Detection & prevention of website vulnerabilities: Current scenario and future trends

Conference Paper · October 2017

DOI: 10.1109/CESYS.2017.8321315

2 authors:

Mohit Dua
National Institute of Technology, Kurukshetra
114 PUBLICATIONS **1,225** CITATIONS
SEE PROFILE

Himanshi Singh
University of Strathclyde
6 PUBLICATIONS **13** CITATIONS
SEE PROFILE

Proceedings of the 2nd International Conference on Communication and Electronics Systems (ICCES 2017)
IEEE Xplore Compliant - Part Number:CFP17AWO-ART, ISBN:978-1-5090-5013-0

Detection & prevention of website vulnerabilities: Current Scenario and Future Trends

Himanshi Singh
Department of Computer Science
NIT Kurukshetra
Kurukshetra, India
himanshisingh198@gmail.com

Mohit Dua
Assistant Professor, Department of Computer Science
NIT Kurukshetra
Kurukshetra, India
er.mohitdua@gmail.com

Abstract- Nowadays, web applications occupies larger part of our life. Major exercises are appurtenant on the security of these applications. Web application attacks like brute force, injection have become normal since past few years. Many attacks work at the real time. Mostly methods focus on prevention and detection of these attacks on the web applications. The main goal of the presented work is different from the studies going on these days on web application security. This work focus on the possible attacks on websites whether SQL injection, cross site scripting(XSS), etc. The proposed tool teaches the security analysts and the student pursing in information security about how to check the web applications vulnerabilities. This work has used XAMPP server for server and client environment.

Keywords- Web application vulnerability, cross site scripting attack, SQL injection attack, brute force attack, vampbug web application tool

I. Introduction

Website security has become a huge topic these days. All the activities are taking place on machines. With the help of different browsers the client can access server database or another resources to exchange information. The websites which gives huge online facilities like social networking, banking, business transactions, and many more, has the higher responsibilities of securing the database online. They should keep there servers up to date so that there are no loop holes left behind in the developing state. The security analysts should provide the developers the loose points on the web applications.

The statistics[24] of 2017 says that almost 30% of attacks are made on web applications. 62% data has been involved in hacking to accomplish vulnerabilities. 81% bug correlated apertures that bargaining chip fragile or embezzled passwords. 93% fiscally galvanized, perpetrated by established illegitimate bodies. 77% hauls out by botnets, not individuals. 32% oppressed SQL Injection errors.

The major attacks listed by OWASP in their 2015 survey, SQL injection attacks and XSS holds the top spots. The websites are more prone to these attacks. The injection attack happens when one input is pass through the website and it is able to access the database and other resources of the web application. Whenever there is an interaction between client and server, the actual attack starts wherever the application is rendered, it gives attacker a chance to input malicious code or commands through the browser. Later information is passed to the server, whenever it reaches to the backend through the query leading to the SQL injection attack and sometimes it post back to the client-server browser, which is known as cross site scripting attack. These attacks gives the unauthorized access to the user for system resources.

Majorly, these attacks misconduct innocent user once the malicious queries are retrieved and becomes the part of the internet. To overcome these types of attacks we should stop users to use internet, which is impossible in these days. Although internet has the capability to question the user if activities are malicious. There is no access at back end to retrieve the database neither there is any method to have secure transactions online. So, the measure for these attacks is to recognize them at benign and asking developers to fix them.

In this research paper we have discussed the various ideas of various authors presented in this field past few years. Later we have discussed about our tool which focus on teaching the methodology of performing these attacks for information security scholars and cooperate developers for securing their websites from these attacks.

The proposed tool focuses on all the OWASP top 10 listed attacks. This gives the developers a strength about how they can fix those bugs by using queries for accessing back end information. The tool gives the security to developers that how they can get their websites free from attackers.

The researches around the tools developed in past few years. [1]Dwivedi etl., in their propose tool has discusses about the perception of SQL injection which is a trivial problem for all the web-based applications in past few days. They have recapitulated up a array of proficiency existing for protecting data from SQL injections, that make changes in the program behavior of the website permitting the attacker to recapture and tweak the database. SQLAS detect the vulnerabilities in the source code. [2]Jin etl., have developed a tool which assesses the extent of vulnerability in android applications, and enforce a tool to analyze PhoneGap apps collected from the android vendors. The tool abate 478 applications as vulnerable, with only 2.30% negatives. [3]Tajpour etl., have discusses legion types of SQL attacks. The tool detects and prevents SQL injection attacks. And, also compares the tool with other existing tools. [4] Khochare etl., have proposes an application firewall tool. This tool helps the web applications to be protected from hackers. It analyzes the incoming request

and outoing response. [5]Bau etl., proposes a tool for detecting black box vulnerabilities in the web applications. They have also found the black box scanners for finding vulnerabilities. [6]Bhojak etl., proposes a vulnerability scanner that interpret websites for exploitable SQL injection and XSS attack vulnerabilities. [7]Rohilla etl., proposes a tool to show XSS vulnerabilities that how easy it is to inject malicious code in cookies.

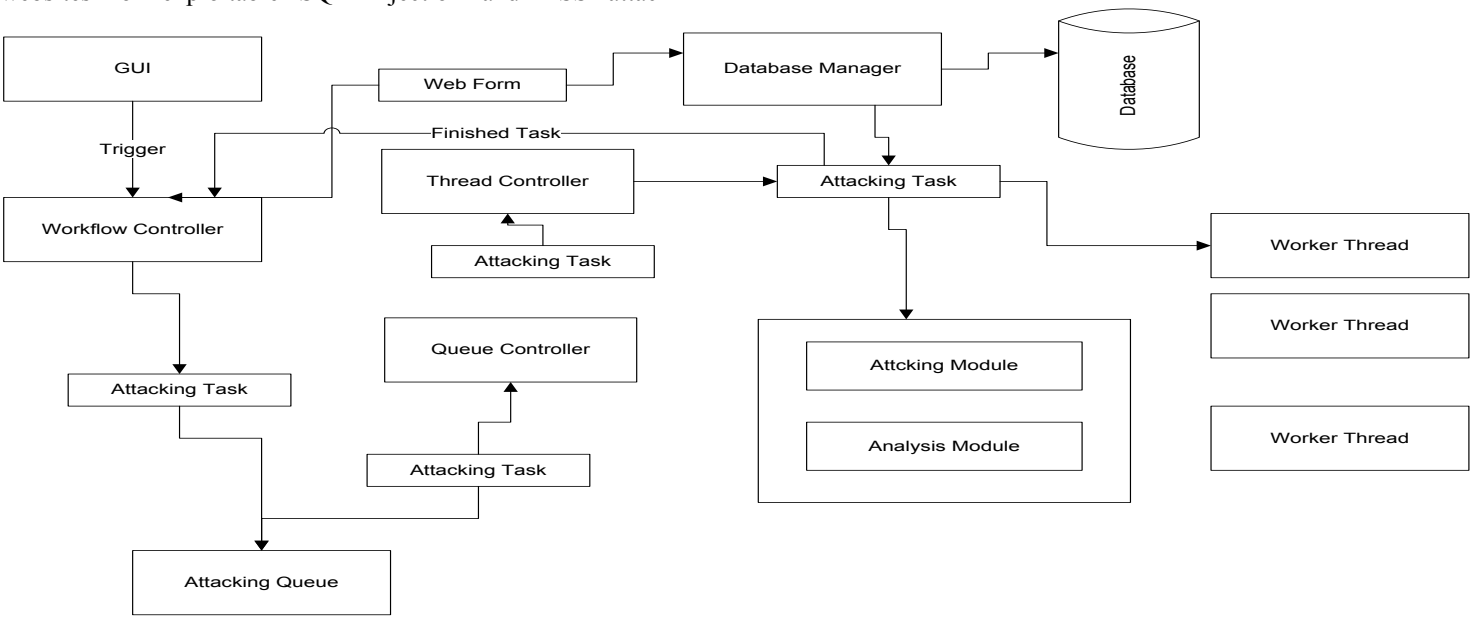

Figure 1: Flow Chart of Present Architecture

II. Related Work

In the first paper of discussion, [1]Dwivedi etl., have discusses about SQL injection attack, which is a normal problem for website attacks these days. They have proposed a tool for prevention and detection of SQL injection attack, which is SQLAS. They have discussed about various techniques for protecting the database of websites from SQL injection attack. Their proposed tool SQLAS, detects the errors in the source code which can be vulnerable in future. It gives the developer a chance of eyeball the structure. Detection of vulnerabilities is done by endorsing the queries of SQL which are based on XML apothegm with the description of the injection type performed on them. If possible vulnerability is exposed in the nature type then a warning message appeared to the developer. It also provides the line number in which vulnerability has detected. This helps the developer to make a code attack free by making modifications in the SQL commands. They concluded by expressing the robust feature of their tool as to tell the developers an error during the development time only.

[2]Jin etl., studied the embryonic of risk imposed by HTML5 mobile apps. In a research they have identified many unique channels which can be easily inject in the code such as, contacts, messages, barcode etc. they implemented the tool on PhoneGap aps from Google playstore to find the vulnerabilities in those apps. 478 tools were used to find the vulnerabilities. A prototype was used named NoInjection as a patch to phone gap framework in android. It filters the vulnerabilities in code from the attack channel.

[3]Tajpour etl., has presented the various type of SQL injection attacks in their paper. Also, compares various type of tool for detection and prevention of SQL injection attack, on the basis of their ability to stop SQL injection attack. Factors of comparison are deployment requirement and evaluation programs like, efficiency, effectiveness, stability, flexibility, performance.

[4]Khochare etl., proposes an application firewall tool for protection of web applications from attakers. The proposed tool analyzes the incoming request to web applications and responses of those requests from web applications. There is a business logic module in the tool which has all database of incursion, apothegms and approaches for detection and prevention of web applications. The tool secures web applications from SQL injection attacks, cross site scripting (XSS) attack, buffer overflow attack, cookie poisoning, forceful browsing and directory traversal attacks.

[5]Bau etl., have discusses about the black box vulnerabilities in their article. They have made black box scanners for detecting vulnerabilities of black box. They have found that black-box web application vulnerability scanners do expend testing effort in fuzzy distribution to the vulnerability citizenry in the wild.

[6]Bhojak etl., article tells about how accessible it is to find vulnerabilities and exploit them in many web applications. The popular example of vulnerabilities is consider as input validation attacks. They have discussed that many developers are unaware of security in web applications. Due to which there is general dissension that prevail a large number of vulnerable applications and web sites on the web. Their research proposed the flow of web vulnerability scanner that inspect web sites for naive SQL and XSS vulnerabilities.

[7]Rohilla etl., in their study has conducted a survey on Cross Site Scripting attack (XSS) with different real life worms to show how facile it is to attain the vulnerabilities of a web based application. The worms taken consider are actually affecting banking, social networking, healthcare, etc. These worms on web applications like stealing cookies details, credit

card number, passwords and data breaches, are serious effects on the web applications.

III. PRESENT ARCHITECTURE

All the authors discussed above has used the architecture as discussed above in figure 1.

IV. CORE TECHNOLOGIES

In this topic we discuss about the technologies used in the studies paper. [1]Dwivedi etl., have discussed the SQL injection for PHP web applications they have proposed a tool for detection and prevention of SQL attack which will be effectual for simple and composite data structures. This tool makes it easy for implementation. During implementation all the data will be at the secure place. These will be stored in some XML data format and referred as XML-rules. So whenever any script comes as an input from client , the server checks it in XML sheet. The XML rules will be written and stored individually for each input. This makes the developer effort accessible, now they just need to do is to conclude the authentication task for validating any web application data.

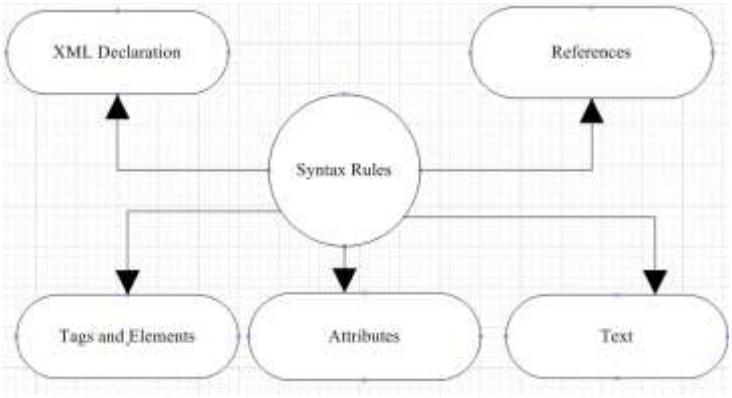

Figure 2: XML Rules

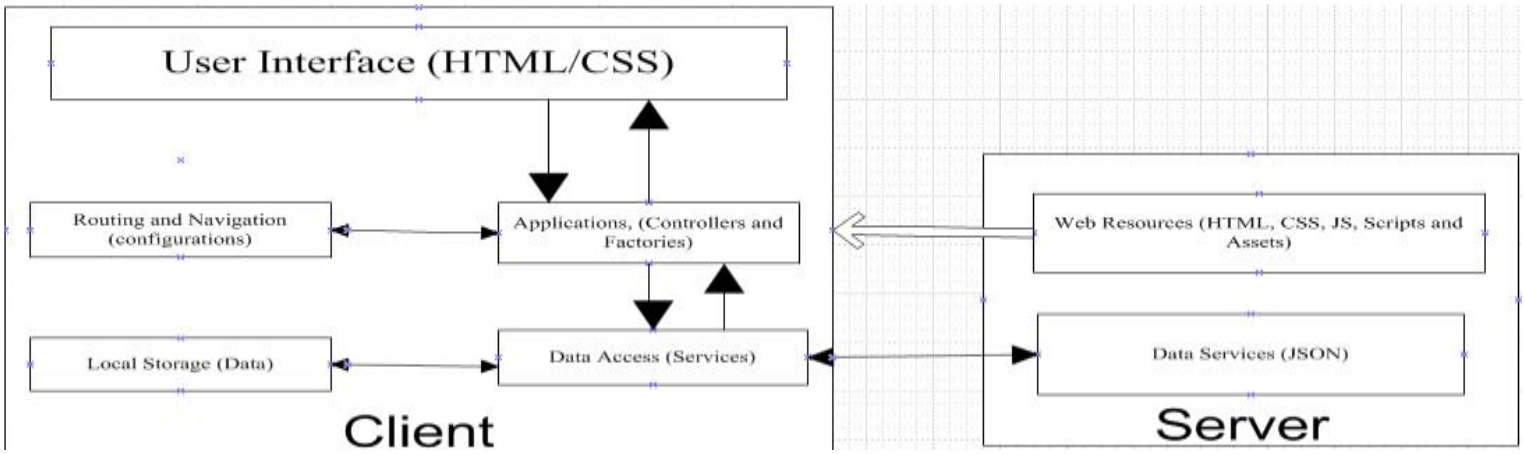

Figure 3: Web Application Security in HTML5

[2]Jin etl., have discussed various techniques of android phones vulnerabilities. They have studied the potential of risk imposed by HTML5 mobile applications. In their research they have identified many unique channels which can be inject in the code such as, contacts, messages, barcode etc. they implemented the tool on PhoneGap aps from Google playstore to find the vulnerabilities in those apps. 478 tools were used to find the vulnerabilities. A prototype was used named NoInjection as a plot to phone gap groundwork in android. It filters the vulnerabilities in code from the attack channel.

[3]Tajpour etl. , have discussed different types of SQL attacks with the queries applied on it. They have described vulnerabilities in three types such as, an attacker sending malicious HTTP inquiry to web application, creating SQL commands and acknowledging the SQL statements to beaconed database.

They have also discussed the various SQL injection attack types like blind injection, timing attack, inference and alternate encodings. In the end they have also discussed about different tools of SQL injection attack detection and prevention in comparison with their proposed tool.

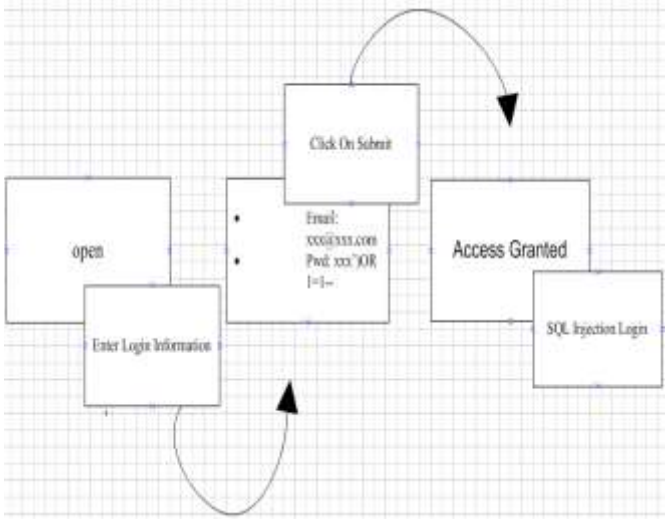

Figure 4: SQL Injection Login Flowchart

[4]Khochare etl., has proposed an access to inhibit web applications and servers consists of different scheduling. It aims at four factors, such as, auditing incoming and outgoing request of web servers, identifying all the calls and feedback response with firewall guidelines, policies, and the attack

definition currently at database, block the malicious calling and feedback, and users become observant about the attack that has been detected.

The tool prevents web application from malicious request from attackers and unauthorized access to web sites and web servers. Before sending to final destination, firewall application filters the traffic. Also, examines each request and response of web servers.

[5]Bau etl., in their research has discussed the usage of tool by writing URL of web application and also providing single user credential for application. User need to specify option for the page crawler for making maximization of page scanning. The scanner allows crawl only mode so; user can verify that given credentials are working properly. After this user sets up the scanning profile that is used in the detection of the vulnerability. After profile selection all scanners can run automatically. Many of them provide user mode where user can put values for scanning.

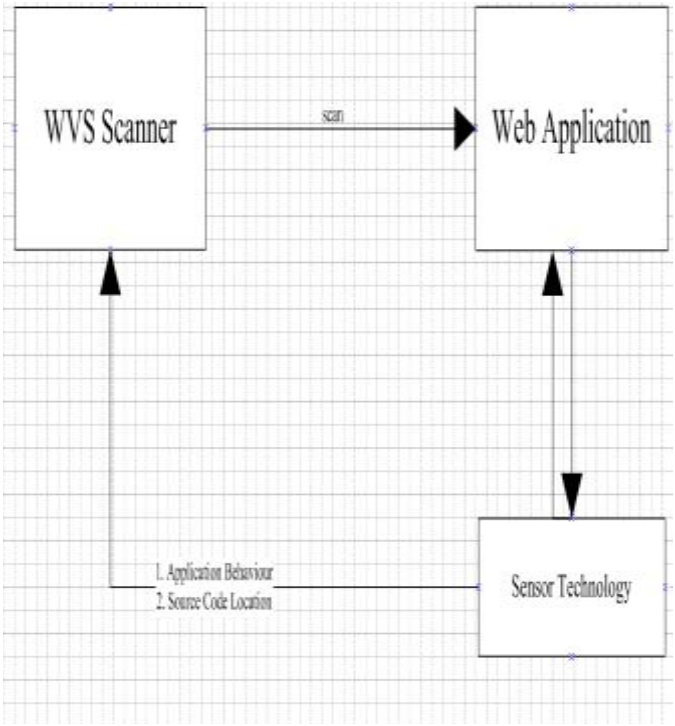

Figure 5: URL Scanner

[6]Bhojak etl., have discussed the various SQL queries used for web application attack. Like, bypassing command which is "1=1--", for accessing database of website. For XSS attack, the attacks are reflected and stored. The string used for the attack can be URL encoded such that content is scrawled for normal user on internet. The outbreak is considered lucrative when victim visits the particular URL. The query for accessing database is given as:

```
SELECT ID, LastLogin FROM Users WHERE User = ''OR 1=1 -- AND Password = '
```

[7]Rohilla etl., in their study has conducted a survey on Cross Site Scripting attack (XSS) with different real life worms that show easiness to exploit the vulnerabilities of an application designed on web. The worms taken consider are actually affecting banking, social networking, healthcare, etc. These worms on web applications like stealing cookies details, credit card number, passwords and data breaches, are serious effects on the web applications. They have discussed various worms on banking site, social networking site, etc.

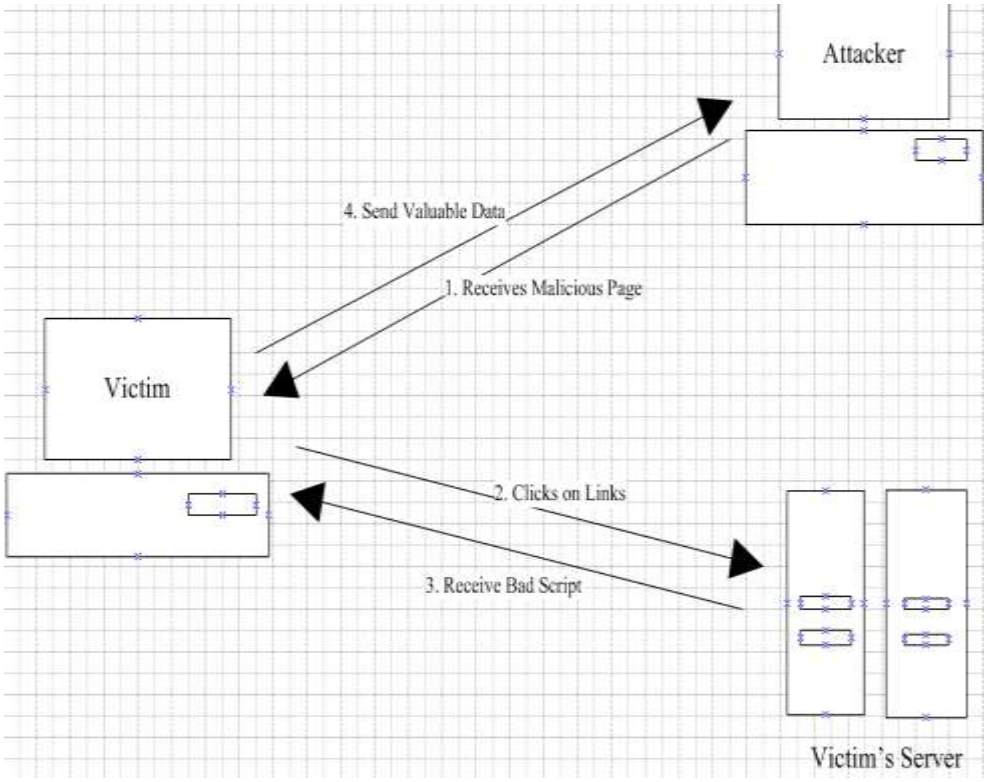

Figure 6: XSS Flow Chart

V. FUTURE WORK

Till now, all the present tools focus on particular vulnerabilities on website. Accuracy percentage of these tools is 77% approximately.

The proposed tool will show vulnerabilities in website. All the OWASP top 10 attacks will be covered in it. The tool will focus on all the loopholes in the website regarding the cyber attacks and a report will be generated in a PDF manner for the tester. The tester can also download the report and then generalize all the vulnerabilities.

The graphics are used to show the estimated percentage of loopholes in website, making easier for tester to understand.

VI. Conclusion

Main goal of this paper is to carry out a survey on the web application tools for detection and prevention of vulnerabilities. Further, it also carries a comparative analysis of the survey and is thus concludes with that web application tool is recommended above other means of optimizations because it provides clarity and errors in case of particular attacks. The paper provides various results as per their tools working.

Acknowledgement

This research is supported by NIT, Kurukshetra. We are thankful to university for their support and for providing necessary guidance concerning project implementation.

TABLE 2: A COMPARATIVE ANALYSIS OF WEB APPLICATIONS INTO VARIOUS APPLICATION DOMAINS.

Sr.No.	Author's name	Issue addressed	Area of application	Merits	Demerits	Tools used
1.	Vandana Dwivedi (2015)	SQL injection attack	PHP web application	It provides developers a method to detect and prevent their web applications.	It only consider PHP based applications.	PHP, MySQL, XML
2.	Xing Jin (2014)	HTML5based mobile applications.	HTML-5 android applications	Examines different channels used for injecting code. To determine the term of such a vulnerability in android applications.	The tool abates 478 applications as vulnerable, with only 2.30% negativity.	PhoneGap
3.	Atefeh Tajpour (2012)	SQL Injection	Web Application	Comparison of tools is based on requirements of deploying and common evaluation parameters.	Tools are not compared on the basis of efficiency, effectiveness, stability, and performance.	SQL
4.	Nilesh Khochare (2012)	Web Attacks	Web Application Firewalls	The tool prevents web application from different types of attacks like SQL injection, cross site scripting, buffer overflow, cookie poisoning, forceful browsing and directory traversal.	Anomaly detection module is not discussed in the paper	Firewalls
5.	Jason Bau (2010)	Black Box Vulnerabilities	Web Application	The idea explains that black-box web application vulnerability scanners executes, consumes testing endeavor in fuzzy distribution to the vulnerability populace in native.	Low coverage result for SilverLight, Flash and JAVA applet	JAVA
6.	Priyank Bhojak (2015)	SQL Injection And XSS Vulnerabilities	Web Application	It shows easiness in discovering and escapades large number of vulnerabilities in web applications.	Detection is considered as resultant.	Not specified
7.	Monika Rohilla (2016)	XSS Attacks	Web Application	Discussed serious effect of these worms on web applications like stealing cookie details, credit card number, password and data breaches.	Expanding research on other web applications	JAVA

References

[1] Vandana Dwivedi, Himanshu Yadav and Anurag Jain, "SQLAS: TOOL TO DETECT AND PREVENT ATTACKS IN PHP WEB APPLICATIONS" ,International Journal of Security, Privacy and Trust Management (IJSPTM) Vol 4, No 1, February 2015

[2] Xing Jin, Xunchao Hu, Kailiang Ying, Wenliang Du, Heng Yin and Gautam Nagesh Peri "Code Injection Attacks on HTML5-based Mobile Apps: Characterization, Detection and Mitigation"

[3] Atefeh Tajpour , Suhaimi Ibrahim, Mohammad Sharifi , "Web Application Security by SQL Injection DetectionTools", IJCSIInternational Journal of Computer Science Issues, Vol. 9, Issue 2, No 3, March 2012 ISSN (Online): 1694-0814 www.IJCSI.org

[4] Nilesh Khochare , Dr.B.B.Meshram , "Tool to Detect and Prevent Web Attacks ", ISSN: 2278 – 1323 International Journal of Advanced Research in Computer Engineering & Technology Volume 1, Issue 4, June 2012

[5] Jason Bau, Elie Bursztein, Divij Gupta, John Mitchell "StateoftheArt:AutomatedBlack-BoxWebApplicationVulnerabilityTesting"

[6] Priyank Bhojak, Kanu Patel, Vikram Agrawal, Vatsal Shah, "SQL Injection and XSS Vulnerability Detection in Web Application " Volume 5, Issue 12, December 2015 ISSN: 2277 128X International Journal of Advanced Research in Computer Science and Software Engineering

[7] Monika Rohilla, Rakesh Kumar, Girdhar Gopal , "XSS Attacks: Analysis, Prevention & Detection" Volume 6, Issue 6, June 2016 ISSN: 2277 128X International Journal of Advanced Research in Computer Science and Software Engineering

[8] OWASP, "Top Ten Project http://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

[9] D. Stuttard, M. Pinto, "The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws", Wiley, 2007

[10] PHP Security Consortium, "PHP Security Guide", http://phpsec.org/projects/guide/

[11] "Ruby On Rails Security Guide", http://guides.rubyonrails.org/security.html

[12] C. Anley, "Advanced SQL Injection In SQL Server Applications", http://www.ngssoftware.com/papers/advanced_sql_injection.pdf

[13] K. Spett , "Blind SQL Injection", http://p17linuxzone.de/docs/pdf/Blind_SQL_Injection.pdf

[14] R. A. McClure and I. H. Kr¨uger, "Sql dom: compile time checking of dynamic sql statements," in Proceedings of the 27th international conference on Software engineering, ser. ICSE '05, 2005, pp. 88–96.

[15] K. Kemalis and T. Tzouramanis, "Sql-ids: a specification based approach for sql-injection detection," in Proceedings of the 2008 ACM symposium on Applied computing, ser. SAC '08. ACM, 2008, pp. 2153–2158.

[16] D. Scott and R. Sharp, "Abstracting application-level web security," in Proceedings of the 11th international conference on World Wide Web, ser. WWW '02, 2002, pp. 396–407.

[17] P.Grazie, "Phd sqlprevent thesis," Ph.D. dissertation, University of British Columbia(UBC) Vancouver, Canada, 2008.

[18] M. Cova, D. Balzarotti, V. Felmetsger, and G. Vigna, "Swaddler: An approach for the anomaly-based detection of state violations in web applications," 2007.

[19] S. W. Boyd and A. D. Keromytis, "Sqlrand: Preventing sql injection attacks," in In Proceedings of the 2nd Applied Cryptography and Network Security (ACNS) Conference, 2004, pp. 292–302.

[20] W. G. J. Halfond, A. Orso, and P. Manolios, "Using positive tainting and syntax-aware evaluation to counter sql injection attacks," in Proceedings of the 14th ACM SIGSOFT international symposium on Foundations of software engineering, ser. SIGSOFT '06/FSE-14, 2006, pp. 175–185.

[21] V. Haldar, D. Chandra, and M. Franz, "Dynamic taint propagation for java," in Proceedings of the 21st Annual Computer Security Applications Conference, ser. ACSAC '05, 2005, pp. 303–311.

[22] G. Buehrer, B. W. Weide, and P. A. G. Sivilotti, "Using parse tree validation to prevent sql injection attacks," in Proceedings of the 5th international workshop on Software engineering and middleware, ser. SEM '05, 2005, pp. 106–113.

[23] Z. Su and G. Wassermann, "The essence of command injection attacks in web applications," SIGPLAN Not., vol. 41, no. 1, pp. 372–382, Jan. 2006.

[24] Web Application Security Statistics Report, the Case for DevSecOps by White Hat security, volume 12, 2017 (https://info.whitehatsec.com/rs/675-YBI 674/images/WHS%202017%20Application%20Security%20Report%20FINAL.pdf)