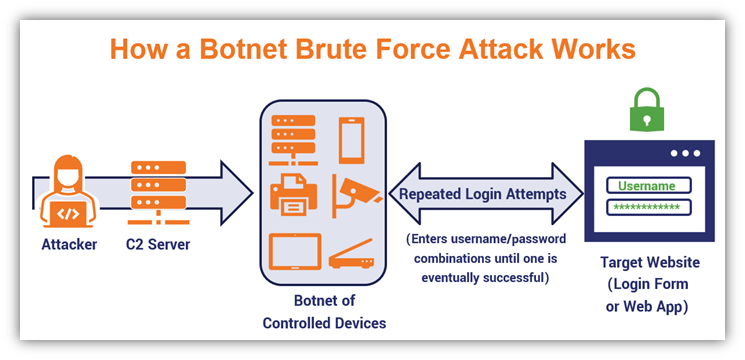
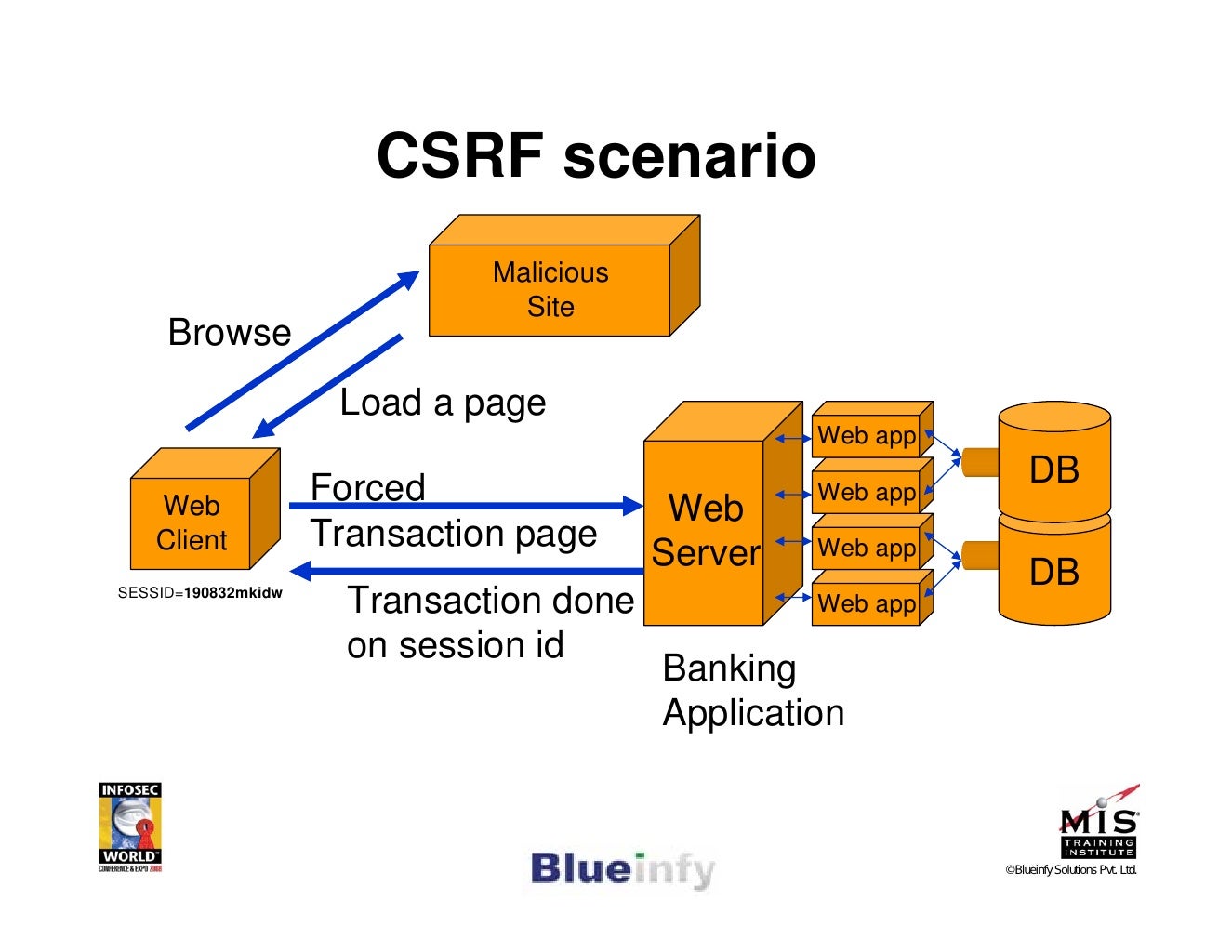
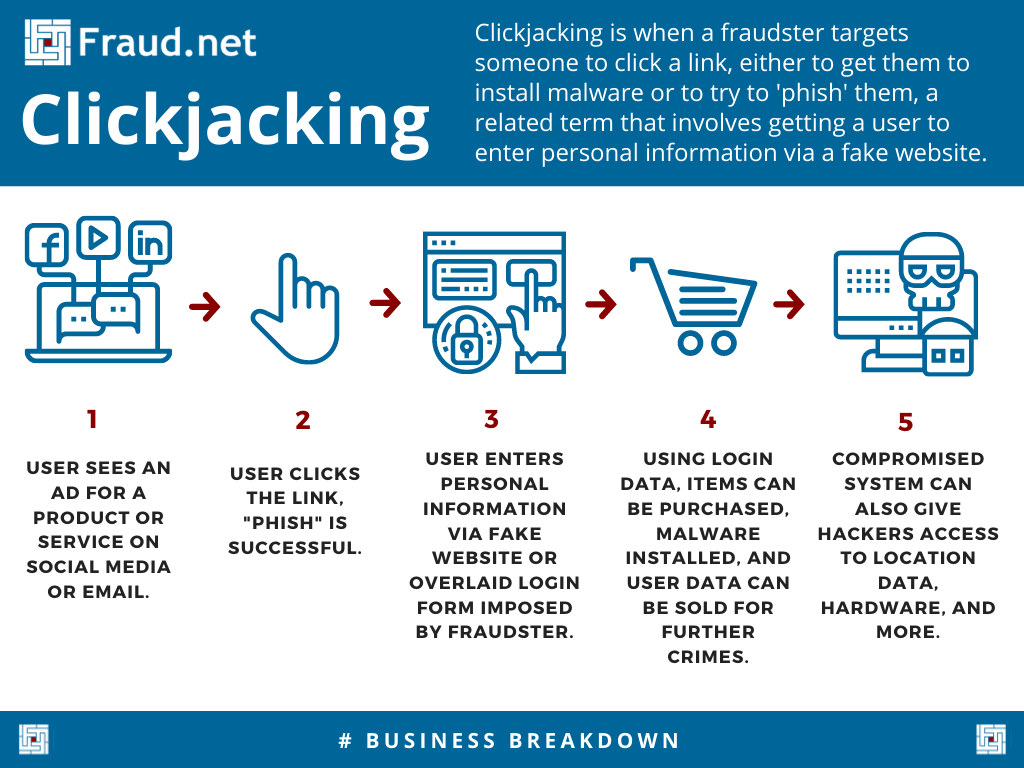
**Brute Force Attack:-** A brute force attack is a hacking method that involves trying every possible combination of characters or inputs until the correct one is found, often used to crack passwords or encryption keys. It is a time-consuming and resource-intensive approach, but it can be effective if security measures are weak.



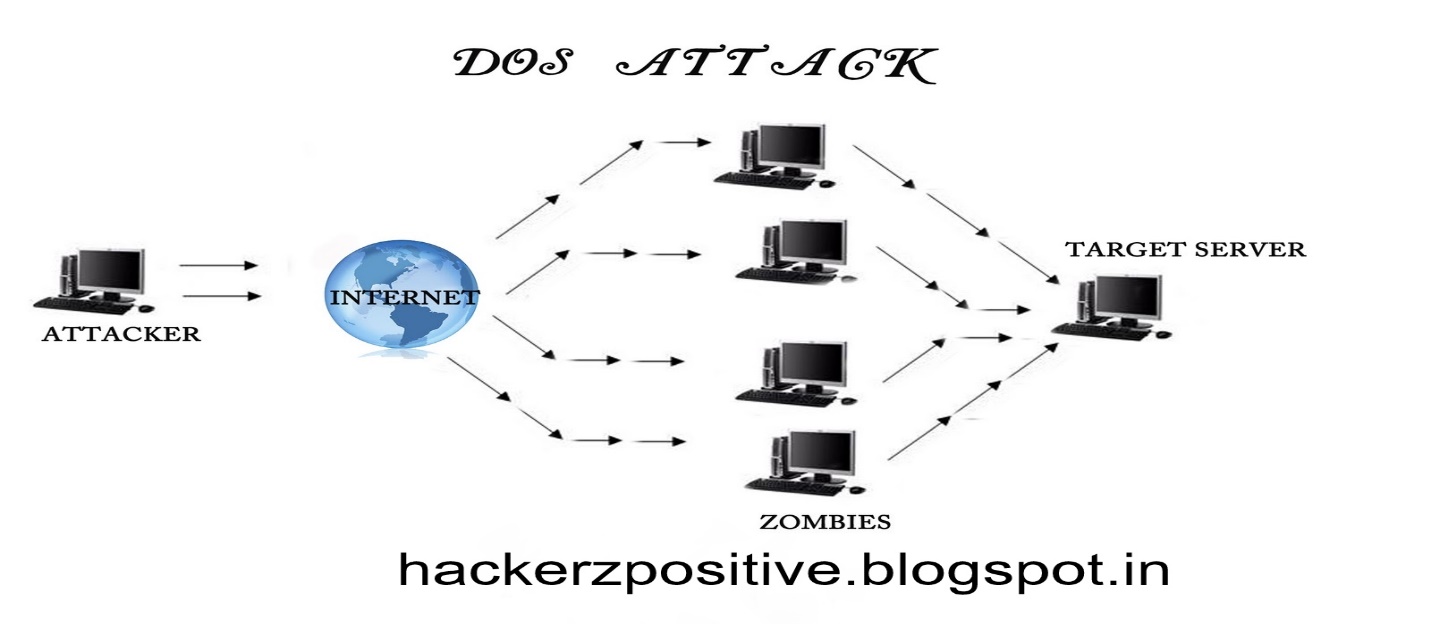
**CRSF Vulnerability**:- Cross-Site Request Forgery (CSRF) is a web security vulnerability where an attacker tricks a user into unknowingly making a malicious request on a trusted website, potentially leading to unauthorized actions being performed on the user's behalf. Preventing CSRF typically involves using tokens to verify the legitimacy of requests and implementing secure session management practices.



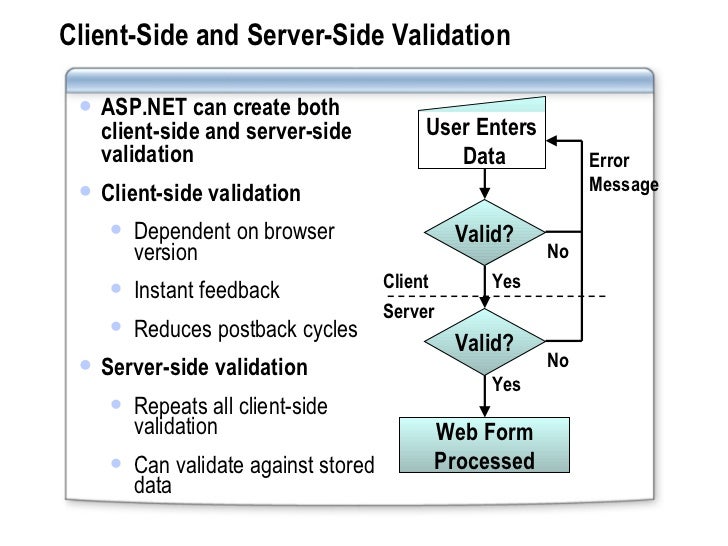
**Clickjacking Vulnerability:** Clickjacking is a web security vulnerability where an attacker conceals a malicious action or link behind a legitimate-looking website element, tricking users into performing unintended actions without their knowledge or consent. To mitigate clickjacking, websites can implement measures like frame-busting scripts and the X-Frame-Options HTTP header to prevent their content from being embedded in malicious frames**.**

o

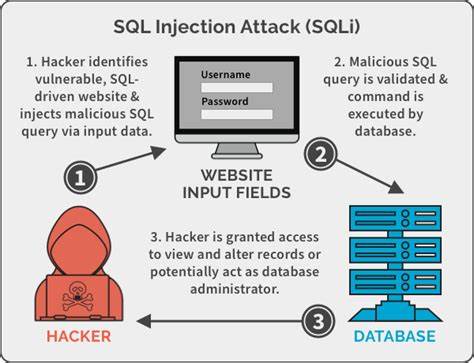
**Long password dos attack** :- A "long password DoS attack" typically involves an attacker attempting to overwhelm a system by submitting an excessively long password, potentially causing resource exhaustion, system slowdown, or service disruption. To mitigate this type of attack, systems can implement password length limits and rate limiting for login attempts, along with robust error handling to prevent excessive resource consumption**.**



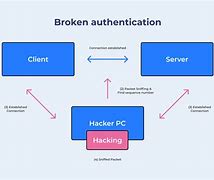
**Client Side Validiation Bypass Vulnerability**:- A "Client-Side Validation Bypass Vulnerability" refers to a security weakness in a web application where the validation checks for user input are only performed on the client side (in the user's browser) rather than being enforced on the server side. Attackers can exploit this by manipulating or bypassing the client-side validation, submitting malicious or unauthorized data to the server. To mitigate this vulnerability, it's crucial to implement strong server-side validation and security controls to prevent unauthorized or malicious input from reaching the server.



**SQL Injection**:- SQL Injection is a type of cyberattack that occurs when malicious actors manipulate user input in a way that allows them to execute arbitrary SQL (Structured Query Language) code on a web application's database. This can lead to unauthorized access, data leakage, data manipulation, or even the complete compromise of a database. To prevent SQL Injection, developers should use prepared statements or parameterized queries, validate and sanitize user inputs, and implement proper access controls and authentication mechanisms in their applications.



**Broken Authencitation** :- Broken Authentication is a security vulnerability that occurs when an application's authentication and session management mechanisms are not properly implemented or configured. This can lead to various security risks, including unauthorized access to user accounts, sensitive data exposure, and session hijacking. To mitigate the risks associated with broken authentication, developers should implement strong authentication methods, use secure session management techniques, enforce proper password policies, and regularly test and audit their application's security controls.



**Rate limit Issues:-** Rate limit issues refer to situations where an application or system doesn't have appropriate rate limiting measures in place. Rate limiting is a crucial security and performance mechanism that restricts the number of requests a user or system can make within a specified time frame. Without adequate rate limiting, an application can become vulnerable to various types of abuse, such as Distributed Denial of Service (DDoS) attacks, brute-force attacks, or excessive API usage.

**Broken access issues** :-Broken access control is a security vulnerability that occurs when a web application doesn't properly enforce restrictions on what authenticated users can do, allowing them to access unauthorized resources or perform actions beyond their privileges. This can lead to unauthorized data access, account takeovers, and other security breaches. To mitigate broken access control, developers should implement strict access controls, enforce proper authentication and authorization mechanisms, and regularly test and audit their application's security to ensure that users can only access what they're supposed to according to their roles and permissions.

