Task 3

Top 10 web application vulnerabilities

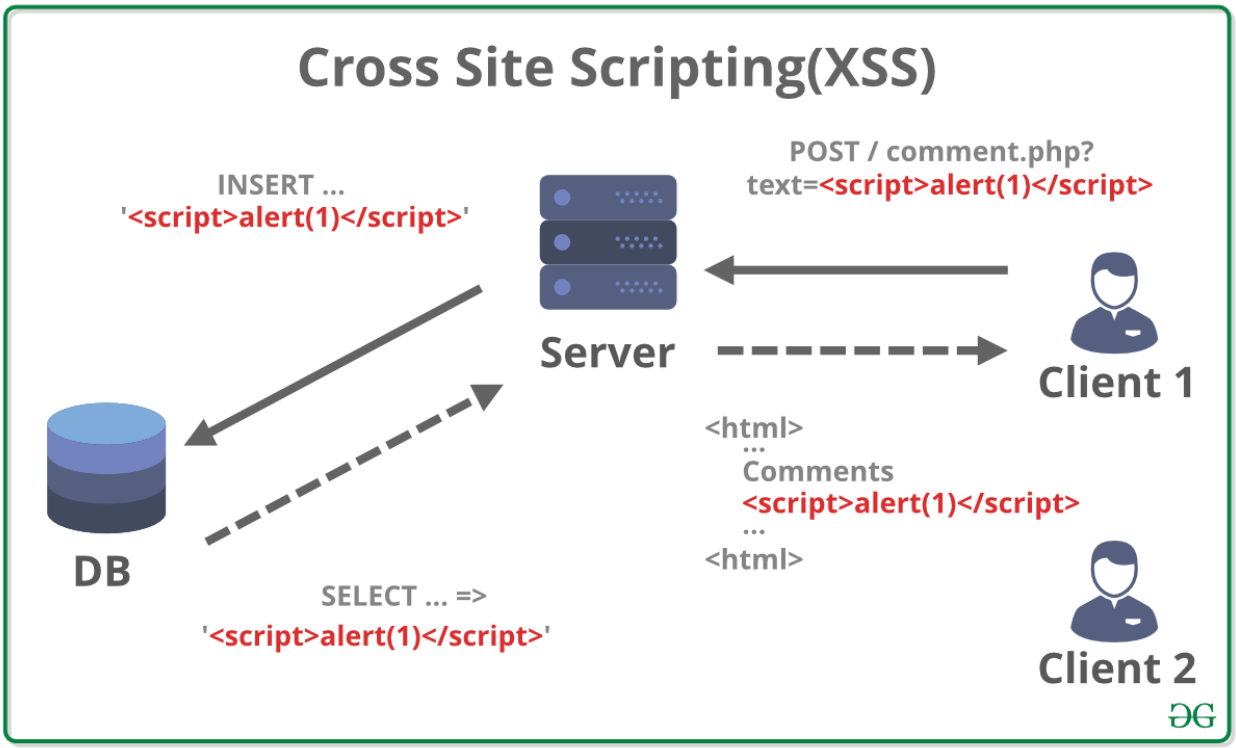
- **Injection Attacks (e.g., SQL Injection, NoSQL Injection):**
  Injection attacks occur when an attacker manipulates input data, like a user's input in a search field, to inject malicious code into a web application's backend. This can lead to the execution of unintended commands, potentially compromising databases or gaining unauthorized access to the system

What Are Injection Attacks | Acunetix

Injection attacks refer to a broad class of attack vectors that allow an attacker to supply untrusted input to a program.

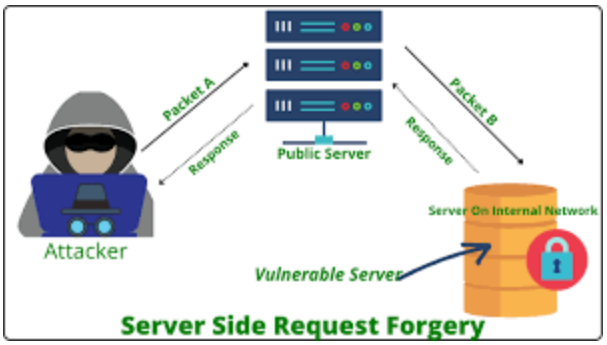
https://www.acunetix.com/blog/articles/injection-attacks/

- **Cross-Site Scripting (XSS):**
  Cross-Site Scripting involves an attacker injecting malicious scripts into a website's content, forms, or URLs. When unsuspecting users visit the compromised page, these scripts execute in their browsers, allowing the attacker to steal data, hijack sessions, or perform other malicious actions.



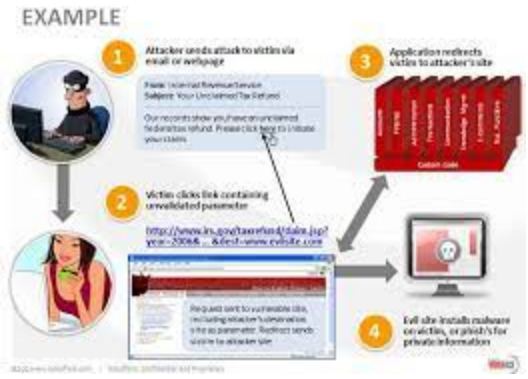
- **Cross-Site Request Forgery (CSRF):**
  CSRF attacks trick authenticated users into unknowingly executing malicious actions on a web application. By crafting a link or form that triggers actions on a target website, attackers can exploit the user's active session to perform actions like changing settings or making unauthorized transactions.

- **Server-Side Request Forgery (SSRF):**
  In SSRF attacks, an attacker manipulates a web application to make requests to other internal resources or external services. This can expose sensitive data, bypass security controls, or even lead to further attacks on the internal network.

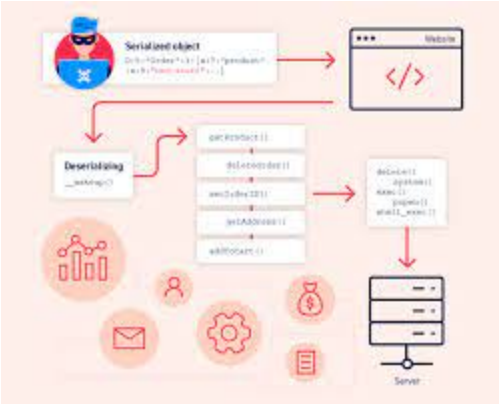


- **Broken Authentication and Session Management:**
  Weak authentication mechanisms or improper session management can allow attackers to gain unauthorized access to user accounts or bypass authentication altogether. This can lead to data theft, impersonation, and unauthorized system access.

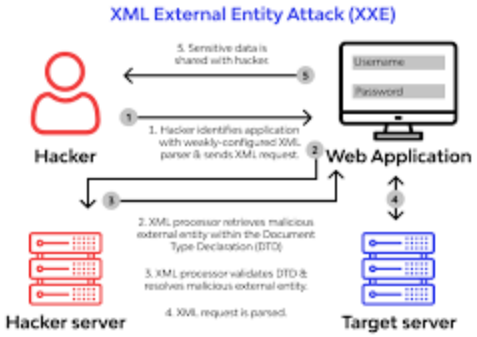
- **Sensitive Data Exposure:**
  Sensitive data exposure occurs when sensitive information like passwords, credit card numbers, or personal data is not properly protected. If attackers gain access to this data, it can lead to identity theft, financial loss, and other serious consequences.



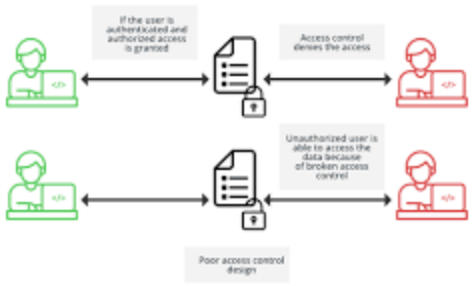
- **Insecure Deserialization:**
  Insecure deserialization vulnerabilities arise when serialized objects from untrusted sources are improperly handled by the application. Attackers can exploit this weakness to execute malicious code during the deserialization process, potentially leading to remote code execution or other attacks.



- **XML External Entity (XXE) Attacks:**
  XXE attacks target vulnerable XML parsers by exploiting entities defined in the XML document. Attackers can use this to read local files, conduct denial of service attacks, or retrieve sensitive data from the application's environment.



- **Broken Access Control:**
  Broken access control occurs when an application fails to properly enforce restrictions on user access to resources. Attackers can exploit this to gain unauthorized access to sensitive functionality or data.



- **Security Misconfigurations:**
  Security misconfigurations happen when system components, servers, databases, or web applications are not properly configured to ensure security. This can expose unnecessary attack surfaces and provide attackers with entry points into the system.

It's important to address these vulnerabilities through thorough security testing, regular updates, and best practices to ensure the security and integrity of web applications.

- [x] ~~Injection Attacks (e.g., SQL Injection, NoSQL Injection)~~
- [x] ~~Cross-Site Scripting (XSS)~~
- [x] ~~Cross-Site Request Forgery (CSRF)~~
- [x] ~~Server-Side Request Forgery (SSRF)~~
- [x] ~~Broken Authentication and Session Management~~
- [x] ~~Sensitive Data Exposure~~
- [x] ~~Insecure Deserialization~~
- [x] ~~XML External Entity (XXE) Attacks~~

- [x] ~~Broken Access Control~~
- [x] ~~Security Misconfigurations~~