International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 6, August - 2012

Web Application Evaluation

Mr. Pavan Kumar Vadrevu

Abstract --- In this era of modern business web world, the use of web applications is increased beyond common comprehension. Every field throughout the world depends on it, but concurrently faces the risk of being copied. Many ways came to light to conquer the security problems even then new attacks see light every day. This is definitely a non-stop cycle, so keeping pace with identifying possible vulnerabilities is the only way out to handle the web applications attack-free. Here Penetration testing plays a crucial role to keep pace with spotting possible potential threats. My paper asserts the importance of this Penetration testing, vulnerabilities originated in recent times and their solutions. Web applications play a pivotal role in global market today. Its spread is limitless, with this limitless spread, also developed the problems of security proportionately. So time to time, security alerts became the need of the hour. Now my paper focuses on these possible recent vulnerabilities, their explorations, and fitting solutions. This definitely will contribute to expect even very sensitive and unknown vulnerabilities and thereby give way to fix them thoroughly. This is surely the demand of the hour.

Keywords— vulnerability, mitigation, exploit.

I. INTRODUCTION

Nowadays every organization faces the threat of attacks on web applications. Research shows that more than half of all data breaches are initiated in web applications. The goals of these attacks are information theft or abuse of resources. The reasons these attacks are successful can be broken down in technical and human causes. Web based applications; the organization is now one of the major points of vulnerability. Holes in the web application, stealing millions of credit cards, major financial and reputational damage for hundreds of companies has resulted in a compromise even thousands of achiness, are currently browsing the websites visited by an attacker to interfere.

Web application security scanner is a program that communicates with a web application through a web front end to identify potential vulnerabilities and weakness of building web applications. It is the result of black box testing. The source code scanners by trees, Web application scanners are actually carried out the attack, access to source code in order to identify vulnerabilities.

However with the great power of web applications, comes great risk of being exploited. Web application security deals with the security of the internet applications written in dynamic server side languages like Java, Ruby, ASP.net, php etc. Web application scanners can have a variety of vulnerabilities like input and output validation (SQL injection, cross site scripting).

Web applications need redesigned business for improved by creation of e-commerce, online banking, and highly modified client and companion portals likely. Touching business critical requirements facilities like sales, buying to the Web and support, organizations have lengthy, limits of the enterprise opening it up to growth communication with customers, partners, suppliers, and employees. Web applications are speed and simplified internal processes.

There is always a serious backdrop even after improved trust on Web Servers and Web Applications, which are naturally unconfident and easily compromised. About 93 percent of Web application vulnerabilities are as easy to exploit, as stated by Symantec. Web applications which are vulnerable not only the risk and equipment they also credit card numbers, account history, and personal details, such as sensitive information is highly confidential customer data and also provide a direct channel. 6,253 new vulnerabilities Symantec recognized has more vulnerabilities in 2010 than in any previous reporting period, $0.07 to $100 per credit card, the price for credit card data on underground forums ranged widely in 2010. Factors dictating prices include the rarity of the card and discounts offered for bulk purchases. The Internet Security Risk Report derived from data gather the ten million of Internet radars, actual research and active attacker’s communications and it offers a global view of state of the internet security.

II. INTENDED AUDIENCE

This paper is intended for every person in an organization involved in, or just curious about testing the security of web applications, its potential impacts and possible solutions. While all developers need to know the basics of web application security testing, application security testers need to know all the advanced techniques for finding and diagnosing security problems in web applications. Although the same techniques can be used as for functional testing, testing web application security requires special skills and insights of testers and developers. This paper shows the scope of a security test and prioritizes the work, understand the benefits and drawbacks of both manual and automated tools, know the techniques available and when to apply them, and learn how to determine the real risk value. In order to achieve these goals, we will assess the OWASP (Open Web Application Security Project) top ten security areas within a real world application.

III. PREREQUISITES

Although no prior experience with or knowledge about web application security is necessary, a basic understanding of the mechanisms of web applications and a basic awareness of web related security is assumed. The aim of this paper is to create awareness in the field of web application security testing. After reading this paper, you are better able to understand the specific problems in web applications understand and describe the OWASP top 10 vulnerabilities, understand the basics of testing for vulnerabilities in web applications, scope a security test and prioritize the work, understand the benefits and drawbacks of both manual and automated tools, understand the techniques available and when to apply them, determine the real risk value of web server and application vulnerabilities.

This paper explains why security should be considered when developing or deploying web applications. It identifies the locations of current security problems with web applications. During the introduction a definition of web application security is given. Trends that are influencing the current state of web application insecurity are also explained. This paper will provide a high level overview of the working of web applications like HTTP communication essentials, Client-side logic, HTML, CSS, JavaScript, Rich Internet Applications, Browsers and Safe Surfing, Sniffing and the problems of wireless networks, authentication, authorization and sessions management authentication mechanisms, basic or digest authentication, form based authentication, other forms of authentication, session management, misunderstandings about security of web applications, firewalls and network security, authentication and access control, encrypted connections and data encryption.

IV. VULNERABILITY

Vulnerability is a hole or a weakness in the application, which can be a design flaw or an implementation bug that allows an attacker to cause harm to the stakeholders of an application. Stakeholders include the application owner, application users, and other entities that rely on the application. The term vulnerability is often used very loosely. However, here we need to distinguish threats, attacks, and countermeasures. Please do not post any actual vulnerability in products, services, or web applications. Those disclosure reports should be posted to bugtraq or full disclosure mailing lists.

In today's world website security commonly ignored by many organizations in the online world. Organizations fail to protect their initiatives by falling short on performing a careful inspection of their websites. This happens most of the time that these organizations neglect what should have been the main concern of their group activities of hackers usually focuses on the manipulation of common web-based applications including login-pages, forms, and shopping carts etc. It has been exposed during thorough examination of the hacker's preparations that web applications are regularly embattled because of its plainness. It is common information that users can access web applications anytime within a day and seven days of a week. Hackers also find a web application the easiest to wish at since these contain control valued data, which permits access to the backend files like the customer database. It has already been verified that SSL, firewalls, as well as locked down servers are considered to be unusable against the attacks being expected at web applications. In addition, safety at the security level is also observed as inadequate since hackers initiate their attack on web application through the port 80, an area that needs to be reachable at all times. Very common web application exploits are SQL injection, XSS (Cross-site scripting), Buffer overflowed. Hence, the web vulnerability scanner becomes a useful as to offer full security on the websites. The scanner is a very much important since it can mechanically inspect websites and it can identify various code vulnerabilities such as SQL injection, XSS, buffer overflow, Directory traversal as well as ASP code injection. Example vulnerabilities are lack of input validation on user input, lack of sufficient logging mechanism, fail-open error handling and not closing the database connection properly.

V. WEB APPLICATION VULNERABILITIES

In order to successfully test the security of a system an understanding of the potential weak points is essential. The OWASP top 10 represents the areas where security mistakes are most frequently made. These areas can be used as a framework when evaluating the security of a web application and allow you to focus on the key design and implementation choices that most affect your application's security. This will provide on a high level how to test the most common problems in web applications like man-in-the-middle attacks, tools for testing, browsers, proxies, automated vulnerability scanners and specialized tools, OWASP Top 10, input validation errors, A1 – Cross-site scripting (XSS), A2 – SQL injection , A10 – invalidated redirects and forwards ,Broken identity management, A3 – broken authentication and session management, A5 – cross-site request forgery (CSRF), A4 – insecure direct object references , A8 – Failure to restrict URL access, implementation errors. Complete security is important in present system environment. Without it, the web application is facing threats from over-exploitation of any of the 16 classes of application.

A. Server misconfiguration

Hackers, including an inability to complete a web secure server, the server can be used incorrectly, the default accounts and services are limited, or unauthorized access to application to increase the removal of unwanted features. Well known platform vulnerabilities abusing the revealed weakness in web server. SQL Injection attack input rationale which sends commands to SQL web application governmental, who agreed to back-end database. Successfully executed SQL-injection attack the attacker can gain access to confidential information. XSS the attack on the trust relationship between the user and web applications, XSS is attempt trick the user or the user's browser into sending the attacker confidential information that can be used to steal the identity of the user. Command injection, the inserting system command hooked on program variables, such as a form as a resin that is inadvertently running on the server buffer overflow exploits, this command input over data. Succeeds verify buffer overflow attacks, it can help hackers machine running a remote shell, and get the same privileges application was granted under attack. Forceful browsing attempt to access unadvertised and informal URL addresses for promotion of entry into the root calendar of the web server or some marginal areas.

B. CGI-BIN Parameter Manipulation

It is another type of input attack authentication, in which data is to be formally approved at the server-side scripting illegally changed. If the appeal is passed limiting CGI-scripts are not properly authorized, so hackers are good chances for unauthorized system rights to modify files, run commands and perform other operations.

C. Form or Hidden Field Manipulation

It is an attack which contains an invisible field changed to deceive compatible misrepresentation inquiry and intervention in user sessions.

D. Cookie or session poisoning

Engineering weak cookies to snip the user session or make fun of real user applications.

The objective of this paper is to compare web vulnerability scanning tools like Nikto, ParosProxy, Webscarab, Webinspect, Whisker or libwhisker, Burpsuite, and Wikto. I have selected the tool called Nikto because it has awarded the best IT security tool for 2009 in the open source category for application scanners. From top 75 tools Nikto as ranked #16 tools.

VI. APPLICATION SECURITY PRINCIPLE

Application security principles are collections of desirable application properties, behaviors, designs and implementation practices that attempt to reduce the likelihood of threat realization and impact should that threat be realized. Security principles are language-independent, architecturally-neutral primitives that can be leveraged within most software development methodologies to design and construct applications. Principles are important because they help us make security decisions in new situations with the same basic ideas. By considering each of these principles, we can derive security requirements, make architecture and implementation decisions, and identifies possible weaknesses in systems. The important thing to remember is that in order to be useful, principles must be evaluated, interpreted and applied to address a specific problem. Although principles can serve as general guidelines, simply telling a software developer that their software must **fail securely** or that they should do **defense in depth** won't mean that much. Some proven application security principles are apply defense in depth (complete mediation) use a positive security model (fail-safe defaults, minimize attack surface), fail securely, run with least privilege, avoid security by obscurity (open design), keep security simple (verifiable, economy of mechanism), detect intrusions (compromise recording), don’t trust infrastructure, don’t trust services and establish secure defaults (psychological acceptability). Consider the exercise of designing a simple web application that allows one to send email to a friend. By evaluating and interpreting each principle, we can arrive at many of the threats to this application and ultimately derive a set of protection requirements. We want to end up with a complete list of what is required to offer this service securely.

Some of the example technologies, platforms and the languages are File systems, IO, Database, Java, .NET, PHP, C, C++, AJAX, Applet, CORBA, Client communication, Compiler, SQL, SSL Server, Struts, Unix, Web services, Windows, XML etc.. For example, an article on SQL injection in J2EE would be tagged with **Category: Database** and **Category: Java** as well as any other applicable categories like vulnerabilities and countermeasures.

VII. TESTING WEB APPLICATIONS

Although the same techniques as used for functional testing are applicable, additional skills are needed in web application security testing. Knowing the vulnerabilities, how to detect them and what tools to use, next to a risk-based approach are essential for a successful test execution. This module presents a high-level overview of various testing techniques that can be employed when building a testing program. In general for any web based system we will follow the basic testing techniques like differences with functional testing, Black-box vs. White-box, Test Methodology and Approach, Structured testing, Risk-based testing, Exploratory testing, integration into the software development lifecycle, when to test, Waterfall based environments and Agile based environments.

In Web System security testing first of all we need to do different terms to test the system the first and foremost thing is **Information gathering**. A typical entry in a computer network attacks, valuable knowledge about the target network to get help before the attack were included in the scanning activities. If the version of the application infrastructure system have been installed in order to reveal the weakness in the level of security patches will be able to know. Scanning technology has recently advanced, these methods make it difficult to detect. The modern way is to constantly scan, in the process of scanning the data transmission speed can be changes for the worse.

The second one is **Attack Planning**, using the knowledge gained at the stage of collecting information. The attacker can decide what kind of attack would be most effective at the expense of the target network. The aim is to plan the attack, which can be executed in the most effective and efficient wayside implementation, aimed directly at vulnerable network resources (routers, server, and application) and the next one is **Attack Execution**, most of the network and applications can be run using the available online tools. These tools can easily be downloaded over the internet, in rudimentary programming skills. These tools can be modified to perform a deliberate attack with ease.

By doing the above process we can assess different types of Security issues that are discovered during assessments will be mitigated based upon the following risk levels. Risk rating will be based on the **OWASP Risk Rating Methodology** they are of 3 kinds High, Medium, Low.

A. High

Any high risk issue must be fixed immediately or other mitigation strategies must be put in place to limit exposure before deployment. Applications with high risk issues are subject to being taken off-line or denied release into the live environment.

B. Medium

Medium risk issues should be reviewed to determine what is required to mitigate and scheduled accordingly. Applications with medium risk issues may be taken off-line or denied release into the live environment based on the number of issues and if multiple issues increase the risk to an unacceptable level. Issues should be fixed in a patch or point release unless other mitigation strategies will limit exposure.

C. Low

Issue should be reviewed to determine what is required to correct the issue and scheduled accordingly.

Remediation validation testing will be required to validate fix and or mitigation strategies for any discovered issues of Medium risk level or greater. Tools and/or techniques may be used depending upon what is found in the default assessment and the need to determine validity and risk are subject to the discretion of the Security Engineering team.

Full Assessment level is comprised of tests for all known web application vulnerabilities using both automated and manual tools based on the OWASP Testing Guide. A full assessment will use manual penetration testing techniques to validate discovered vulnerabilities to determine the overall risk of any and all discovered. A quick assessment will consist of a (typically) automated scan of an application for the OWASP Top Ten web application security risks at a minimum.

Application security activities are key practices that are performed during the software development lifecycle in order to reduce risk or increase assurance in an application. Note that these are just independent steps, and that they may be integrated into a number of different software development approaches. Given that these activities must support many different development lifecycles, they should not be too tightly coupled. To the extent possible, they may be integrated into a number of different software development approaches. Given that these activities must support many different development lifecycles, they should not be too tightly coupled. To the extent possible, they should be written to augment common software development practices. Differences applying the activity in different lifecycles should be noted.

Threat Risk Modelling is an activity to understand the security in an application. The specific vulnerability, related countermeasures, and impact are not required to discuss a threat, because the threat exists even if the target is well protected against it. For example, there is a threat that an attacker could launch a denial of service attack against your application even if you have sufficient defences in place. All attack articles should follow the Threat Agent template.

TABLE 1
COMMON APPLICATION SECURITY ISSUES

Grouping	Threats and Attacks
Input Confirmation	Buffer overflow; cross-site scripting; SQL injection;
Authentication	Network snooping, Brute force attack, glossary attacks, cookie repeat, recommendation theft
Authorization	Promotion of privilege, discovery of confidential data, tampering, attracting attacks
Configuration management	Unofficial access to organization interfaces, unofficial access to configuration stores, recovery of clear text configuration data lack of individual responsibility, over advantaged process and facility accounts
Sensitive information	Admittance sensitive data in storing, network snooping, data interfering
Session organization	Session control, session repetition,
Cryptography	Poor key group or key organization, weak or habit encryption
Parameter manipulation	Enquiry string management, form field management, cookie management, HTTP header management
Exclusion organization	Material discovery denial of facility
Auditing and Logging	The User repudiates execution an operation, attacker activities an claim without trace attacker covers tracks

VIII. EXAMPLE TOOL

Nikto

No one is a simple tool to assess the internet, we will find much to prevent and unsafe files, structures and programs on any type of web server. Nikto is Open Source web server scanner which performs comprehensive tests against web servers for multiple items, with more than 6400 potentially dangerous files CGI, and it will check an older version on more than 1000 servers, and version of the exact constraints on more than 270 servers. Verification of components and modules are regularly and mechanically updated. It also checks the server's configuration, such as appearance of multiple index files, HTTP server is found to be already installed the Web server software in server. Nikto is a noble CGI scanner. There are certain other tools that go fine with Nikto. It is not intended as a customized as and a over silent tool. It will test the web server as soon as possible in quickest way, and is justly obvious in the log files. Though, there are **care lib-whisker anti-ids** methods in case we want to try. Not every form of security problem, although most of them, there are some substances that only information which type of check that will look for things that might not have security bugs. But the fact that the security engineer may not know is available on the web server. These objects may not be noticeable in the information published. There will also check for unidentified objects that were seen scanned for in log files. The Nikto tool as a different versions 1.00 beta which as released in 27 Dec 2001.Over the sequences of the 2 years Nikto code is changed into most famous free available web vulnerability scanner. The 2.0 version is released Nov 2007 implements the several years of the developments.

Mr.Pavankumar.Vadrevu Working as an Assistant Professor in Dept of IT at **SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN BHIMAVARM** holds Diploma in Electrical and Electronics Engineering from State Board of Technical Education Andhra Pradesh. He obtained B. Tech degree in Information Technology from Swarnandhra Engineering College Narsapuram in 2008. He Completed M.Tech Information Technology in Guru Nanak Engineering College, Ibrahimpatnam, Hyderabad, Andhra Pradesh in 2011. Interested in Application Security and Performance Engineering.

REFERENCES

[1] Saltzer and Schroeder (see section 3)

[2] Ken Houghton. Vulnerabilities and Vulnerability Scanning. The Gary McGraw's 10 steps to secure software

[3] OWASP Development Guide Project

[4] from "Foundations of Security: What Every Programmer Needs To Know" by Neil Daswani, Christoph Kern, and Anita Kesavan (ISBN 1590597842)

[5] Understanding and Developing a Threat Assessment Model, S. Vidalis and A. Blyth, University of Glamorgan.

[6] Dafydd Stuttard,Marcuss Pinto, ‘‘The Web Application Hackers Handbook’’ Published by Wiley Publishing Inc (2008) I.chaudhry, S.Clarke, S.veney, E.Rachner, J.Sutton, ‘‘Web Application Security Assessment’’ published by SANS Institute(September 2009)

[7] Scambray, Vincent Liu,Caleb Sima, ‘‘Hacking Exposed Web Applications,3rd Edition’’ by Joel MCGraw-Hill publishers(2011).

[8] Tony Howlett, ‘Open Source Security tools’: A practical guide to security applications by Prentice Hall Publishers(July 29, 2004).

[9] Mc Clure,Saumil Shah,Shreeraj Shah ‘‘Web Hacking Attacks And Defence’’ byStaurt publisher Pearson Education Inc.(July 2002)

[10] OWASP Whitepaper, version 1.0,2003.

[11] Steven Splaine, ‘‘Testing Web Security: Assessing the Security of Web Sites and Applications’’ Published by John Wiley & Sons(Dec 03rd 2002)

[12] Michael Cross, Steven Palmer, ‘‘Web application vulnerabilities: detect, exploit,Prevent’’ Syngress , 2007

[13] SPI Dynamics. “Web Inspect.” SPI Dynamics Whitepaper, 2003.

[14] Mark Curphey, Joel Scambray,Erik Olson,and Michael Howard. ‘‘Improving Web Application Security threats and countermeasures’’ Published by Microsoft Corporation (2003).

[15] Dafydd Stuttard, The Web Application Hacker’s Handbook

[16] Online http://www.cirt.net/node/88 (Accessed on August 18, 2012)

[17] Online https://www.owasp.org/ (Accessed on July 12th 2012)

[18] Online http://www.perl.com (Accessed on July 8th 2012)

[19] Online http://www.sensepost.com/cms/resources/labs/tools/pentest/wikto/using_wikto.pdf (Accessed on August 18, 2012)

[20] Online http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/#MySQLInlineSamples (Accessed on August 18, 2012)

[21] Online http://www.w3schools.com/sql/sql_groupby.asp (Accessed on August 18, 2012)

[22] Online http://www.w3schools.com/sql/sql_union.asp (Accessed on July 4, 2012)

[23] Online http://www.w3schools.com/sql/sql_drop.asp (Accessed on July 4, 2012)

[24] Online http://www.securityfocus.com(Accessed on July 21, 2012)