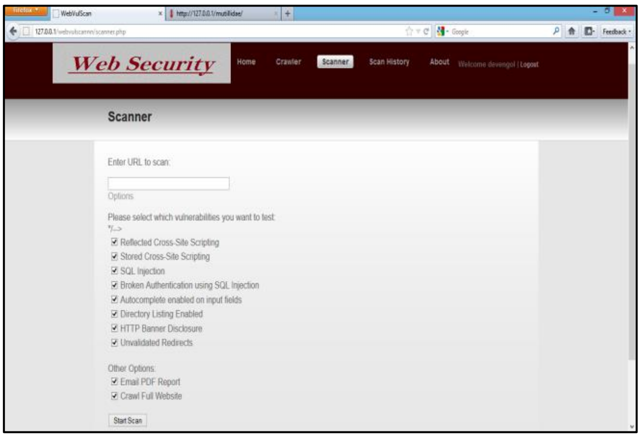
Fig. 4: Flowchart of the System

IV. EXPERIMENTAL SETUP & RESULTS

The essential requirement of this project is it was developed on windows 7 OS based computer. We have been using the PHP for this vulnerable tool for identification. We have used Dreamweaver 8 as a front end tool. And we have used MySQL and Wamp server as a back end [1].

Figure 5 give you an idea about the outcome of decision that will endow with dissimilar attacks phase.

.


WebVulScan
http://127.0.0.1/mutillidae/
127.0.0.1/webvulscanner/scanner.php
Google
Feedback
Web Security
Home
Crawler
Scanner
Scan History
About
Welcome devengol | Logout
Start Scan

Scanner

Enter URL to scan:

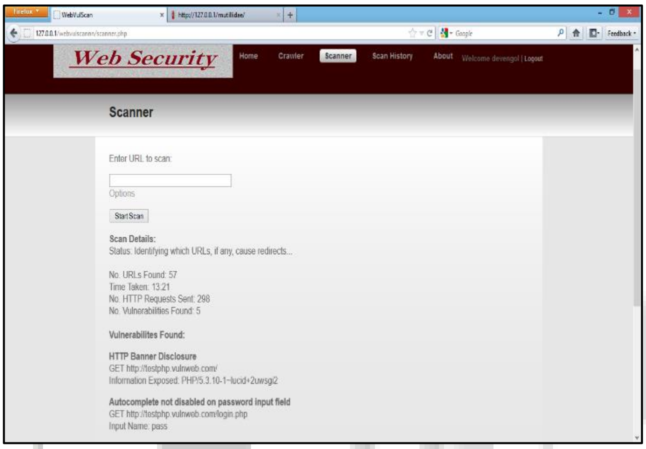
Options

Please select which vulnerabilities you want to test:
"/-->
☑ Reflected Cross-Site Scripting
☑ Stored Cross-Site Scripting
☑ SQL Injection
☑ Broken Authentication using SQL Injection
☑ Autocomplete enabled on input fields
☑ Directory Listing Enabled
☑ HTTP Banner Disclosure
☑ Unvalidated Redirects

Other Options:
☑ Email PDF Report
☑ Crawl Full Website

Fig. 5: List of Attacks

Third Component is Analysis component examine the results returned by the web applications to validate whether an attack was successful.


WebVulScan
http://127.0.0.1/mutillidae/
127.0.0.1/webvulscanner/scanner.php
Google
Feedback
Web Security
Home
Crawler
Scanner
Scan History
About
Welcome devengol | Logout
Start Scan

Scanner

Enter URL to scan:

Options

Scan Details:
Status: Identifying which URLs, if any, cause redirects...

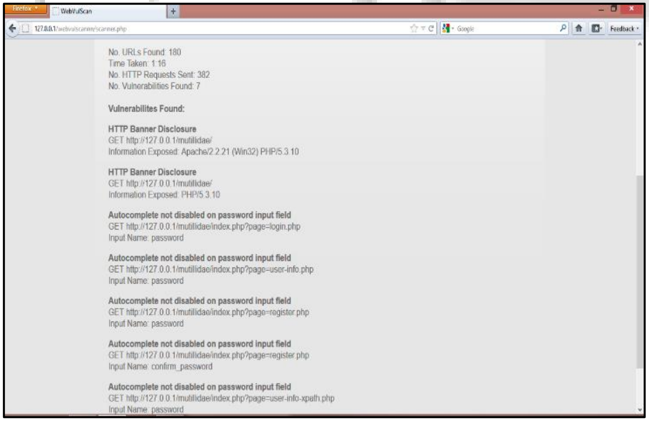
No. URLs Found: 57
Time Taken: 13:21
No. HTTP Requests Sent: 298
No. Vulnerabilities Found: 5

Vulnerabilites Found:

HTTP Banner Disclosure
GET http://testphp.vulnweb.com/
Information Exposed: PHP/5.3.10-1~lucid+2uwsgi2

Autocomplete not disabled on password input field
GET http://testphp.vulnweb.com/login.php
Input Name: pass

Fig. 6: After Analysis of Scan Test



No. URLs Found: 180
Time Taken: 1:16
No. HTTP Requests Sent: 382
No. Vulnerabilities Found: 7

Vulnerabilites Found:

HTTP Banner Disclosure
GET http://127.0.0.1/mutillidae/
Information Exposed: Apache/2.2.21 (Win32) PHP/5.3.10

HTTP Banner Disclosure
GET http://127.0.0.1/mutillidae/
Information Exposed: PHP/5.3.10

Autocomplete not disabled on password input field
GET http://127.0.0.1/mutillidae/index.php?page=login.php
Input Name: password

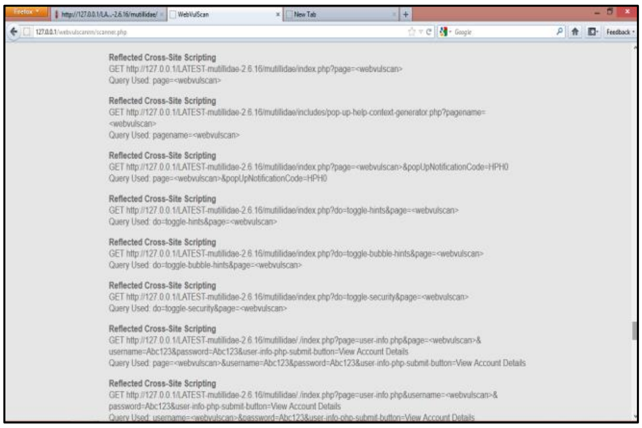
Autocomplete not disabled on password input field
GET http://127.0.0.1/mutillidae/index.php?page=user-info.php
Input Name: password

Autocomplete not disabled on password input field
GET http://127.0.0.1/mutillidae/index.php?page=register.php
Input Name: password

Autocomplete not disabled on password input field
GET http://127.0.0.1/mutillidae/index.php?page=register.php
Input Name: confirm_password

Autocomplete not disabled on password input field
GET http://127.0.0.1/mutillidae/index.php?page=user-info-xpath.php
Input Name: password

Fig. 7: Scanner Module for HTTP Banner Disclosure


http://127.0.0.1/LA...-2.6.16/mutillidae/
WebVulScan
New Tab
127.0.0.1/webvulscanner/scanner.php
Google
Feedback

Reflected Cross-Site Scripting
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/index.php?page=<webvulscan>
Query Used: page=<webvulscan>

Reflected Cross-Site Scripting
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/includes/pop-up-help-context-generator.php?pagename=<webvulscan>
Query Used: pagename=<webvulscan>

Reflected Cross-Site Scripting
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/index.php?page=<webvulscan>&popUpNotificationCode=HPH0
Query Used: page=<webvulscan>&popUpNotificationCode=HPH0

Reflected Cross-Site Scripting
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/index.php?do=toggle-hints&page=<webvulscan>
Query Used: do=toggle-hints&page=<webvulscan>

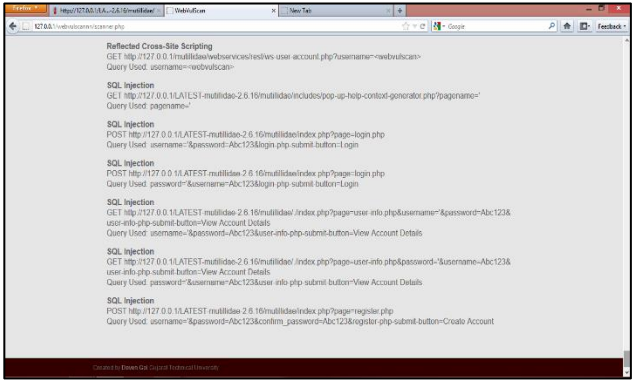
Reflected Cross-Site Scripting
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/index.php?do=toggle-bubble-hints&page=<webvulscan>
Query Used: do=toggle-bubble-hints&page=<webvulscan>

Reflected Cross-Site Scripting
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/index.php?do=toggle-security&page=<webvulscan>
Query Used: do=toggle-security&page=<webvulscan>

Reflected Cross-Site Scripting
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/ /index.php?page=user-info.php&page=<webvulscan>&username=Abc123&password=Abc123&user-info-php-submit-button=View Account Details
Query Used: page=<webvulscan>&username=Abc123&password=Abc123&user-info-php-submit-button=View Account Details

Reflected Cross-Site Scripting
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/ /index.php?page=user-info.php&username=<webvulscan>&password=Abc123&user-info-php-submit-button=View Account Details
Query Used: username=<webvulscan>&password=Abc123&user-info-php-submit-button=View Account Details

Fig. 8: Scanner Module for XSS


http://127.0.0.1/LA...-2.6.16/mutillidae/
WebVulScan
New Tab
127.0.0.1/webvulscanner/scanner.php
Google
Feedback
Created by Deven Gol Gujarat Technical University

Reflected Cross-Site Scripting
GET http://127.0.0.1/mutillidae/webservices/rest/ws-user-account.php?username=<webvulscan>
Query Used: username=<webvulscan>

SQL Injection
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/includes/pop-up-help-context-generator.php?pagename='
Query Used: pagename='

SQL Injection
POST http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/index.php?page=login.php
Query Used: username='&password=Abc123&login-php-submit-button=Login

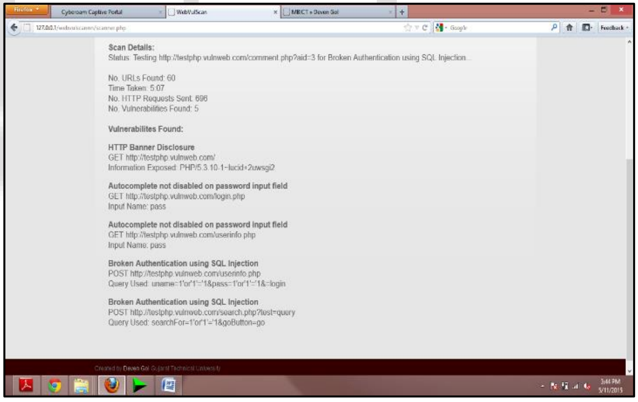
SQL Injection
POST http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/index.php?page=login.php
Query Used: password='&username=Abc123&login-php-submit-button=Login

SQL Injection
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/ /index.php?page=user-info.php&username='&password=Abc123&user-info-php-submit-button=View Account Details
Query Used: username='&password=Abc123&user-info-php-submit-button=View Account Details

SQL Injection
GET http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/ /index.php?page=user-info.php&password='&username=Abc123&user-info-php-submit-button=View Account Details
Query Used: password='&username=Abc123&user-info-php-submit-button=View Account Details

SQL Injection
POST http://127.0.0.1/LATEST-mutillidae-2.6.16/mutillidae/index.php?page=register.php
Query Used: username='&password=Abc123&confirm_password=Abc123&register-php-submit-button=Create Account

Fig. 9: Scanner Module for SQL Injection


Cyberoam Captive Portal
WebVulScan
MBICT » Deven Gol
127.0.0.1/webvulscanner/scanner.php
Google
Feedback
Created by Deven Gol Gujarat Technical University
3:44 PM
5/11/2015

Scan Details:
Status: Testing http://testphp.vulnweb.com/comment.php?aid=3 for Broken Authentication using SQL Injection...

No. URLs Found: 60
Time Taken: 5:07
No. HTTP Requests Sent: 696
No. Vulnerabilities Found: 5

Vulnerabilites Found:

HTTP Banner Disclosure
GET http://testphp.vulnweb.com/
Information Exposed: PHP/5.3.10-1~lucid+2uwsgi2

Autocomplete not disabled on password input field
GET http://testphp.vulnweb.com/login.php
Input Name: pass

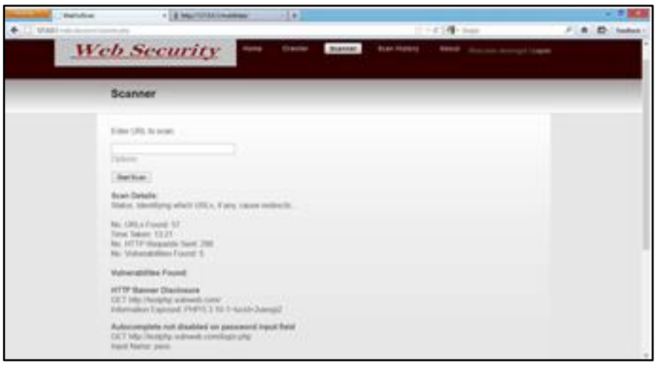
Autocomplete not disabled on password input field
GET http://testphp.vulnweb.com/userinfo.php
Input Name: pass

Broken Authentication using SQL Injection
POST http://testphp.vulnweb.com/userinfo.php
Query Used: uname=1'or'1'='1&pass=1'or'1'='1&=login

Broken Authentication using SQL Injection
POST http://testphp.vulnweb.com/search.php?test=query
Query Used: searchFor=1'or'1'='1&goButton=go

Fig. 10: Scanner Module for Broken Authentication

After enticing the URL based Module of our model Attack module which will launches the predefined or configured vulnerable attacks. It is in opposition to these provided targets which is collected web sites by the crawler component. It is to the fore to examination module that is used monitoring the outcome revisited by the web based applications for the validation of an attack whether it was successful or not. Figure 10 illustrate the evolution of Discovery or identification phase in our web based application device.



Scanner

Enter URL to scan:

Options

Start Scan

Scan Details:
Status: Identifying which URLs, if any, cause redirects...

No. URLs Found: 57
Time Taken: 13:21
No. HTTP Requests Sent: 288
No. Vulnerabilities Found: 5

Vulnerabilities Found

HTTP Banner Disclosure
GET http://testphp.vulnweb.com/
Information Exposed: PHP/5.3.10-1~lucid+2uwsgi2

Autocomplete not disabled on password input field
GET http://testphp.vulnweb.com/login.php
Input Name: pass

Fig. 11: Scanner Result for HTTP Banner Disclosure

		Our Tool		Old Approach
Site	Vulnerabilities	Detected		
		OWASP	Testphp	Vulnerable
Vulnerable Site	Reflected Cross-Site Scripting	29	3	4
	SQL Injection	6	2	3
	Broken Authentication using SQL Injection	-	3	-
	Auto complete Enabled on Password Fields	12	2	-
	Directory Listing Enabled	12	4	-
	HTTP Banner Disclosure	2	1	-
	Unvalidated Redirects	71	2	-
	Cross site Request Forgery	-	-	4
	Xpath Injection	-	-	2

Table 1: Analysis on the Research Done by Previous Tool with Our Proposed Solution

Here in table 1 we specified some results that we found compare to the existing tool. This is far better than previous approach and better solution for web application security.

V. Conclusion & Future Work

The major involvement of this research is to give you an idea about how easily we can do robotically determine and make the most of web application- level vulnerabilities in a huge amount of web applications. Lots of the web application safety measures vulnerabilities result from nonspecific input validation problems. Examples of such vulnerabilities are Different attacks already shown in results. Even though for the most part of web vulnerabilities are trouble-free to recognize and evade, many web developers are unfortunately not security-aware and there is common consensus that there will be present a large quantity of vulnerable applications and web sites on the web. To this end, we apply a generic web vulnerability scanner that analyzes web sites for exploitable most vulnerable attacks for web application vulnerabilities. There are so many web application vulnerabilities consequence from nonspecific input validation. So we have developed a tool which is used to recognize the Web application vulnerabilities which will be based on the RUP (Rational Unified Process) [1] Model. We have provided an enhanced approach to construct vulnerability recognition in appropriate way.

References

[1] D. Gol and N. Shah, "Detection of web appication vulnerability based on RUP model," 2015 National Conference on Recent Advances in Electronics & Computer Engineering (RAECE), Roorkee, 2015, pp. 96-100. doi: 10.1109/RAECE.2015.7510233

[2] http://www.vpnfaqs.com/2015/06/deep-sql-injection

[3] Priya. R. L1, Lifna. C. S, “Rational Unified Treatment for Web Application Vulnerability Assessment “©2014 IEEE.

[4] Guowei Dong1, YanZhang2, Xin Wang1, Peng Wang2, Liangkun Liu2, “Detecting Cross Site Scripting Vulnerabilities Introduced by HTML5”, Renmin University of China, China ©2014 JCSSE.

[5] Yuan-Hsin Tung, Chen-Chiu Lin, Hwai-Ling Shan Telecommunication Lab.,Chunghwa Telecom Co., Ltd., Taiwan,ROC , “Test as a Service: A framework for Web security TaaS service in cloud environment ”, Beijing University of Posts and Telecommunications, Beijing, China. 978-1-4244-6769-3/10/$26.00 ©2014 IEEE,p.- 14-18.

[6] Yuan-Hsin Tung,23Shian-Shyong Tseng , 1Jen-Feng Shihl, 1Hwai-Ling Shanl 1Telecommunication Lab., Chunghwa Telecom Co., Ltd., Taiwan,ROC , “A Cost-Effective Approach to evaluating Security Vulnerability Scanner ”, Beijing University of Posts and Telecommunications, Beijing, China. © IEICE, 2013.

[7] Xin Gopal R. Chaudhari, Prof. Madhav V. Vaidya Department of Information Technology,SGGS IE & T, Nanded, Maharashtra, “A Survey on Security and Vulnerabilities of Web Application ”, ©2014 IJCSIT.

[8] Rajesh M. Lomte1, Prof. S. A. Bhura2 Computer Science & Engineering Department, BNCOE, India, “A Survey on Security and Vulnerabilities of Web Application IOSR-JCE, 2013.

[9] Xin Wang, Luhua Wang, Gengyu Wei, Dongmei Zhang and Yixian Yang, “Hidden Web Crawling For Sql Injection Detection ”, Beijing University of Posts and Telecommunications, Beijing, China. 978-1-4244-6769-3/10/$26.00 ©2010 IEEE.

[10] Andrey Petukhov and Dmitry Kozlov, “Detecting Security Vulnerabilities in Web Applications Using Dynamic Analysis with Penetration Testing”, Dept. of Computer Science, Moscow State University.

[11] Nuno Antunes and Marco Vieira , “Defending against Web Application Vulnerabilities”, University of Coimbra, Portugal, 0018-9162/12/$31.00 © 2012 IEEE,vol.-2,p.- 66-72.Katkar Anjali S and Kulkarni Raj B, “Web Vulnerability Detection and Security Mechanism”, International Journal of Soft Computing and Engineering (IJSCE),ISSN:2231-2307, Volume-2, Issue-4, p.-237-241

[12] https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

[13] Acunetix Ltd. Acunetix Web Vulnerability Scanner. http://www.acunetix.com/, 2005.

[14] Jeremiah Grossman WhiteHat Security founder & CTO “Website Vulnerabilities Revealed “WhiteHat Security.

[15] J.Dhanamma, and T. Rohini, “The Unified Approach for Organizational Network Vulnerability Assessment”, IJSEA, Vol 4, No.5, September 2013.

[16] Riancho, “w3af User Guide”–Document Version 2.1, August, 2012.

[17] Jacobson, G. Booch, and J. Rumbaugh, “Rational Unified Process – Best Practices for Software Development Teams”, Rational Software Corp.,White Paper , TP026B, Rev 11/01.

[18] P. Kruchten, “The Rational Unified Process 3rd Edition: An Introduction". Reading, MA: Addison-Wesley Longman, Inc., 2004.

[19] W. Royce, “Software Project Management: A Unified Framework”. Reading, MA: Addison-Wesley Longman, Inc., 1998.