***Introduction OWASP (Open Web Application Security Project) top 10***

1. Injection
2. Broken authentication & session management
3. Corss-site scripting
4. Broken access control
5. Security misconfiguration
6. Sensitive data exposure
7. Insufficient attack protection
8. Cross-site request forgery
9. Using components with known vulnerabilities
10. Under protected APIs

***Updated OWASP***

1. Injection
2. Broken authentication
3. Sensitive data exposure
4. XML External Entities
5. Broken access control
6. Security misconfiguration
7. Corss-site scripting (XSS)
8. Insecure Deserialization
9. Using components with known vulnerabilities
10. Insufficient logging & monitoring

***Injection***

Untrusted user input is interpreted by server and executed.

Data can be stolen, modified or deleted

* Prevent
  + Reject untrusted/ invalid input data
  + Use latest frameworks
  + Typically found by penetration testers/ secure code review
  + SQL injection is the most common

***Broken authentication and session management***

Incorrectly build auth. And session man. Scheme that allows an attacker to impersonate another user

Attacker can take identity of the victim

Do not develop your own authentication schemes

* Use open source frameworks that are actively maintained by the community
* Use strong passwords
* Require current credential when sensitive information is requested
* Multi factor authentication
* Log out or expire session after X amount of time
* Be careful with remember me functionality

***Cross site scripting***

Untrusted user input is interpreted by browser and executed

Hijack user sessions, deface web sites, change content

* Escape untrusted input data
* Latest UI framework

***Broken access control***

Restrictions on what authenticated users are allowed to do are not properly enforced

Attackers can assess data, view sensitive files and modify data

* Application should not solely rely on user input; check access rights on UI level and server level for requests to resources
* Deny access by default

***Security misconfiguration***

Human mistake of misconfigurating the system (providing a user with a default password)

Depends on the misconfiguration. Worst misconfiguration could result in loss of the system.

* Force change of default credentials
* Least privilege: turn everything off by default
* Static tools that scan code for default settings
* Keep patching, updating and testing the system
* Regularly audit system deployment in production

***Sensitive data exposure***

Sensitive data is exposed

Data that are lost, exposed or corrupted can have severe impact on business continuity

Prevent

* Always obscure data
* Update cryptographic algorithm
* Use salted encryption on storage of passwords

***Inssufficient Attack protection***

Applications that are attacked but do not recognize it as an attack, letting the attacker attack again and again

Leak of data, decrease application availability

* Detect and log normal and abnormal use of application
* Respond by automatically blocking abnormal users of range of IP addresses
* Patch abnormal use quickly

***Cross -Site request forgery (CSRF)***

An attack that forces a victim to execute unwanted actions on a web application in which they are currently authenticated

Victim unknowingly executes transactions

* Reauthenticate for all critical actions
* Include hidden token request
* Most web frameworks have buit in CSRF protection, but isn’t enabled by default

***Using components with known vulnearbilities***

Third party components that the focal system uses

Depednfing on the vulnerability it could range from subtle to seriously bad

* Always stay current with third party components
* If possible, follow best practice of virtual patching

***Underprotected APIs***

Applications expose rich connectivity options through APIs, in the browser to a user. These APIs are often unprotected and contain numerous of vulnerabilities.

Data theft, corruption, unauthorized access, etc..

* Ensure secure communication between client browser and server API
* Reject untrusted/ invalid input data
* Use latest framworks
* Vulnerability are typically found by penetration testers and secure code reviewers