6. Security Misconfiguration

- Happens when the security structure or framework left open of an application by Database or system administrator.

Vulnerability:

* Default or weak password
* Obsolete software
* Unprotected files, records or databases
* Unnecessary highlighted features that are enabled

Attack Scenario:

* Weak or default accounts are not yet changed
* Directory listing isn’t disabled on server.
* Application Server accompanies test application that are not removed from the created server

Prevention:

* Review of the whole IT environment
* Featuring like issues as software that necessities patches or updates
* Default accounts that have unique passwords
* Audit security settings in systems and applications that are not set to secure qualities or values

7. Cross-Site Scripting (XSS)

- It is a web-based accomplished on a vulnerable or sensitive web application.

- The victim is not the application but the user.

Vulnerability:

* Reflected XSS – the attacker transmits a link to the victim’s application through social media, email, etc.
* Stored XSS - the attacker is able to store or plant a script. Once it is already stored, it is already permanent on the application.
* DOM XSS – the script is plant as a output of changing the DOM of the site on the code of client side.

Attack Scenario:

* The attacker uses an application that has an untrusted data in the structure of the HTML snippet without a validation.

Prevention:

* Use frameworks that instinctively escape or break XSS.
* Apply a sensitive code context when modifying the document browser on the client side.

8. Insecure Deserialization

- It happens when untrusted information or data is utilized to mishandle the logic of an application, deliver a denial of service (DoS) attack, or even execute self-assertive code.

Vulnerability:

* Untrusted information or data is utilized to abuse the application’s logic.
* Data or information tampering attacks.

Attack Scenario:

* Changing the serialized object to have an access on server
* Uses a PHP object serialization on the PHP forum to redeem a super cookie.

Prevention:

* Separating and running codes that deserializes in authorize environments.
* Monitoring deserialization

9. Using Components with Known Vulnerabilities

-

Vulnerability:

* The software is obsolete, unsupported or sensitive.
* Not upgrading frameworks, dependencies, platform etc,.
* No knowledge on the version that are being used in all components.

Attack Scenario:

* The attacker attempts to abuse a sensitive part on the server’s component.

Prevention:

* Removed unnecessary features, dependencies, files, components and documents that are not using.
* Monitor for components and libraries that are unmaintained.

10. Insufficient Logging and Monitoring

- Occurs when the security events are not logged properly and not monitoring the activity logs by the system.

Vulnerability:

* Unmonitored suspicious activity on applications and APIs
* The logs are only kept locally.
* Reviewable events like high value transactions, failed logins and logins are not logged

Attack Scenario:

* The attacker uses a common password using scans for users.
* Software was hacked by the attacker using a flaw in its software

Prevention:

* Make sure all control failures, login and server-side input validation is properly logged.
* Make sure that the logs are monitored.
* Build an effective alerting and monitoring.