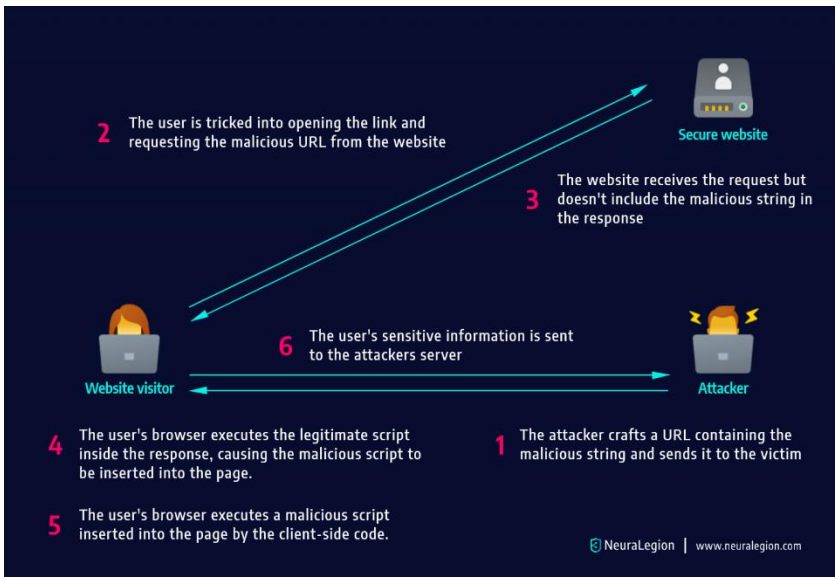
Task – 4

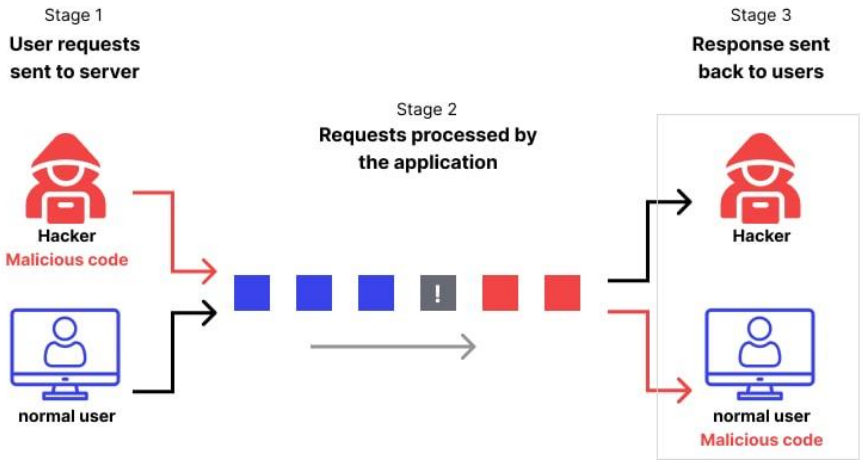
Understanding Web Application Attacks Another 10 attacks

1. **DOM Based Attack**



DOM-based attacks manipulate the Document Object Model (DOM) in web browsers to execute malicious scripts. Attackers exploit client-side vulnerabilities by injecting code that modifies a web page's structure, leading to unexpected behavior. These attacks can steal sensitive information, perform actions on behalf of users, or compromise their privacy. Proper input validation and output encoding are essential to prevent these attacks.

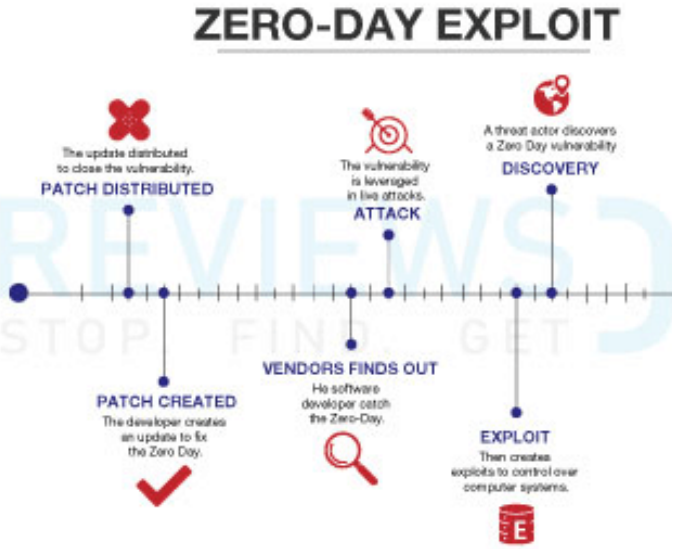
2. **HTTP Request Smuggling**

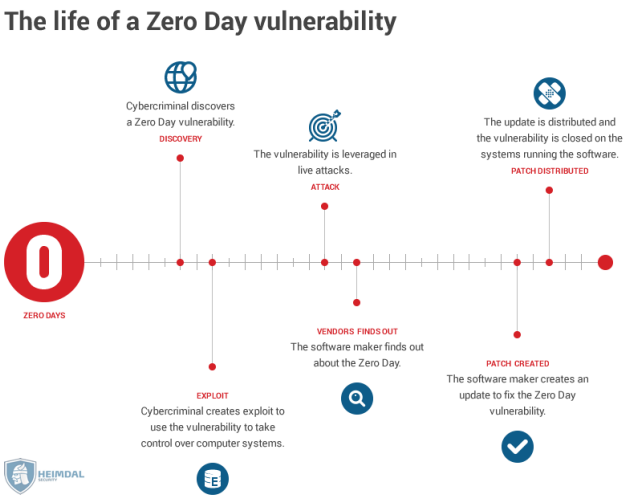


HTTP Request Smuggling is a web attack that exploits inconsistencies in how web servers and proxy servers interpret and handle HTTP requests. By manipulating request headers or content, attackers can cause these servers to misunderstand the sequence of requests, potentially leading to request manipulation or bypassing security controls. This can result in

unauthorized access, data exposure, or other security breaches. Proper server configuration and security mechanisms are crucial to prevent such attacks.

3. Zero day Attacks





Zero-day attacks are cyber attacks exploiting software vulnerabilities that developers aren't aware of or haven't patched. Attackers exploit these unknown weaknesses, bypassing defenses and causing damage before fixes are available. They're highly dangerous due to lack of defense and can lead to data breaches, system control, and malware spread. After the attack only the vendors find out.

4. Server side Request Forgery:

Server-Side Request Forgery (SSRF) is a web vulnerability where an attacker tricks a server into making unauthorized requests to other resources, often within the same network. By exploiting this, attackers can access sensitive information, interact with internal systems, or perform attacks against external entities. SSRF is a significant threat as it can lead to data exposure, remote code execution, and potentially compromise the entire application infrastructure.

5. **Watering Hole Attacks**

A watering hole attack is when attackers compromise websites frequently visited by a specific target group. They inject malicious code into these sites to infect visitors' devices. Once infected, attackers gain access to users' networks or data, exploiting the trust users place in these familiar sites for surreptitious cyberattacks.

6. **Credential Stuffing:**

Credential stuffing is a cyber attack where attackers use stolen username and password pairs from previous data breaches to gain unauthorized access to user accounts on other platforms. By exploiting the tendency of users to reuse passwords across different sites, attackers automate login attempts, potentially leading to account compromise, data breaches, and unauthorized activities.

7. **HTTP Parameter Pollution:**

HTTP Parameter Pollution (HPP) is a web application vulnerability where an attacker manipulates or injects additional HTTP parameters into a web request to confuse or exploit the behavior of the application. This can lead to unexpected behavior, security weaknesses, or application dysfunction. HPP attacks can occur when multiple parameters with the same name are processed by the server, causing conflicts that attackers can abuse to their advantage.

8. **Unvalidated Redirects and Forwards:**

Unvalidated Redirects and Forwards occur when attackers manipulate web application redirects or forwards to trick users into visiting malicious websites or perform unintended actions. By exploiting vulnerabilities in these processes, attackers can lure victims into disclosing sensitive information or engaging in actions they didn't intend, potentially leading to phishing attacks or unauthorized access to accounts.

9. **Serverless Function Attacks**

Serverless function attacks target serverless computing platforms, exploiting misconfigurations, weak permissions, or insecure deployments. Attackers can compromise serverless applications, steal data, execute arbitrary code, or use the platform's resources for malicious purposes. Protecting against such attacks requires thorough security configurations, access controls, and monitoring to ensure the safe operation of serverless functions.

10. **JSON Web Token (JWT) Attacks**

JSON Web Token (JWT) attacks exploit weaknesses in the way JWTs are implemented. Attackers can tamper with token data to impersonate users, escalate privileges, or gain unauthorized access to protected resources. Vulnerabilities can result from inadequate token validation or poor encryption, allowing attackers to manipulate tokens and exploit authentication and authorization flaws in web applications.