EXPERIMENT 5

Aim -

Study of OWASP vulnerabilities.

Theory -

What is OWASP?

The Open Worldwide Application Security Project (OWASP) is an online community that produces freely-available articles, methodologies, documentation, tools, and technologies in the field of web application security.[8][9] The OWASP provides free and open resources. It is led by a non-profit called The OWASP Foundation. The OWASP Top 10 - 2021 is the published result of recent research based on comprehensive data compiled from over 40 partner organizations.

Vulnerability-

1. **Injection**

Attackers can provide hostile data as input into applications. Applications will process the data without realizing the hidden agenda. This will result in executing unintended commands or accessing data without proper authorization.
SQL injection, LDAP injection are well-known attacks.

Example
In a web application, the following SQL statement is used to get a record that belongs to a particular user. Let us assume the user would enter user id and password in the login screen.
String query = “SELECT * FROM Users
WHERE UseID = ’ ” + request.getParameter(“id”) +” ’ ”;

This query is supposed to return the record that belongs to a particular user.
Suppose the user enters 2’ or 1 = 1’ as id then query will be modified into

String query = “SELECT * FROM Users WHERE UseID = ’ 2’ or 1 = 1 ”;

In the above case, this query would return all records in the table instead of a particular user. Hence you would get access to other personal data.

Mitigation

Use server-side validation.

2. Broken authentication

Application functions related to authentication and session management are often implemented incorrectly so they allow anyone to assume other users' identities temporarily or permanently.
Attackers could compromise passwords, keys, or session tokens.

Example
Application session timeout is not handled properly. A user is doing some activity in an online banking application. Then the user closes the browser tab instead of doing "log out" and moves out of the place. If someone else opens the same browser after some time then they will have access to the previous user bank account.

Mitigation
Use a server-side, secure built-in session manager that invalidates session ID after idle and timeouts.

3. Sensitive data exposure

Security precautions should be given to data in rest as well as data in transit. Data can be seen when it's stored in a hard disk or when it's sent over the network as well. Many web applications do not protect the data properly. Attackers expose the weekly protected data using simple methods.
Attackers could steal sensitive data such as credit cards, passwords, etc.

Example
A simple example is Password is sent as plain text format in the network. Attackers can monitor the network and intercept the traffic using tools if required to get the details.

Mitigation
Apply security controls based on security standards such as PCI-DSS

4. XML External Entities (XXE)

Many old or poorly configured XML processors take an XML file as an input. Attackers can include hostile content in the XML file so that they can extract data or execute commands.

Example
POST http://example.com/xml HTTP/1.1
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY>

<!ENTITY xxe SYSTEM "file:///etc/passwd">]>
<foo> &xxe; </foo>
This XML file will get a password file from the server.

Mitigation
Upgrade all XML processors

5. **Broken access control**

Web applications support multiple roles in the operation. For example, there would be roles such as admin, regular user, manager, etc. Attackers can exploit flaws in implementation so that they can gain privileges to access data and perform operations where they don't have authorization.

Example
An attacker can simply try to browse different URLs.
http://www.example.com/app/getappinfo
The next URL can be accessed by admin only.
http://www.example.com/app/admin_getappinfo
If an attacker doesn't have admin privilege but he is able to access admin privileged pages then there is a security flaw.

Mitigation
Public pages can be accessible to everyone. Access to other pages should be prevented by default.
Disable web server directory listing

6. **Security misconfiguration**

Security misconfiguration is the most commonly seen issue. When we install new software users don't change the default user account username and password. Sometimes users don't update recent patches for security flaws.

Example
The application server comes with example applications. They are not uninstalled from the production server. Attackers can use known security flaws in the application to gain control of the production server.

Mitigation
Remove or do not install unused features and frameworks. Use a minimum platform without samples, documentation on the server. Please ensure that the default password is changed when you start to use the application.

7. Cross-Site Scripting (XSS)

XSS attack allows attackers to run javascript code into victim's browser

Example
Attackers could send an email to a victim that appears to be from a trusted company. The link could contain malicious javascript code. When a victim clicks this link, the javascript code collects information from the victim and sends data to the attacker website in the background. The victim will not be aware of the activity.

Mitigation
Escaping untrusted HTTP requests and validating user-generated content.

8. Insecure deserialization

Serialization is the process of converting an object into a stream of bytes so that it can be restored later. As part of deserialization, the object can be restored into its original state. Therefore, neutralizing the OWASP vulnerability.

Example
Suppose we store user id, password, and role for the given user in a cookie. This cookie can be serialized as an object. Attackers could change serialized objects and put the attacker's role as an admin user. In this case when an object is deserialized the attacker would get admin privileges.

Mitigation
Not to accept serialized objects from untrusted sources. If this is not possible, then implement integrity checks such as digital signatures on any serialized objects.

9. Using components with known vulnerabilities

Each application is made of multiple components such as libraries, software modules, and other frameworks. These components run with the same privileges as the application. If a component has known vulnerability then attackers can exploit the component first then the entire application.

Example
There are automated software tools available that will find the systems that are not patched and misconfigured.

Mitigation
Only obtain components from official sources over secure links

10. Insufficient logging & monitoring

Insufficient logging and monitoring allow hackers to experiment with hacking activities without being detected for a long time.

Example
A major US retailer reported that their internal malware analysis sandbox software had detected potentially unwanted sandbox software but no one responded to this detection. The sandbox was producing warnings for some time before the breach.

Mitigation
Establish or adopt an incident response and recovery plan.
As we have seen a quick overview of the top ten vulnerabilities, Let me provide one more perspective to see the need for security awareness irrespective of your current role in your organization.

Conclusion-

Thus, we have successfully studied OWASP and the various vulnerabilities in OWASP along with mitigation methods for tackling these vulnerabilities.