### **Report on 5 Real-World Web Application Attacks**

In this report, we will examine five well-known real-world web application attacks, identify their threats and vulnerabilities, and discuss the affected security pillars: Confidentiality, Integrity, and Availability (CIA). We will also analyze the risks involved, their impact, and propose security best practices for remediation, along with strategies to mitigate associated risks.

### **1. SQL Injection (SQLi)**

#### **Threats, Vulnerabilities, and Affected Security Pillars:**

* **Threats:** Cyberattackers exploit vulnerabilities in SQL query execution on web applications.
* **Vulnerabilities:**
  + Improper input validation and sanitization.
  + Direct user input being inserted into SQL queries without proper escaping.
* **Affected Security Pillars:**
  + **Confidentiality:** Attackers can extract sensitive information (e.g., customer data, passwords, financial records).
  + **Integrity:** Attackers may modify or delete data from the database.
  + **Availability:** Attackers may corrupt or delete data, making the application or service unavailable.

#### **Risk Analysis:**

* **Impact:**
  + **Legal:** Breach of data protection laws (e.g., GDPR, HIPAA) due to unauthorized access to sensitive data.
  + **Financial:** Direct financial loss from stolen credit card information, fines, and lawsuits.
  + **Reputational:** Loss of customer trust due to exposure of sensitive data, leading to a damaged brand reputation.

#### **Remediation Measures:**

* **Use Prepared Statements/Parameterized Queries:** Ensure all SQL queries are executed with bound parameters rather than directly concatenating user input.
* **Input Validation and Sanitization:** Sanitize all user inputs to prevent malicious code from being executed.
* **Limit Database Permissions:** Restrict user privileges to prevent unauthorized access or modification of critical data.

#### **Mitigation Strategies:**

* Implement Web Application Firewalls (WAF) to filter out malicious SQL injection attempts.
* Conduct regular security audits and vulnerability assessments.

#### **Source: OWASP SQL Injection Attack (**[**https://owasp.org/www-community/attacks/SQL\_Injection**](https://owasp.org/www-community/attacks/SQL_Injection)**)**

### **2. Cross-Site Scripting (XSS)**

#### **Threats, Vulnerabilities, and Affected Security Pillars:**

* **Threats:** Malicious scripts injected into web applications to steal information or perform malicious actions.
* **Vulnerabilities:**
  + Failure to properly validate or sanitize user input.
  + Web applications failing to encode content before displaying it in a web browser.
* **Affected Security Pillars:**
  + **Confidentiality:** Attacker can steal session tokens, cookies, or user credentials.
  + **Integrity:** Attackers may alter the content of the application or redirect users to malicious sites.
  + **Availability:** Attackers can exploit XSS for DoS (Denial of Service) by redirecting users to malicious websites.

#### **Risk Analysis:**

* **Impact:**
  + **Legal:** Violation of user privacy and potentially exposing them to identity theft or data theft.
  + **Financial:** Potential lawsuits, fines, and compensation to affected customers.
  + **Reputational:** Erosion of customer trust if the web application is found to be vulnerable to XSS attacks.

#### **Remediation Measures:**

* **Input Sanitization and Validation:** Ensure all user inputs are validated and sanitized before processing.
* **Content Security Policy (CSP):** Enforce policies that control which resources can be loaded and executed on the site.
* **Escape Data in HTML, JavaScript, and CSS Contexts:** Ensure data is properly escaped to prevent malicious code execution.

#### **Mitigation Strategies:**

* Implement a Web Application Firewall (WAF) to detect and block XSS payloads.
* Regularly update and patch web application frameworks to address known vulnerabilities.

#### **Source: OWASP Cross-Site Scripting (XSS) (**[**https://owasp.org/www-community/attacks/xss/**](https://owasp.org/www-community/attacks/xss/)**)**

### **3. Cross-Site Request Forgery (CSRF)**

#### **Threats, Vulnerabilities, and Affected Security Pillars:**

* **Threats:** An attacker forces an authenticated user to perform unintended actions on a web application.
* **Vulnerabilities:**
  + Lack of anti-CSRF tokens to validate requests.
  + Reliance on cookies for authentication without any additional safeguards.
* **Affected Security Pillars:**
  + **Confidentiality:** CSRF could lead to unauthorized actions that compromise the security of user data.
  + **Integrity:** Attackers may modify data on behalf of an authenticated user.
  + **Availability:** Attackers can trigger malicious actions that could disrupt the service.

#### **Risk Analysis:**

* **Impact:**
  + **Legal:** Loss of control over user actions leading to violations of privacy and user trust.
  + **Financial:** Financial loss if attackers perform transactions on behalf of users.
  + **Reputational:** Damage to the organization’s reputation if users' data is misused.

#### **Remediation Measures:**

* **Use Anti-CSRF Tokens:** Ensure that all state-changing requests require an anti-CSRF token.
* **SameSite Cookies:** Set cookies to SameSite to prevent them from being sent in cross-site requests.
* **User Session Protection:** Use session expiration and re-authentication for sensitive actions.

#### **Mitigation Strategies:**

* Regularly test for CSRF vulnerabilities.
* Educate users to always log out after sensitive transactions.

#### **Source: OWASP Cross-Site Request Forgery (CSRF) (**[**https://owasp.org/www-community/attacks/csrf**](https://owasp.org/www-community/attacks/csrf)**)**

### **4. Denial of Service (DoS) and Distributed Denial of Service (DDoS)**

#### **Threats, Vulnerabilities, and Affected Security Pillars:**

* **Threats:** Attackers overwhelm web servers with excessive traffic, making the website or application unavailable.
* **Vulnerabilities:**
  + Lack of rate limiting or throttling.
  + Insufficient server capacity or resilience.
* **Affected Security Pillars:**
  + **Availability:** The main security pillar affected is availability as the service becomes unavailable or slow due to the excessive load.

#### **Risk Analysis:**

* **Impact:**
  + **Legal:** Service outages can violate service-level agreements (SLAs) or breach contracts.
  + **Financial:** The organization can face financial losses from lost business during the downtime.
  + **Reputational:** Persistent outages damage the reputation of the business, potentially leading to customer churn.

#### **Remediation Measures:**

* **Implement Rate Limiting:** Set up rate limiting to prevent excessive requests from overloading the server.
* **Use Content Delivery Networks (CDNs):** Distribute traffic across multiple servers to mitigate DDoS attacks.
* **Deploy Intrusion Detection Systems (IDS):** Monitor and detect anomalous traffic patterns.

#### **Mitigation Strategies:**

* Use cloud-based DDoS protection services like Cloudflare or AWS Shield.
* Set up redundant infrastructure to prevent a single point of failure.

#### **Source: DDoS Attacks: What You Need to Know (**[**https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/**](https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/)**)**

### **5. Insecure Deserialization**

#### **Threats, Vulnerabilities, and Affected Security Pillars:**

* **Threats:** Attackers exploit deserialization vulnerabilities to execute arbitrary code on the server or alter application logic.
* **Vulnerabilities:**
  + Application deserializes user input or data without proper validation.
* **Affected Security Pillars:**
  + **Confidentiality:** Attackers may access sensitive data in the application.
  + **Integrity:** Attackers can modify or inject malicious data.
  + **Availability:** Attackers may cause a denial of service by injecting malicious data.

#### **Risk Analysis:**

* **Impact:**
  + **Legal:** Data breaches and unauthorized access to sensitive information can lead to legal consequences.
  + **Financial:** Financial loss from reputational damage, recovery, and potential fines for non-compliance.
  + **Reputational:** A compromised system could severely damage the trust customers place in the service.

#### **Remediation Measures:**

* **Avoid Serialization of Sensitive Data:** Do not serialize sensitive data or objects that contain authentication details.
* **Validate and Sanitize Data:** Ensure any data being deserialized is validated against a predefined schema.
* **Use Strong Encryption:** Encrypt serialized data to ensure integrity and prevent tampering.

#### **Mitigation Strategies:**

* Implement security controls and cryptographic measures to protect serialized data.
* Use strict data validation and avoid using vulnerable deserialization methods.

#### **Source: OWASP Insecure Deserialization (**[**https://owasp.org/www-project-top-ten/2017/A8\_2017-Insecure\_Deserialization**](https://owasp.org/www-project-top-ten/2017/A8_2017-Insecure_Deserialization)**)**

### **Conclusion**

Each of these real-world web application attacks presents distinct threats and risks to organizations, including severe legal, financial, and reputational consequences. Implementing best practices such as input validation, secure coding practices, rate limiting, and using appropriate security controls can mitigate these risks and safeguard both the confidentiality, integrity, and availability of the systems involved. Regular security assessments and timely patching are essential to stay ahead of emerging vulnerabilities.