Web application security

Kalyan Deshpande#1, Gaurav Khandkar#2
Sinhgad Institute of Management and Computer Application, Narhe Pune

ABSTRACT:-

Number of security vulnerabilities in web application has grown with the tremendous growth of web application in last two decades. As the domain of Web Applications is maturing, large number of empirical studies has been reported in web applications to address the solution of vulnerable web application. However, before advancing towards finding new approaches of web applications security vulnerability detection, there is a need to analyze and synthesize existing evidence based studies in web applications area

KEYWORDS:-

Acunetix, Web security, Injection , Vulnerabilities, Xss ,Scanner ,Security tool, Data Security, Viruses, Attacks

Web application security:-

It is the process of protecting websites and online services against different security threats that exploit vulnerabilities in an application's code. The inherent complexity of their source code, which increases the likelihood of unattended vulnerabilities and malicious code manipulation.

High value rewards, including sensitive private data collected from successful source code

Ease of execution, as most attacks can be easily automated and launched indiscriminately against thousands, or even tens or hundreds of thousands of targets at a time.

METHODOLOGY

1. Injection:-**Injection** flaws, such as SQL injection, LDAP injection, and CRLF injection, occur when an attacker sends untrusted data to an interpreter that is executed as a command without proper authorization.

Q: What to inject?
A: Queries, OS commands, codes and URL argument manipulations.
Q: Where to inject?
A: Wherever a user input is required or use can modify data. It can be a text box, username/password field, feedback fields, comment field, URL etc.

2. Broken Authentication and Session Management
Incorrectly configured user and session authentication could allow attackers to compromise passwords, keys, or session tokens, or take control of users' accounts to assume their identities.
* **Multi-factor authentication**, such as FIDO or dedicated apps, reduces the risk of compromised accounts.

3. Sensitive Data Exposure
Applications and APIs that don't properly protect sensitive data such as financial data, usernames and passwords, or health information, could enable attackers to access such information to commit fraud or steal identities.* **Encryption of data at rest and in transit** can help you comply with data protection regulations.

4. XML External Entity
Poorly configured XML processors evaluate external entity references within XML documents. Attackers can use external entities for attacks including remote code execution, and to disclose internal files and SMB file shares.
*** Static application security testing (SAST)** can discover this issue by inspecting dependencies and configuration.

5. Broken Access Control
Improperly configured or missing restrictions on authenticated users allow them to access unauthorized functionality or data, such as accessing other users' accounts, viewing sensitive documents, and modifying data and access rights.* **Penetration testing** is essential for detecting non-functional access controls; other testing methods only detect where access controls are missing.
6. Security Misconfiguration
This risk refers to improper implementation of controls intended to keep application data safe, such as misconfiguration of security headers, error messages containing sensitive information (information leakage), and not patching or upgrading systems, frameworks, and components.* **Dynamic application security testing (DAST)** can detect misconfigurations, such as leaky APIs

ISBN- 978-93-5254-640-7

7. Cross-Site Scripting

Cross-site scripting (XSS) flaws give attackers the capability to inject client-side scripts into the application, for example, to redirect users to malicious websites.

* **Developer training complements security testing** to help programmers prevent cross-site scripting with best coding best practices, such as encoding data and input validation**.**

8. Insecure deserialization

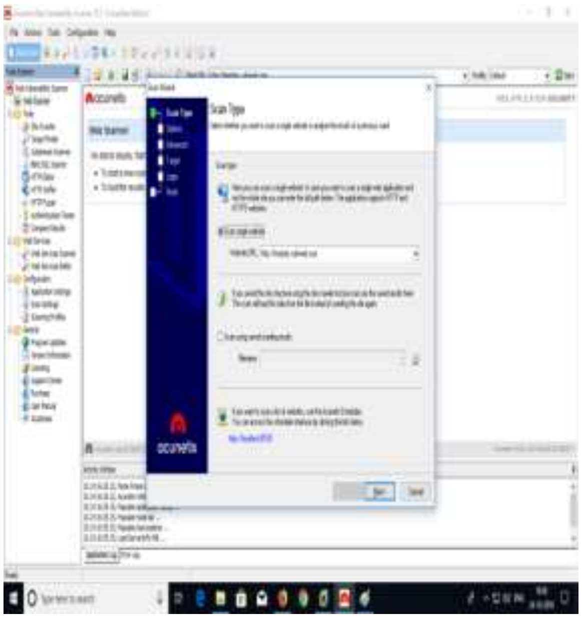
Insecure deserialization flaws can enable an attacker to execute code in the application remotely, tamper or delete serialized (written to disk) objects, conduct injection attacks, and elevate privileges.

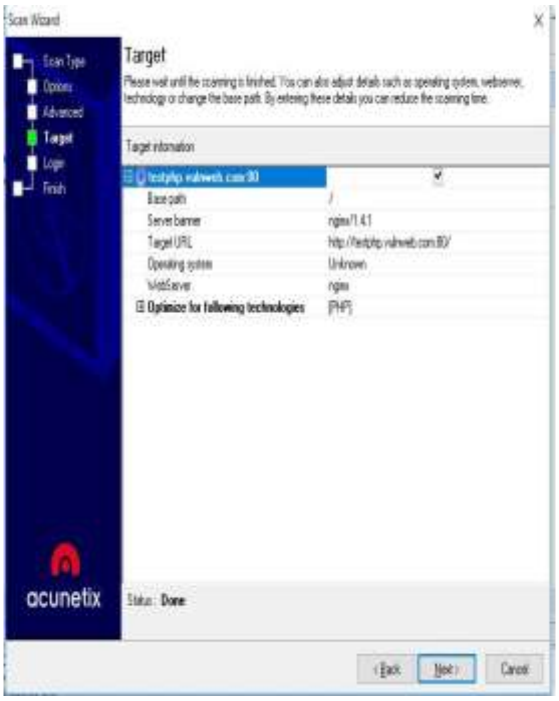
* **Application security tools can detect deserialization flaws** but penetration testing is frequently needed to validate the problem.

Tool:- Acunetix is an automated web application security ,testing , founded to combat the rise in attacks the web application layer. Acunetix WVS audits a website's security by launching a series of attacks against the site. It then provides concise reports of any vulnerabilities it found and will even offer suggestions on how to fix them.

Use cases
1)Penetration testing sowtware
2) website security scanner
3)external vulnerability scanner
4)web application security
5)vulnerability management software

Website security
1)Cross site scripting
2)Sql injection
3)Reflected xss
4)Csrf attack
5)Directory treversal





Target

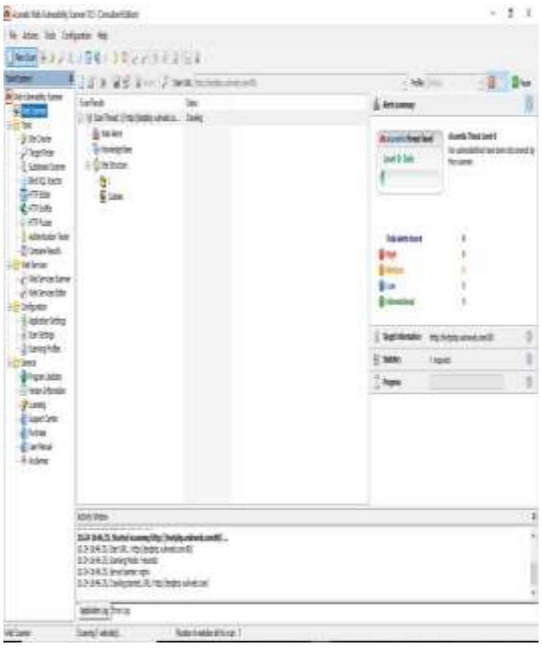
Please wait until the scanning is finished. You can also adjust details such as operating system, webserver, technology or change the base path. By entering these details you can reduce the scanning time.

Target information

testphp.vulnweb.com:80	✓
Base path	/
Server banner	nginx/1.4.1
Target URL	http://testphp.vulnweb.com:80/
Operating system	Unknown
WebServer	nginx
Optimize for following technologies	[PHP]

Status : **Done**

ISBN- 978-93-5254-640-7



Scan of http://testphp.vulnweb.com:80/

SCAN DETAILS

SCAN INFORMATION	
STARTTIME	24-10-2018 17:32:35
FINISH TIME	24-10-2018 17:54:24
SCAN TIME	21 MINUTES, 49
PROFILE	DEFAULT
SERVER INFORMATION	
RESPONSIVE	TRUE
SERVER	NGINX/1.4.1
SERVER OS	UNKNOWN
SERVER	PHP

Threat level

ACUNETIX

ONE OR MORE HIGH-SEVERITY TYPE

ALERTS DISTRIBUTION

TOTAL		**1**	
●	**HIG**	8	🟥
●	**Medi**	5	🟧
●	**Low**	9	🟦
●	**Infor**	3	🟩

Executive summary

Alert group	S	A
Blind SQL Injection	H	2
Cross site scripting	H	2
Cross site scripting	H	2
Directory traversal	H	2
Macromedia Dreamweaver	H	1
nginx SPDY heap buffer	H	1
Script source code	H	1
Server side request forgery	H	2
SQL injection	H	1
SQL injection (verified)	H	2
Weak password	H	1
.htaccess file readable	M	1
Application error message	M	8
Backup files	M	2
CRLF injection/HTTP	M	1
Cross domain data	M	2
Cross site scripting	M	2
Directory listing	M	1
Error message on page	M	7
HTML form without	M	5
HTTP parameter	Medium	2
Insecure	Medium	1
JetBrains .idea	Medium	1
PHP	Medium	1
PHP errors	Medium	1
PHP open_basedir	Medium	1
PHP	Medium	1
PHPinfo page	Medium	1
PHPinfo page	Medium	1
Source code	Medium	2
URL redirection	Medium	1
User credentials	Medium	2
WS_FTP log file	Medium	1
Clickjacking: X-	Low	1
Hidden form input	Low	1
Login page	Low	1
MySQL username	Low	2
Possible sensitive	Low	3
Possible virtual	Low	1
Broken links	Informat	7
Email address	Informat	1
Microsoft Office	Informat	1
Password type	Informat	3
Possible internal	Informat	3
Possible server	Informat	2
Possible username	Informat	3

OBJECTIVE :-

WEB APPLICATION THREATS (RISK)

Web application vulnerabilities are typically the result of a lack of input/output sanitization, which are often exploited to either manipulate source code or gain unauthorized access.

Such vulnerabilities enable the use of different attack vectors, including:

SQL Injection – Occurs when a perpetrator uses malicious SQL code to manipulate a backend database so it reveals information. Consequences include the unauthorized viewing of lists, deletion of tables and unauthorized administrative access.

Cross-site Scripting (XSS) – XSS is an injection attack targeting users in order to access accounts, activate Trojans or modify page content. Stored XSS occurs when

malicious code is injected directly into an application. Reflected XSS takes place when malicious script is reflected off of an application onto a user's browser.

Remote File Inclusion – A hacker uses this type of attack to remotely inject a file onto a web application server. This can result in the execution of malicious scripts or code within the application, as well as data theft or manipulation.

Cross-site Request Forgery (CSRF) – An attack that could result in an unsolicited transfer of funds, changed passwords or data theft. It's caused when a malicious web

application makes a user's browser perform an unwanted action in a site to which a user is logged on.

SCOPE:-

i)TLS

In this series on TLS security, we will focus on two widely known and used protocols in computer security, SSL and TLS. We will first start off with SSL, which stands for Secure Socket Layer and then we will talk about its successor, TLS, which stands for Transport Layer Security**.**

What is TLS/SSL?

The Transport Layer Security (TLS), and Secure Socket Layer (SSL) protocols' main purpose is to provide the following

ii)SQL Security: Securing MySQL Server

Databases can be found in everything from desktop applications, we applications, corporate servers to smart phones and other devices. Almost every software program relies on some sort of database to store its data. As applications continue to grow, so is the amount of data that is being stored into their databases, which is the main reason why they are the number one target of attackers–databases frequently contain information that is useful or desirable to an attacker.

iii) Word Press Security:-

Word Press is the most popular open source content management system (CMS). According to the latest W3Techs survey, almost 60% of all CMS instances use the platform — and 32.5 of all the websites on the Internet are Word Press sites. From the standpoints of deployment and usage, this is exciting: given its popularity, Word Press is well documented and full featured. But, it also means attackers are constantly looking to compromise vulnerable Word Press installations and the web servers behind them. To stay one step ahead you need Acunetix: a Word Press vulnerability scanner you can trust.

Web Service Security – The Technology and its Concerns:-

This white paper examines the technology behind Web Services and web service security – how the system is made available to the user, and the way connections are made to back-end (and therefore sensitive) data. These different elements come together to make Web Services a portal for users to access data, but also provide different entry points which may be exploited for illegitimate purposes. These security flaws bring about the need for an added security-assessing component in the Acunetix On Premise solution. Support for Web Services vulnerability scanning is now provided by a dedicated component which is specifically designed to detect exploitable entry-points in a Web Services system.

What can go wrong?

(DISADVANTAGES)

In some cases the web application vulnerability scanners may fail to detect some of the vulnerabilities mentioned above or may not work properly. Below is the list of the top reasons why automated vulnerability scanners might not work:

1) Custom-built authentication mechanism – when a web application uses proprietary approaches to authenticate the users (multi-step authentication, non-standard session management, etc) sometimes the scanner my fail to login and only scan unauthenticated parts of the web application

2) Web applications with heavy use of AJAX – many of the web scanners spiders do not handle dynamic AJAX content properly

3) Web sites with a lot of content or with a high number of dynamically generated pages – sites like Amazon, Facebook, eBay and YouTube where there are thousands of similar pages with different content, new pages are generated because of the user actions (including vulnerability scanner generating new pages while doing the scan) or when each requests receives a unique response URL

What do vulnerability scanners are not able to do?

In addition to the limitations mentioned above the scanners are not smart enough to test for specific vulnerabilities in the application logic. Web application vulnerability scanners are not able to test for:

1)Authentication vulnerabilities such as username enumeration, resilience to password guessing, any account recovery functions, credentials predictability or any other logic flaws in the authentication

2)Session management flaws like meaningful or predictable tokens, session fixation, mapping tokens to session, session hijacking etc.

3)Vulnerabilities in access controls where a user can access others' user data or admin functionality without having admin privileges assigned

4)Application logic flaws such as ordering negative number of items, skipping a stage in multi-stage processes (i.e. going straight to shipping page skipping the payment page) etc.
Shared hosting vulnerabilities – test

5)Leakage of sensitive information such as password hashes in the hidden fields or user logs

CONCLUSION:-

Automation makes it Easy:

Web application security is not a piece of cake, but by using automated tools it can be made easy.

Keeps Hackers Away:

To identify vulnerabilities in web applications, the hacker uses their own versions of automated scanners. By using automated web application security scanning, an organization can conduct a vulnerability test to avoid any unhandled flaw or vulnerability that allows attackers to attack. The only way to prevent attackers is to use automated security tools to find the vulnerabilities and weaknesses before they do.

Helpful in Vulnerability Outbreak:

A trusted & well know web application security scanner for automated scanning enable us to launch a scan within minutes and notify us regarding threats so that an organization can act accordingly.

Stay One Step Ahead from Hackers:

A manual web application security test restricts an organization to a number of known vulnerabilities. On the other hand, by using an automated web vulnerability scanner we can make sure that all parameters are being checked against all types of web application security variants.

REFERENCES:-

- https://www.acunetix.com/
- http://testphp.vulnweb.com/
- http://www.isis.vanderbilt.edu/sites/default/files/main_0.pdf
- https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5733964
- http://testphp.vulnweb
- www.google.co.in
- www.wikipedia.com
- https://www.acunetix.com/