

OWASP Top 10 Web Application Security Risks 2021

The OWASP Top 10 is a powerful tool to highlight the most critical web application security risks. The Open Web Application Security Project (OWASP) publishes the list to help developers and security professionals address common vulnerabilities in their web applications. Below are the top 10 risks identified for 2021, including detailed descriptions and suggested mitigations.

**A01:2021 - Broken Access Control**

Description: Access control issues occur when users can access data or functionality they are not authorized to. This may allow users to perform unauthorized actions, such as viewing sensitive data or modifying records.

Examples:

- Users accessing admin functionalities without proper authorization.

- URL manipulation to access other users' resources (e.g., changing the user ID in the URL to access another user's profile).

Mitigations:

- Use role-based access control (RBAC) to restrict access to certain resources.

- Validate access controls at the backend, not just the frontend.

**A02:2021 - Cryptographic Failures**

Description: This risk occurs when sensitive data, such as passwords or financial details, is not properly protected using strong encryption methods.

Examples:

- Storing passwords in plaintext or using weak encryption.

- Transmitting sensitive data over non-secure channels (e.g., HTTP instead of HTTPS).

Mitigations:

- Encrypt sensitive data both in transit (e.g., TLS) and at rest (e.g., AES-256).

- Ensure strong key management practices.

**A03:2021 - Injection**

Description: Injection flaws, such as SQL injection, occur when untrusted data is passed to an interpreter as part of a command or query. This can lead to arbitrary code execution or unauthorized access to data.

Examples:

- SQL injection: SELECT \* FROM users WHERE username = '' OR 1=1;

- Command injection: Executing arbitrary system commands due to improper validation.

Mitigations:

- Use prepared statements and parameterized queries.

- Validate and sanitize user input to prevent injection.

**A04:2021 - Insecure Design**

Description: Insecure design refers to flaws in the architecture or design of a web application, which can lead to vulnerabilities that attackers can exploit.

Examples:

- Poorly designed authentication mechanisms.

- Lack of proper error handling leading to exposure of system details.

Mitigations:

- Perform threat modeling and security design reviews.

- Adopt secure design principles and best practices.

**A05:2021 - Security Misconfiguration**

Description: Security misconfiguration occurs when web applications are not securely configured, leading to exposed services, default settings, or unnecessary services that attackers can exploit.

Examples:

- Leaving default credentials enabled.

- Exposing error messages that reveal server information.

Mitigations:

- Perform regular security audits and configuration checks.

- Remove unnecessary services and ensure secure default configurations.

**A06:2021 - Vulnerable and Outdated Components**

Description: This risk involves using outdated or unpatched components with known vulnerabilities in the web application stack, which can be exploited by attackers.

Examples:

- Using old versions of software libraries with known security flaws.

- Using unsupported versions of programming frameworks or components.

Mitigations:

- Regularly update and patch components.

- Use automated tools to track vulnerabilities in components.

**A07:2021 - Identification and Authentication Failures**

Description: Identification and authentication failures occur when attackers can bypass authentication mechanisms or use weak passwords to impersonate users.

Examples:

- Weak passwords or poor password management.

- Failure to implement multi-factor authentication (MFA).

Mitigations:

- Enforce strong password policies.

- Implement multi-factor authentication (MFA) where possible.

**A08:2021 - Software and Data Integrity Failures**

Description: This risk involves the failure to properly validate or protect software and data, which can lead to manipulation or corruption of application data.

Examples:

- Allowing the upload of untrusted files or executable code.

- Failure to validate data integrity before processing.

Mitigations:

- Implement integrity checks like digital signatures or hashes.

- Verify uploaded files' authenticity and integrity.

**A09:2021 - Security Logging and Monitoring Failures**

Description: This risk refers to the lack of effective logging and monitoring of web application activities, which can prevent early detection of security breaches.

Examples:

- Failure to log critical user actions or system errors.

- Not monitoring for unusual or unauthorized access patterns.

Mitigations:

- Implement proper logging for all sensitive actions and events.

- Use security monitoring systems to detect and respond to potential threats.

**A10:2021 - Server-Side Request Forgery (SSRF)**

Description: Server-Side Request Forgery (SSRF) occurs when an attacker can manipulate a server into making requests to internal resources, potentially leading to unauthorized access.

Examples:

- Attacker forces the server to make requests to internal systems, leading to data exposure.

- Exploiting SSRF to perform attacks on internal services.

Mitigations:

- Implement input validation and allowlisting of URLs.

- Block outgoing requests to internal and sensitive systems.