Web Security Fundamental Concepts

Owner: Nii Nartey

Reviewer: Thomas Darko

Introduction to Web Application Security

Modern web development presents numerous challenges, and among these, security stands out as both crucial and frequently underemphasized. Web security encompasses the protection of web applications and network infrastructures, ensuring that information and resources are safeguarded against unauthorised access and misuse. Any organisation exposing its computing services to network access must prioritise security to protect sensitive information.

The internet, while a primary communication channel, also introduces significant risks related to access and misuse of services and information. Web application security is therefore essential to mitigate these risks and maintain the integrity of applications and data.

Key Aspects of Web Application Security

In the context of web applications, security can be divided into several key areas:

- **Availability:** Ensures that authorised entities or processes can access the application when required.
- **Authenticity:** Confirms that an entity is who they claim to be, and guarantees the source of the data.
- **Integrity:** Ensures that information assets are not altered in an unauthorised manner.
- **Confidentiality:** Protects sensitive information from being accessed or disclosed to unauthorised individuals, entities, or processes.
- **Traceability:** Ensures that the actions of an entity can be attributed exclusively to that entity.

Web application security refers to the measures and practices employed to protect web applications from unauthorised access, data breaches, and other malicious activities. This involves mitigating vulnerabilities and addressing threats throughout the software development life cycle.

Common Web Application Vulnerabilities

Before diving into the specifics of web application security, it's essential to understand the most prevalent vulnerabilities that threaten applications. The **OWASP Top 10** is a crucial resource in this regard.

OWASP Top 10

The OWASP (Open Web Application Security Project) is a non-profit organization dedicated to improving software security. The OWASP Top 10 is a regularly updated list of the most critical security risks to web applications, based on data from real-world attacks and vulnerabilities. The latest update was in 2021.

This list highlights vulnerabilities that require immediate resolution in any software. It serves as a guide for developers and security professionals to identify and address common threats. Existing code should be reviewed for these vulnerabilities, as attackers actively target such flaws.

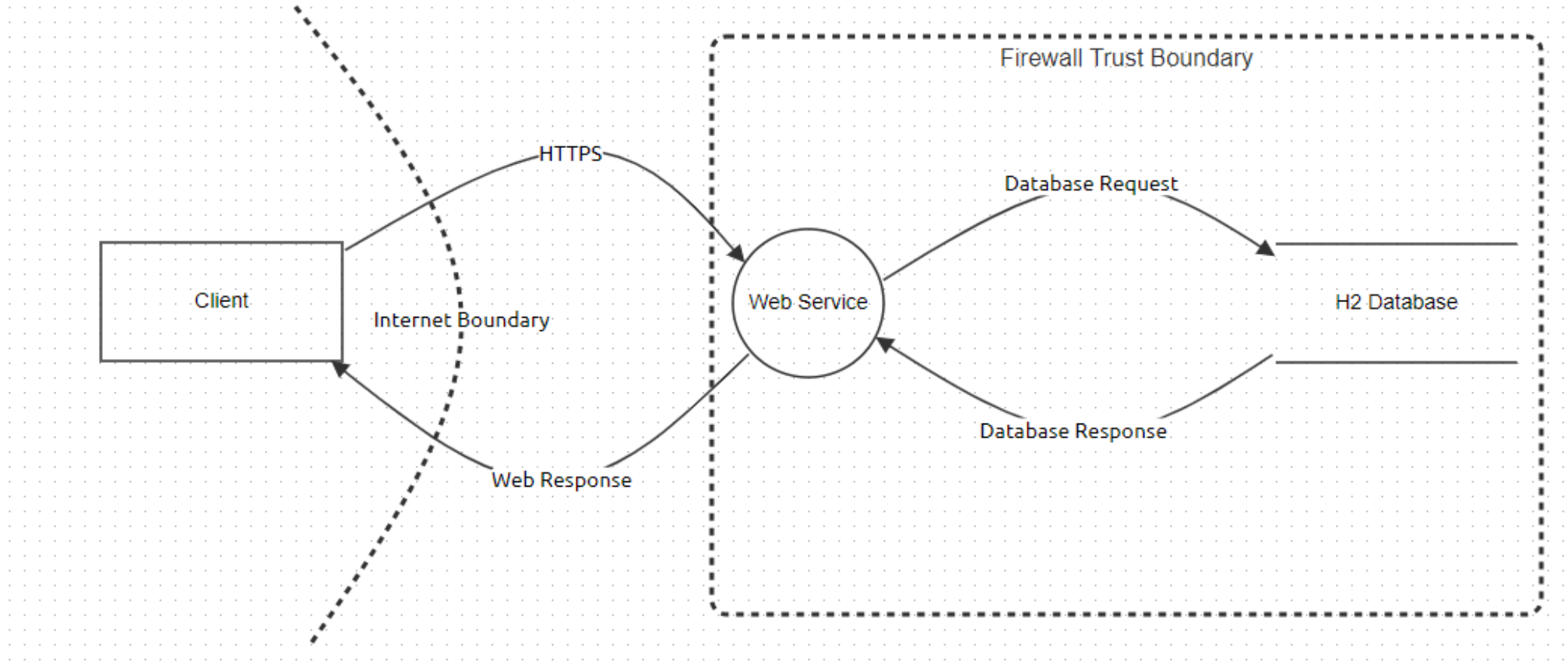
In development projects, these vulnerabilities should be checked in design documents, and the code should be tested during the development phase. Regular testing ensures that no new vulnerabilities are introduced and that any existing vulnerabilities are addressed.

Threat Identification and Mitigation

Critical to securing a web application is the identification of potential threats. One widely used methodology for threat categorization is STRIDE. This approach helps in identifying threats from an attacker's perspective.

In the documentation below, we demonstrate threat identification and mitigation for a web application with the following components:

- A Client Server
- A Backend Web Server
- A Database



High-Level System Description

This system is a comprehensive software solution designed to assist businesses in managing their interactions with current and potential customers. The platform provides tools for tracking customer interactions, managing leads, and automating sales processes, ultimately helping businesses improve their customer relationships and increase sales efficiency.

Key Components:

- **Client Server:** The interface through which users interact with the system. It handles user inputs and communicates with the backend to process and display data.
- **Backend Web Server:** This component manages the application logic, processes requests from the client server, interacts with the database, and ensures that data is processed and returned to the client in a secure manner.
- **Database:** The repository where all customer data, interaction logs, and system configurations are stored. It is a critical component, as it houses sensitive information that must be protected from unauthorised access and breaches.

Threat Summary

In the process of securing this system, a thorough threat analysis was conducted, identifying potential risks and implementing mitigation strategies. The following is a summary of the threat analysis:

Total Identified Threats: 21

Total Mitigated Threats: 17

Total Unmitigated Threats: 3

C.M.S High-Level Architecture and Threat Model

Client (Actor)

Description: This is the client server that the user interacts with to make requests on the customer management application.

Title	Type	Priority	Status	Description	Mitigations
Imperson ation by Spoofing	Spoofing	High	Mitigated	Hackers may impersonate the user entity on the system	Use Secure Tokens: Ensure that tokens used for authentication and authorization are securely generated, stored, and transmitted.

Web Service (Process)

Description: The backend process handling application logic, receiving and responding to client requests.

Title	Type	Priority	Status	Description	Mitigations
Potential Data Repudiation by Web Service	Repudiati on	High	Mitigated	The Web Service could claim it did not receive data from a source outside the trust boundary	Consider using logging or auditing to record the source, time, and summary of the received data.
Repudiation Threat	Denial of service	Medium	Mitigated	Ensuring the Web Service controls resource consumption to prevent denial of service	Rate Limiting: Implement rate limiting to control the number of requests a user can make in a given time frame. Load Balancing: Distribute traffic across servers.
Weak Credential Storage	Informati on disclosure	High	Mitigated	Credentials stored on the server are often disclosed	Store a salted hash of credentials instead of plain text. If not possible, encrypt credentials before storage using an SDL-approved mechanism.
Elevation Using Impersonati on	Elevation of privilege	Medium	Mitigated	The Web Service might impersonate a Human User to gain additional privileges	Use Secure Tokens: Ensure tokens are securely generated, stored, and transmitted.

Elevation by Changing Execution Flow	Elevation of privilege	High	Mitigated	An attacker might manipulate data to alter the execution flow within the Web Service	Validate Inputs: Ensure all inputs are validated to prevent injection attacks that could alter program flow.
Footprinting	Information disclosure	Medium	Mitigated	Information gathering aimed at learning the composition, configuration, and security mechanisms	Shut down unnecessary services/ports, and exclude sensitive information that could compromise the organisation's security.

---

h2 Database (Store)

Description: The database stores all critical customer data, interaction logs, and system configurations.

Title	Type	Priority	Status	Description	Mitigations
Scraping	Information disclosure	High	Mitigated	Collecting accessible data or processed output from the application	ensure authorization
Skewing	Elevation of privilege	Medium	Not Mitigated	Automated actions affecting application-based metrics such as counts and frequency	Control interaction frequency, enforce a single unique action, and properly enforce metrics.
Spamming	Elevation of privilege	Medium	Not Mitigated	Storing malicious content such as malware, Iframes, advertisements, and tracking code	Detect embedded malicious code, control interaction frequency, and enforce a single unique action.

HTTPS (Data Flow)

Description: Secures the data transmission between the client and the web service.

Title	Type	Priority	Status	Description	Mitigations
Generic Tampering	Tampering	High	Mitigated	A generic tampering threat, such as SQL injections	Validate and sanitise all user inputs. Implement allow-lists to ensure only expected inputs are processed, e.g., ensuring email fields only accept valid emails.
Generic DoS	Denial of service	Medium	Mitigated	An external agent interrupts data flow across a trust boundary in either direction	Configure firewalls to block malicious traffic. Implement rate limiting to prevent overwhelming the system with excessive requests.

Web Response (Data Flow)

Description: Handles the responses sent back to the client after processing requests.

Title	Type	Priority	Status	Description	Mitigations
Generic DoS	Denial of service	High	Mitigated	An external agent interrupts data flow across a trust boundary in either direction	Configure firewalls to block malicious traffic. Implement rate limiting to control the number of requests a user can make.
Vulnerable Transport Protocol	Information disclosure	Medium	Mitigated	Older transport protocols are vulnerable and have known vulnerabilities	Use up-to-date cryptography and transport protocols.

Database Request (Data Flow)

Description: Handles communication between the application and the database.

Title	Type	Priority	Status	Description	Mitigations
Vulnerable Transport Protocol	Information disclosure	Medium	Mitigated	Older transport protocols are vulnerable and have known vulnerabilities	Use up-to-date cryptography and transport protocols.

Firewall (Generic Trust Border Boundary)

Description: Secures the network boundary between internal systems and external networks, protecting against unauthorised access and data breaches.

Title	Type	Priority	Status	Description	Mitigations
Vulnerable Transport Protocol	Information disclosure	Medium	Mitigated	Older transport protocols are vulnerable and have known vulnerabilities	Use up-to-date cryptography and transport protocols to ensure secure communication.
Spamming	Elevation of privilege	Medium	Mitigated	Storing malicious content such as malware, Iframes, advertisements, and tracking code	Detect embedded malicious code, control interaction frequency, and enforce a single unique action.
Elevation Using Impersonation	Elevation of privilege	Medium	Mitigated	The Web Service may impersonate a Human User to gain additional privileges	Use Secure Tokens: Ensure tokens for authentication and authorization are securely generated, stored, and transmitted.
Elevation by Changing Execution Flow	Elevation of privilege	High	Mitigated	An attacker might manipulate data to alter the execution flow within the Web Service	Validate Inputs: Ensure all inputs are validated to prevent injection attacks that could alter program flow.
Footprinting	Information disclosure	Medium	Mitigated	Gathering information about the application's composition, configuration, and security mechanisms	Shut down unnecessary services/ports and exclude information that could identify and compromise the organisation's security.

Conclusion

Web application security is a complex and critical topic that requires continuous attention and effort. By understanding the fundamentals, common vulnerabilities, and best practices, you can significantly improve the security posture of your web applications. Implement a holistic approach that incorporates secure development practices, regular testing, and ongoing monitoring to safeguard your applications and protect your users’ data.

The OWASP Top 10 is a widely recognized and referenced document in the cybersecurity industry. It provides a framework of the top 10 most critical web application security risks that organisations should be aware of and address. Understanding the OWASP Top 10 is important for developers, security professionals, and decision-makers to assess and prioritise security risks within their organisation’s web applications. By addressing the OWASP Top 10, organisations can strengthen their security posture and reduce the likelihood of successful attacks against their web applications.

Sources :

Spring Documentation :

- OWASP Top 10
- spring security reference documentation
- spring security reference expression language
- spring security reference headers

Articles & blogs :

- Nagarjun Nagesh