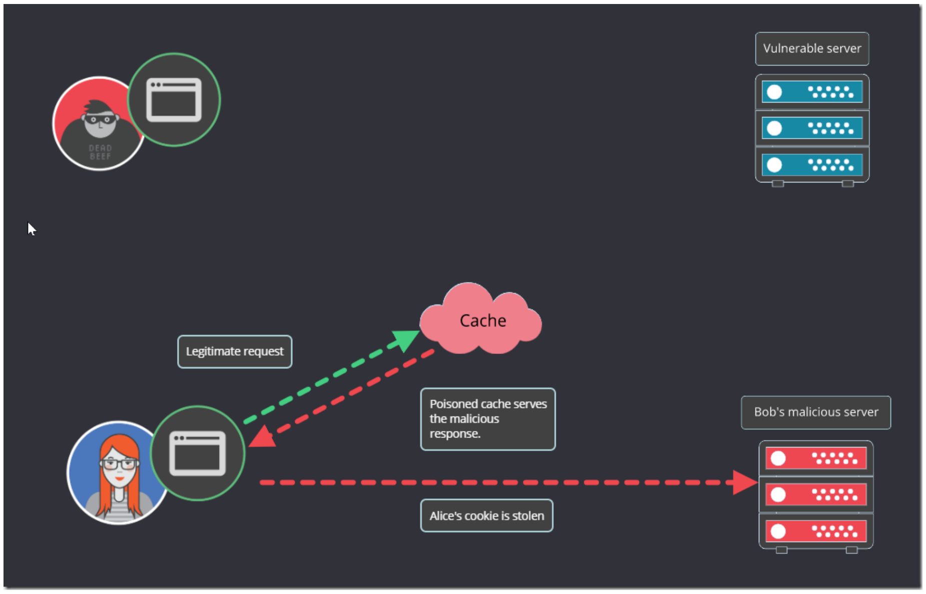
Saloni Ghule

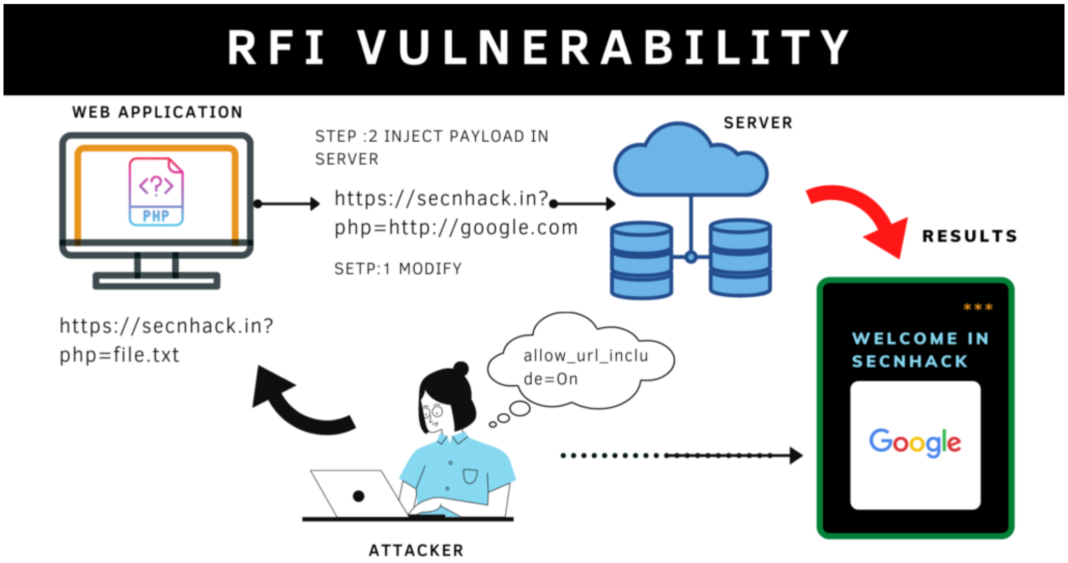
Task 4:

UNDERSTANDING WEB APPLICATION VULNERAILITIES

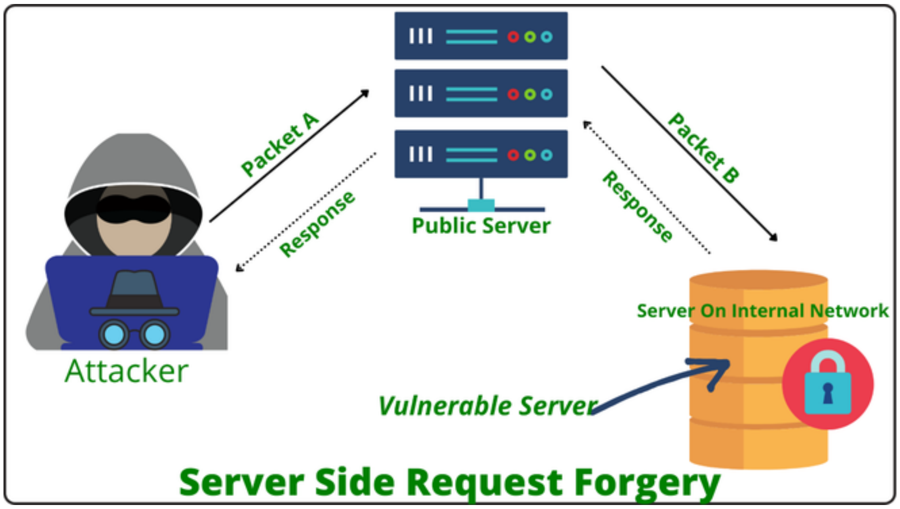
1. **HTTP Response Splitting:** HTTP Response Splitting occurs when an attacker manipulates input to inject newline characters into HTTP responses. This can lead to the creation of multiple responses, allowing them to inject malicious content or headers into the response, potentially bypassing security controls and causing a range of attacks, including cache poisoning and cross-site scripting (XSS).



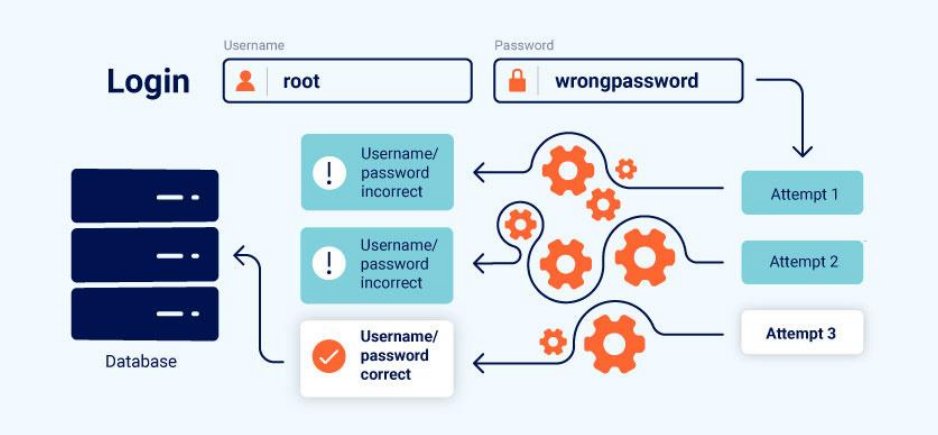
2. **Remote File Inclusion (RFI):** RFI is a vulnerability that occurs when an attacker is able to manipulate the inclusion of files from a remote server. This can lead to remote code execution by loading malicious scripts from an external source, allowing attackers to take control of the targeted system.



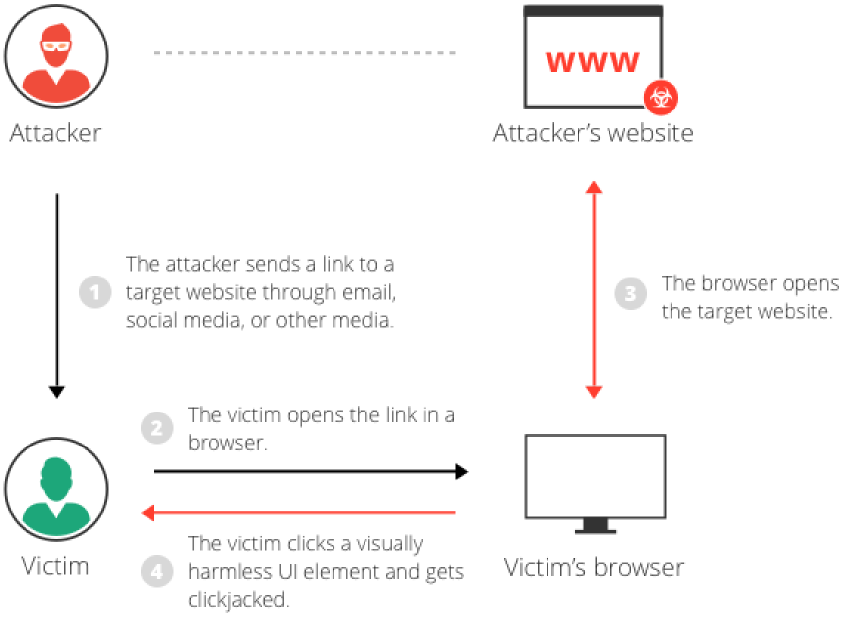
3. **Server-Side Request Forgery (SSRF):** SSRF is an attack that allows an attacker to make unauthorized requests from a vulnerable server to internal resources, potentially leading to data exposure, remote code execution, and further attacks against internal systems.



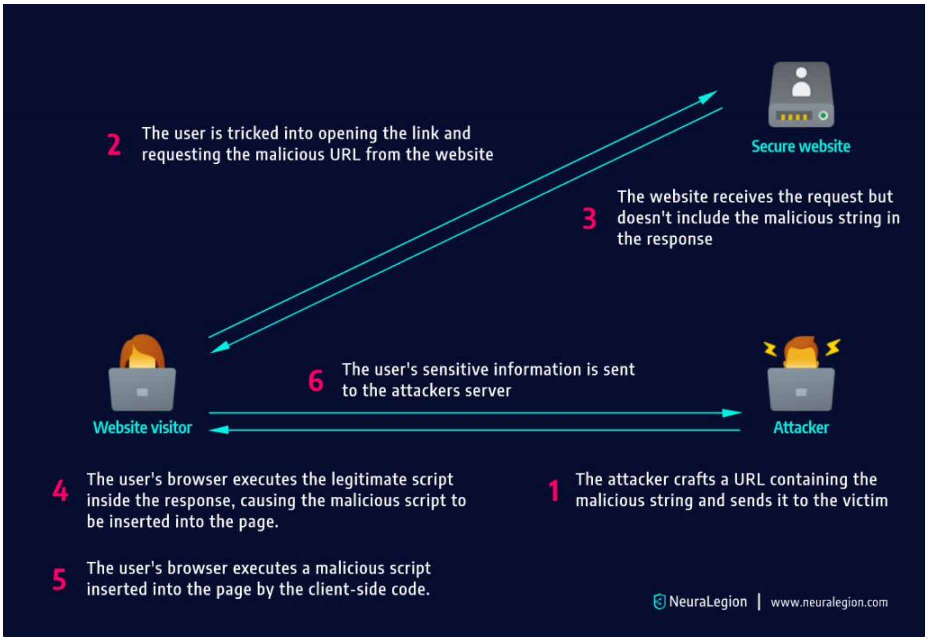
4. **Business Logic Vulnerabilities:** These vulnerabilities stem from flaws in the design and implementation of the application's business logic. Attackers exploit these vulnerabilities to perform actions that violate the intended workflow of the application, such as unauthorized access to restricted features or manipulating transactions.


Login
Username
root
Password
wrongpassword
Attempt 1
Attempt 2
Attempt 3
Username/ password incorrect
Username/ password incorrect
Username/ password correct
Database

5. **Clickjacking:** Clickjacking involves deceiving a user into clicking on something different from what they perceive. Attackers overlay transparent elements on top of legitimate website elements, tricking users into performing actions they didn't intend, potentially leading to unauthorized actions or data leakage.

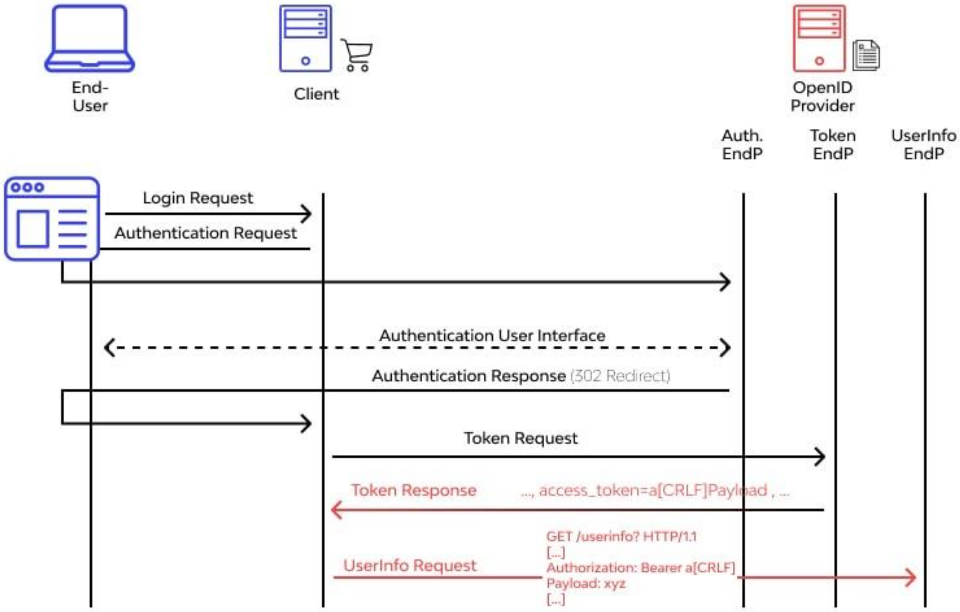

Attacker
www
Attacker's website
1 The attacker sends a link to a target website through email, social media, or other media.
3 The browser opens the target website.
2 The victim opens the link in a browser.
4 The victim clicks a visually harmless UI element and gets clickjacked.
Victim
Victim's browser

6. **DOM-based Cross-Site Scripting (DOM XSS):** DOM XSS occurs when an attacker manipulates the Document Object Model (DOM) of a web page to execute malicious scripts in the victim's browser. Unlike traditional XSS, which relies on server-side vulnerabilities, DOM XSS attacks manipulate the client-side code directly.

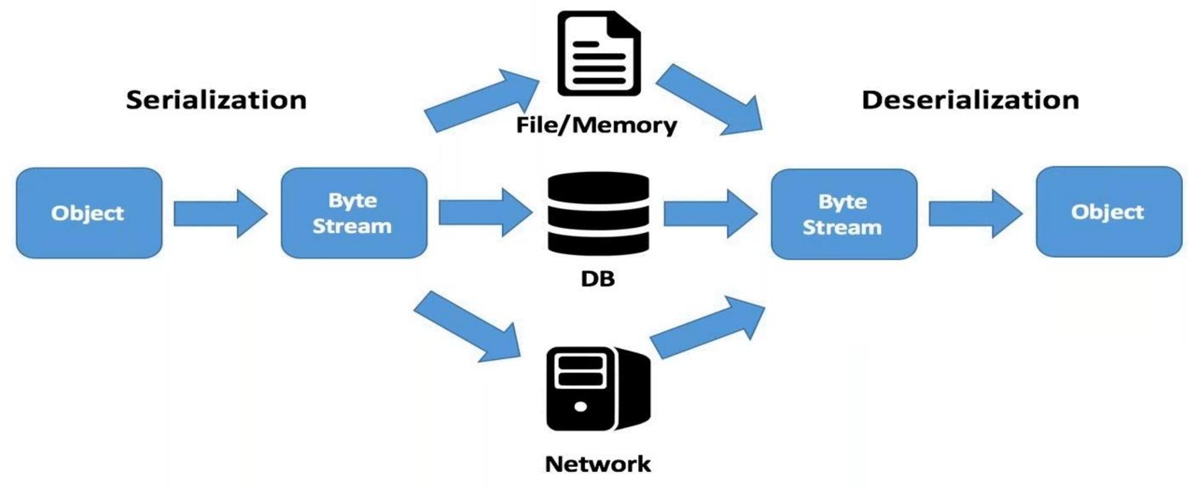


2 The user is tricked into opening the link and requesting the malicious URL from the website
Secure website
3 The website receives the request but doesn't include the malicious string in the response
6 The user's sensitive information is sent to the attackers server
Website visitor
Attacker
4 The user's browser executes the legitimate script inside the response, causing the malicious script to be inserted into the page.
1 The attacker crafts a URL containing the malicious string and sends it to the victim
5 The user's browser executes a malicious script inserted into the page by the client-side code.
NeuraLegion | www.neuralegion.com

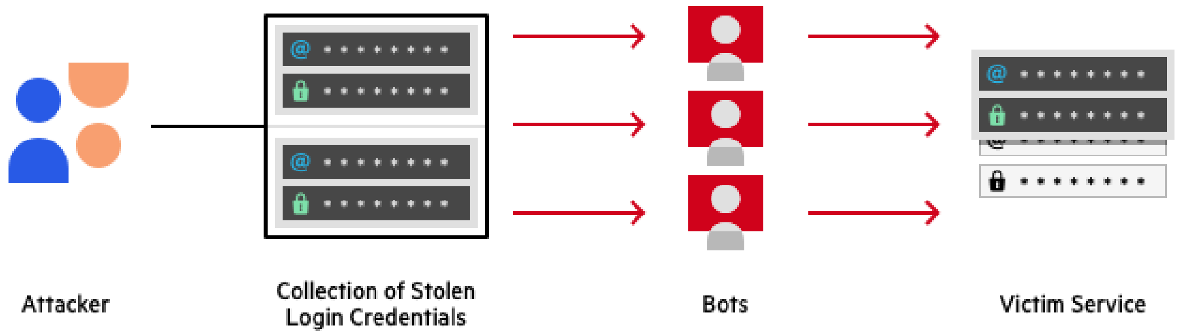
7. **CRLF Injection:** Carriage Return Line Feed (CRLF) injection is a vulnerability where an attacker injects CRLF characters into application input, potentially leading to HTTP header manipulation, HTTP response splitting, or other attacks that exploit the improper handling of newlines.



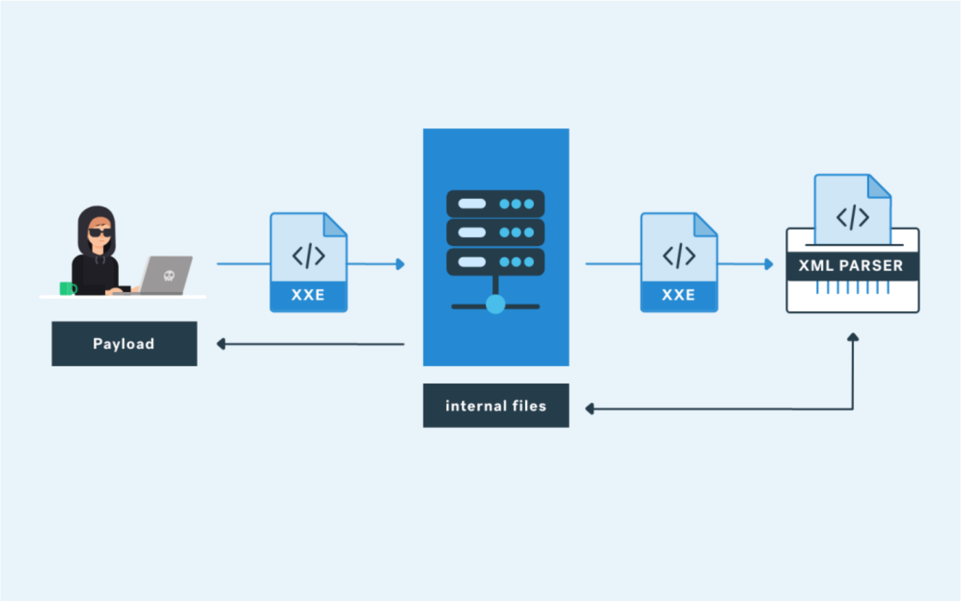
8. Insecure Deserialization:Description: Insecure deserialization occurs when an application processes untrusted serialized data, which can lead to remote code execution, authentication bypass, or other security issues.



9. Credential Stuffing:Credential stuffing occurs when attackers use leaked username and password pairs from one service to gain unauthorized access to other accounts, exploiting users who reuse passwords across multiple platforms.



10. XML External Entity (XXE) Injection: XML External Entity (XXE) Injection is a vulnerability that occurs when an application processes XML input from an untrusted source without proper validation and safeguards. XML (eXtensible Markup Language) is a widely used format for structuring and sharing data between different systems. XXE attacks exploit the way XML parsers process external entities, which are references to data outside of the XML document itself.


XXE
XXE
XML PARSER
Payload
internal files