Vulnerability Assessment and Penetration Testing of Web Application

Prof. Sangeeta Nagpure
Department of Information Technology
K. J. Somaiya College of Engineering
Mumbai, India
sangeetanagpure@somaiya.edu

Sonal Kurkure
Department of Information Technology
K. J. Somaiya College of Engineering
Mumbai, India
kurkure.sonal@gmail.com

***Abstract*— As Internet usage is rising day by day security has become a vital facet to the Internet world. Security of the website in today's world is very important. Vulnerability Assessment and Penetration Testing are two different vulnerability testing. These tests have different strengths and are frequently combined to get a more complete vulnerability analysis. Penetration Testing and Vulnerability Assessments execute two different tasks, usually with distinctive outcomes, within the same area of application. For any organization, proper working of security arrangement is checked by Vulnerability Assessment and Penetration Testing. Web applications vulnerable to attacks like Session exploitation, Cross-Site Scripting, SQL injection, Cross Site Request Forgery, Buffer over Flows, and Security Misconfiguration etc. are described in Open Web Application Security Project Top 10. The manual penetration test or automatically penetration test can be done, which depends upon vulnerabilities. Comparison is made between these two tests.**

Keywords*— *Vulnerability Assessment and Penetration testing (VAPT), Cross-Site Scripting (XSS), SQL Injection (SQLi), Cross Site Request Forgery (CSRF), Open Web Application Security Project (OWASP).

I. INTRODUCTION

In most recent years, web hacking activities have been used excessively in internet applications. The main target of attackers would be web applications. Security of the website in recent years is very important because now all the events like communication, sharing the resources, social networking, e-governing, online banking, e-commerce, payment of utilities bills etc. through the Internet [1]. In web applications, security vulnerabilities may result in breach of data integrity, stealing of confidential data or affect web application availability. Thus the job of securing web applications is one of the most crucial.

The security of web applications from cyber threats involves very substantial challenges since security issues cannot be compromised. Security loopholes are observed by Vulnerability Assessment and Penetration Testing techniques [2]**.** Vulnerability assessment is a method where penetration tester scans a website loophole. After scanning, the next step is to find vulnerabilities which are the inherent security loopholes within the web application. In Penetration testing, penetration tester actually performs actions to exploit those loopholes and create an evidence of the test. It is also checked if there are more underlying vulnerabilities which are exposed as a side effect of the exploitation, and if those could further be exploited. Open Web Application Security Project (OWASP) Top 10 consists of various attacks to which web applications are vulnerable [3]. The major impact of attacks is a data loss or financial loss or reputation loss.

There are manual and automated methods to perform the security assessments of a web application. In automation method, there are lot of penetration testing tools, which are available either as open source or as a commercial product with different functionalities and applicability. Now the problem is to choose one the best vulnerability assessment tool. The answer is none of the tools are entirely complete in nature to identify the security risks in a web application [3].

In this paper, we provide a comparative and collective analysis of the web application vulnerability assessment and penetration testing methods. Section II provides the proposed work associated with the web application vulnerability assessments. Section III provides testing methods for vulnerability assessment and penetration testing of a web application. In section III, we also provide some of the parameters, which can help a security tester to select a set of appropriate vulnerability assessment tools, along with a comparative analysis of the vulnerability assessment tools that are available in the market. Section IV provides which testing method is most appropriate to provide a comprehensive security analysis of a web application. Section V provides related work associated with the web application vulnerability assessments and penetration testing. Finally, a section VI and VII concludes with an effective approach towards vulnerability assessment and penetration testing tests.

II.WEB APPLICATION VULNERABILITIES

Vulnerability is a flaw which allows an attacker to decrease system's information guarantee. Vulnerability Assessment is a method which tests the security of interactive applications such as e-banking, news broadcast and e-commerce web applications.

Web application penetration testing involves techniques leading to identification of potential vulnerabilities, which may compromise the web applications. Web Application

978-1-5386-4008-1/17/$31.00 ©2017 IEEE

Penetration Testing is expected to reveal vulnerabilities related to the following:

- OWASP top ten issues
- Session manipulation – fixation, hijacking, Horizontal privilege escalation, Vertical privilege escalation.
- Input validation failures resulting in SQL injection, cross site scripting, CSRF, etc.

A. *SQL Injection:*

Web application use database servers in the backend, whereby the Web page connects to the database, queries for data, and presents the fetched data to the browser. SQL injection attacks can occur if the input on the client side is not filtered properly before it is sent to the database. This can result in the risk of manipulating SQL statements, in order to perform illegal operations on the database. The web application developer does not make sure that values received from a web form, cookie and input parameter, at that time SQL injection vulnerability mostly occurred. These SQL injection vulnerabilities are authenticated or encoded before passing them to SQL queries that will be executed on a database server [4].

SQL injections can do more harmful attacks including updating data, deleting data, inserting data by executing commands on the server that can take and set up the malicious programs such as viruses, exporting valuable data such as email and passwords to the attacker‘s remote server and getting user login details etc. To Bypass Authentication, use one of the following Queries in the User Input:

- 'or ‘1’ = ‘1
- “or “1” = “1
- 1 or 1 = 1
- ‘or 1 = 1; --
- ‘or 1 = 1 –

B. *Cross site scripting:*

In cross site scripting, target scripts are inserted in a page that are executed on the client side i.e. user browser. These vulnerabilities can arise when the application proceeds with untrusted data and send it to the web browser without proper validation. Vulnerable object for XSS attack is textbox present in web application.

Vulnerabilities can be used to steal the identity, confidential data, bypass restrictions in websites, introduce malware attack, Session Hijacking, Denial of Service attacks and website defacement by attackers [2].

Followings are types of XSS vulnerabilities:

1. *Persistent XSS:* This attack is also known as Stored XSS. This attack occurs while attacker inject the malicious script into vulnerable web server, XSS script gets stored into database with other data and then it is visible to other users also who visit that webpage [2].

2. *Non- Persistent XSS:* This attack is also known as Reflected XSS. It occurs when user input is instantly returned by a web application in a form of an error message, search result, or any other response. User provided these input as a part of the request, without that data being made safe for rendering it into the *browser.* This user provided data is not stored in the database forever [2].

3. *DOM-based XSS:* DOM-based XSS attack is usually implemented using HTTP query parameters or URL parameter field. If web server runs the malicious script injected through the URL and shows the output of the script on the attacker's browser then this attack is successful. Consider the example in which the attacker aims to pop-up an alert message "111" in his own browser. Attacker is trying to check the XSS attack is possible on the attacked website. Attacker sends an alert script through an URL parameter. The server runs the script and a pop-up alert bar with a message 111 shows up in the webpage on attacker’s browser. This specifies that the website is vulnerable to DOM based XSS attack. In DOM-based XSS, all scripts are stored in browser’s cache and maintain record [5].

C. *Session Hijacking:*

Session Hijacking occurs when an attacker gets access to the session of a specific user. The attacker snips a valid session ID which is used to get into the system and steal the data. The below Fig.1 shows the scenario of session hijacking. This session is established among the client and server. Hacker sniffs the session ID and send the request to the server. Server cannot validate the malicious request due to the same session ID and gives successful response.

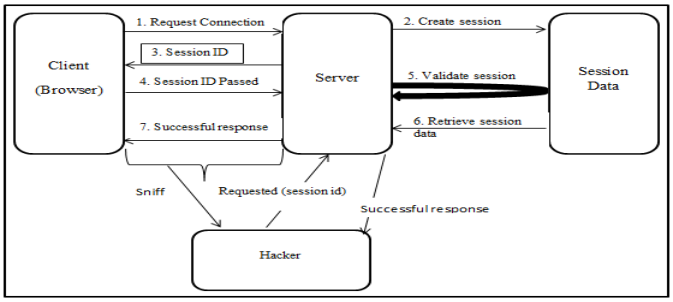

Fig. 1. Session Hijacking

D. *Privilege Escalation:*

Privilege Escalation means user receives privileges of the other users. These privileges can be used to delete the files, view private information or install unwanted programs such as viruses.

Privilege Escalation occurs in two ways:

1. *Vertical Privilege Escalation:* In Vertical Privilege Escalation, lower-level privilege user receives the high-level privilege user’s access. E.g. Normal user receives the privileges of admin user and normal user logged in as an admin user.

2. Horizontal Privilege Escalation: In Horizontal Privilege Escalation, normal user receives the privileges of other normal user.

E. Browser Replay Attack:

In web application, the websites usually generate a session cookie and unique session ID for each valid session. These cookies contain sensitive data like username, password etc. When the session is ended either by logout or browser closed shortly, these cookies should be invalidated. That means for each session there should be a new cookie. If the cookies are not invalidated, the sensitive data will exist/stored in the system. For example, to expose an attacker, a user will proceeds with a public computer (Cyber Cafe) and the cookies of the vulnerable site sits on the system. An attacker customs the similar public computer, after some time the sensitive data is compromised.

F. Insufficient Session Expiration:

Insufficient session expiration consists of some weak point. It is a consequence of poorly implemented session management. Due to this limitation attackers can stand up on design and implementation levels to gain unauthorized access to the particular application. Web developers are generally depends on either on server tokens or generate session identifiers within the application, while handling sessions. Each session should be expired, when user hits the log off button or after a certain period of time.

G. Session Fixation:

Session Fixation is an attack that allows an attacker to hijack a valid user session. The attacker has to establish a legitimate connection with the web server which issues a session ID or, the attacker can create a new session with the proposed session ID, then, the attacker has to send a link with the established session ID to the victim, she has to click on the link sent from the attacker accessing the site, the Web Server saw that session was already established and a new one need not to be created, the victim provides his credentials to the Web Server, knowing the session ID, the attacker can access the user's account.

H. Directory Traversal:

A directory traversal also known as path traversal. Directory traversal vulnerability occurs due to insufficient filtering of browser inputs from users. Therefore attacker can gain unauthorized access to restricted directories and files. These vulnerabilities can be found in web server or in application code that is executed on the server. Directory Traversal can result into exposing sensitive information such as content or code, to the malicious user [2].

I. Authentication Bypass:

For this, the access is given to the server or database because input is unfiltered. The information is extracted by the attacker if it is not cleaned by the developer. In Authentication bypass an attacker can forge the request and response, and get in without knowing user id or password [6].

J. Cross Site Request Forgery:

Cross site request forgery is a fake request came from the cross site. Cross-site Request Forgery (CSRF) vulnerability is possible as the attacker can mount any of the actions that can be done by the user such as creating user/entries, modifying / deleting data. This is possible because there is no client level components that can help server differentiate between a legitimate and illegitimate request.

K. File Upload:

Vulnerable File upload functionality is found because file extension is not being parsed for malicious intents on the server's and client's side. Uploaded files represent a significant risk to applications. Any attacker wants to find a way to get a code onto a target system, and then looks for a way to execute that code. Unrestricted file upload vulnerabilities can allow missing proper validation of file name, file content and size.

L. Clickjacking:

Clickjacking is a vulnerability in which malicious code is hidden behind the legitimate button or other clickable content on a website. Consider example of Clickjacking as shown in Fig 2. A User can trick into clicking LIKE button, on attacker's website when user visits attackers.com. Button visual integrity is claim your free iPad instead of LIKE button which is hidden behind claim your free iPad button. When user clicks on button, one LIKE is going on Facebook for links or images.

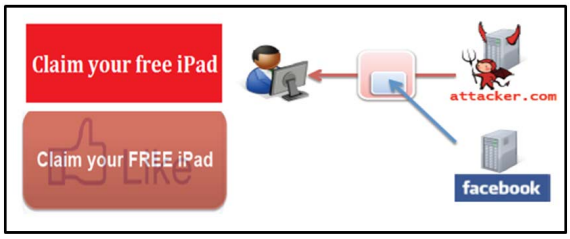

Fig. 2. Clickjacking

M. Browser Cache Weakness:

In this attack tester checks that the application correctly instructs the browser to not remember sensitive data. The simplest test consists of entering sensitive information into the application and logging out. Then the tester clicks the "Back" button of the browser to check whether previously displayed sensitive information can be accessed whilst unauthenticated. If by pressing the "Back" button the tester can access previous pages but not access new ones, then it is not an authentication issue, but a browser history issue. If these pages contain sensitive data, it means that the application did not forbid the browser from storing it.

III. Types of Testing Methods

The penetration test can be done either manually or automatically.

A. Manual Testing:

Some vulnerability is difficult to find using automated tools. That vulnerability can be identified by manual scan only. Web applications based on, their skills and knowledge of system, so that penetration testers can perform better attacks for it. Only humans can perform these methods like social engineering. Manual testing includes design, business logic as well as code verification. This technique is used by hackers to hack website or web application so that access can be gained [6, 8].

B. Automation Testing:

There are a wide variety of tools that are used in vulnerability assessment and penetration testing. Web applications are scanned by various softwares such as Acunetix, OWASP ZAP, Burpsuit, etc. Many companies used these tools to scan their products. It automatically shows various weaknesses.

Automated testing is the technique which uses the software that scans each page of web application. After this scanning a report is generated containing risks and methods to resolve them [6].

1. Acunetix:

Any of the web applications must be secured. In any technology based organization, their priority is highest. All the vulnerabilities for web application are discovered by Acunetix. In order to work any application accurately, it must be highly secured. Hence Acunetix helps in security of any web application. “Acunetix web vulnerability scanner” is one of the highly secured testing tools which find all the vulnerabilities of the system. Different types of vulnerabilities are varieties SQL injections and cross site scripting which can be found in a system.

Following are the steps for “Acunetix web vulnerability scanner”:

- Analysis of website: Acunetix Deep Scan scanner scans website which displays website related links and pages. All the information is obtained about the web application.
- Enabling the sensor called Acunetix Acu in scanner provides directories and files of the web application.
- Displaying the possible vulnerabilities of web application.
- After the completion of the scan, the results of scan are saved in a document. The Reporter allows generating reports of scan results in a printable format [6].

2. Burp Suit:

Burp suite is a grouping of various tools that place together to work in an passive/active mode, this support the penetration tester in the whole testing process, from the planning phase to identifying vulnerabilities and exploiting these vulnerabilities [9]. Burp-Proxy can work like the man-in-the-middle attack vector due to interruption in the traffic among the browser and target application. Burp-suit needs to be manually setup as the browser's proxy. Burp-suit uses the address "local host" and port "8080" by default, but this can be improved via the "Options" screen within the program. Secondly, configure the browser to use the Burp Proxy listener as its HTTP proxy server. To use Burp proxy, change browser's proxy settings by using the proxy host address (by default, 127.0.0.1) and port (by default, 8080) for both HTTP and HTTPS protocols, with no exceptions.

Burp Scanner is used to find the security vulnerabilities automatically in web applications. Passive scanning mode analyzes the contents of existing requests and responses, and deduces vulnerabilities from those. In active scanning mode, burpsuit sends various created requests to the application, and analyzes the responses to collect the proof of vulnerabilities [9].

3. OWASP Zed Attack Proxy (ZAP):

The OWASP Zed Attack Proxy Tool (or ZAP) is a pen testing and proxy tool. ZAP needs to setup the browser's proxy manually. The purpose of this tool is to allow developers to test the steadiness and security of their website or application. ZAP has a functionality to quickly attack a website with just a click of a button. Once the URL is entered and the user clicks on "Attack" button, the program will actively attack the website and report a list of problems that the website has. OWASP ZAP allows generating the reports of scan results in a HTML, XML format. ZAP uses the address "local host" and port "8080" by default, but this can be changed via the "Options Local Proxy" screen within the program. Check the browser’s proxy settings, and ZAP’s proxy settings [10].

Comparative analysis of different Automation Tools as shown in TABLE I:

TABLE I. COMPARISON BETWEEN AUTOMATION TOOLS

Features↓/ Tools→	**Burpsuit**	**Acunetix**	**OWASP ZAP**
Vulnerability Assessment	✓	✓	✓
Penetration Testing		✓	
Manual Testing	✓		✓
Passive Scan	✓		✓
Active Scan	✓	✓	✓
Login Sequence		✓	
Availability	Free/Paid	Paid	Free/Paid

IV. MANUAL VS AUTOMATION TESTING

There are multiple tools available in market, to make tester's life easy. While these automations are an important, there are few holes that a tester should be aware of, and this is especially true in case cyber security vulnerability assessment and penetration testing.

Vulnerability assessment is a method where an ethical hacker scans website loopholes. After scanning, the next step is to find vulnerabilities which are the integral security loopholes within the website or application. Penetration testing is a process in which the ethical hacker actually

performs actions to exploit those loopholes and create an evidence of the test. It is also tested if there are more underlying vulnerabilities which are exposed as a side effect of the exploitation, and if those could further be exploited [3].

It becomes very clear that Vulnerability assessment can be automated but the Penetration testing cannot. The subtle reason behind this is based on how human mind works in each case. To exploit vulnerability, there is need of cascaded intelligence to perform an action. Penetration tester need to take next steps based on the results of the first action. Then each and every application or website is different so need to use different tools which do perform these tests to some extent.

In example of net banking website with a page where user transfers money to another's account. This page can be vulnerable to Cross Site Request Forgery (CSRF) attack. Attacker can create a fake request and submit the form on behalf of the user. One of the techniques to fix this problem is to have a CSRF token as a hidden factor on the page. Now the problem is vulnerability assessment tool can only check the presence of this token and if present, it can pass that test successfully. Regrettably this is not true. The actual test is not only check the presence of token but also performs series of intellectual tests to validate the token at properly on the server side. Thus it can prevent fake submissions.

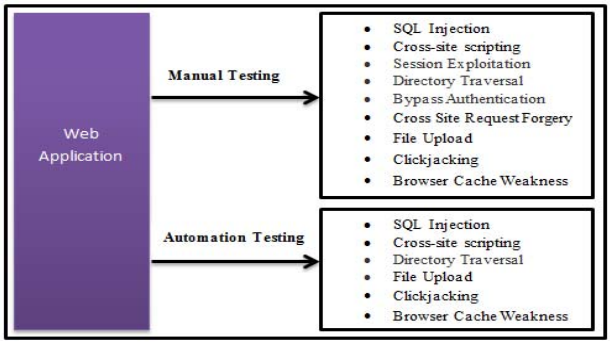

Fig. 3. Manual vs. Automation Testing

Fig.3 shows the analysis of net banking application. As shown in Fig.3 there are lots of attacks like Authentication bypass, Cross Site Request Forgery, Session exploitation etc., proved that automated vulnerability assessment is like swimming and being afloat, while manual vulnerability assessment is analogues to scuba diving, where different world altogether. Automated testing couldn't find these vulnerabilities in net banking application. Manual vulnerability assessment finds all possible vulnerabilities present in the web application but automated vulnerability assessment tools fail to find all vulnerabilities.

V. Related Work

In 2014, Khushal Singh, Vikas [1] have assisted the technique which detect all session checks and itemized number of analyses to evaluate performances of these session exploitation detection techniques. They considered the session exploitation mechanism in detail along with the prevention tactics and risk factor. The risk of web application checkpoint can be low, medium, and high depending upon how deep to manipulate the parameter of the web application. In the second phase, web application securities issues can be analyzed using backtrack. Backtrack is an adaptable functional system that derives with number of security assessment and penetration testing tools.

In 2015, Rohan Vibhandik, Arijit Kumar Bose [3] have proposed a new testing approach for vulnerability assessment of web applications by means of analyzing and using a combined set of tools to address a wide range of security issues. Their Technique demonstrates the vulnerability assessment tests of a web application by using combination of W3AF and Nikto tools. It shows how with a combination of tools, one can increase the vulnerability testing coverage for web applications, considering the OWASP Top 10 [1] based threat modeling of web applications.

In 2015, Insha Altaf, Jawad Ahmad Da [6] studied the possible vulnerabilities for any web application and suggested the removing techniques. Instead of using manual testing automated testing were used so that correctness and exactness can be improved. In addition to this, they also tried SQL injection methods. "Acunetix web vulnerability scanner" is used to carry all the vulnerabilities. While testing, an attempt is made by programmers or hackers to find vulnerabilities of the system. The vulnerable site is patched by using different injection techniques such as union based injection, authentication bypass and blind SQL techniques.

In 2016, Prashant S. Shinde, Shrikant B. Ardhapurkar [2] have proposed a Vulnerability Assessment and Penetration Testing (VAPT) techniques which helps to assess the usefulness and uselessness of the security measures of web application to stay protected against the rising Cyber threats. For any organization, proper working of security arrangement is checked by VAPT. VAPT exploits the number of vulnerabilities such as SQL injection attack, cross site request forgery attack, Cross Site Scripting attack, input validation URL (Uniform Resource Allocator) etc., in web application.

In 2016, Tanjila Farah, et.al [5] proposed the black-box testing methodology to implement and test XSS and CSRF attacks. This methodology gets nearly 30% of the web applications are vulnerable to XSS and CSRF attacks. While using black-box testing approach, executing XSS and CSRF attacks takes time. This is an on-going assessment. Since decade they have assessed the XSS and CSRF vulnerability of 500 web applications in Bangladesh. All testing are done manually. Their focus would be on XSS and CSRF vulnerabilities due to their high ranking on the OWASP list.

VI. Result

We performed manual penetration testing and automation penetration testing on two web applications.

1. E-Commerce Application
2. Cloud Application

During our penetration testing we used open source and commercial Web penetration tools to test known number of vulnerabilities in Web applications.

We used the following tools in automated penetration testing, as shown in TABLE II:

TABLE II. AUTOMATION SCANNERS OVERVIEW

Tools	Vendor	Version
Burpsuit	PortSwigger	1.6
ZAP	OWASP	2.5.0
Acunetix WVS	Acunetix	10.5

The vulnerabilities detected by using manual penetration testing:

- Cross-site scripting
- SQL injection
- Clickjacking
- File Upload
- Browser cache weakness
- Directory traversal
- Authentication Bypass
- Cross site request forgery

The vulnerabilities detected using automated penetration testing:

- Burpsuit detects only clickjacking, Browser cache weakness and directory traversal.
- Zap detect cross-site scripting, SQL injection, clickjacking, Browser cache weakness and Directory traversal but failed to bypass login authentication, cross site request forgery, session exploitation checks and file upload functionality in applications.
- Acunetix tool detect cross-site scripting, SQL injection, clickjacking, File Upload, Browser cache weakness and Directory traversal but failed to bypass login authentication and cross site request forgery in applications.

The Fig.4 shows the graph which presents the analysis of Manual and automated penetration testing:

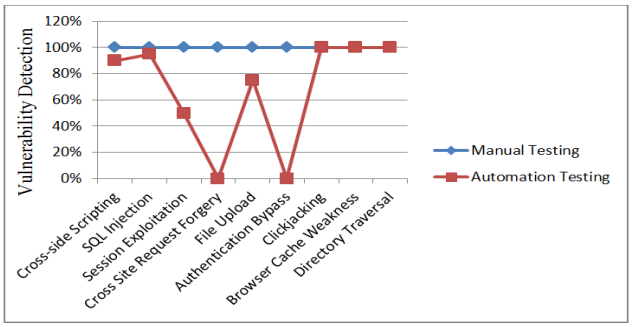

Fig. 4. Result analysis graph

It has been observed that accuracy of manual testing VAPT is 100%. Automated VAPT tools do not provide 100% accuracy.

VII. CONCLUSION

As discussed in above sections, attack volumes and density are on the rise. As attackers become more sophisticated, it is important that companies educate themselves on the threats that they are facing, and on the risk factor that is aligned with that threat. The task of securing web applications is of high priority since security vulnerabilities in web applications could outcomes as plagiarize confidential data, affect web application availability or breaking of data reliability.

Our study and experimentation using manual and automation reveals manual penetration test is more effective in terms of accuracy. On the basis of application skill and knowledge of the machine /system, Penetration testers can implement better attacks such that the system is being penetrated. Using Manual testing approach, cross-site scripting, SQL injection, clickjacking, File Upload, Browser cache weakness, Directory traversal, Authentication Bypass and cross site request forgery attacks are detected in web application. In terms of time and money, automation testing approach is used to detect the vulnerabilities in web application. Web scanners are used for performing the automatic web penetration test.

We propose that for VA, organizations should plan an integrated manual and automated testing approach so as to increase accuracy in identification of vulnerabilities in web applications.

REFERENCES

[1] Khushal Singh, Vikas, “Analysis of Security Issues in Web Applications through Penetration Testing”, International Journal of Emerging Research in Management &Technology, Volume 3, March 2014.

[2] Prashant S. Shinde, Shrikant B. Ardhapurkar, “Cyber security analysis using vulnerability assessment and penetration testing”, IEEE 2016.

[3] Rohan Vibhandik, Arijit KumarBose, “Vulnerability Assessment of Web Applications– A Testing Approach”, IEEE 2015 conference.

[4] Ossama B. AlKhurafi, Mohammad A. AlAhmad, “Survey of Web Application Vulnerability Attacks”, International Conference on Advanced Computer Science Applications and Technologies, IEEE 2016.

[5] Tanjila Farah, Moniruzzaman Shojol, Md. Maruf Hassan, DelwarAlam, “Assessment of vulnerabilities of web applications of Bangladesh: A case study of XSS & CSRF”, IEEE 2016.

[6] Insha Altaf, Jawad Ahmad Dar, “Vulnerability Assessment and Patching Management”, International Conference on Soft Computing Techniques and Implementations, IEEE 2015

[7] Mahin Mirjalili, Alireza Nowroozi, “A survey on web penetration test”, Advances in Computer Science: an International Journal, Vol. 3, Issue 6, No.12, November 2014.

[8] Shah, Sugandh, and B. M. Mehtre."A Reliable Strategy for Proactive Self-Defence in Cyber Space using V APT Tools and Techniques", School of Computer and Information Sciences, University of Hyderabad, Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on.

[9] Zoltan Panczel, “Burp Suite(up) with fancy scanning mechanisms”, SANS Institute InfoSec, December 20th, 2015.

[10] Russ McRee, “OWASP ZAP Zed Attack”, ISSA member, Puget Sound (Seattle), USA, November 2011