Web Application Security

SQL injection and Cross-site scripting (XSS)

Techniques to detect and prevent these attacks

Examples of tools or methods you would use

1. SQL Injection (SQLi)

SQL injection allows attackers to execute malicious SQL queries on a database, often resulting in unauthorized access to data.

Detection Techniques:

- **Manual Code Review:** Review the code where user inputs interact with the database, especially where dynamic SQL queries are constructed. Look for concatenation of user inputs without proper sanitization.
- **Automated Vulnerability Scanners:**
  - **SQLMap:** A popular open-source tool that can automatically detect and exploit SQL injection vulnerabilities.
  - **OWASP ZAP (Zed Attack Proxy):** This tool can be used to find potential SQL injection points in web applications by simulating attacks.
  - **Burp Suite:** A comprehensive platform that offers automated and manual tools for detecting SQLi vulnerabilities, including the ability to modify HTTP requests.

Prevention Techniques:

- **Parameterized Queries (Prepared Statements):** Instead of dynamically constructing SQL queries by concatenating user inputs, use parameterized queries to bind inputs at runtime. For example, in Python with `MySQLdb`:
- `cursor.execute("SELECT * FROM users WHERE username = %s", (username,))`
- **ORMs (Object-Relational Mappers):** Using ORM frameworks like Django ORM or SQLAlchemy can abstract direct database queries and reduce the likelihood of SQL injection by automatically handling input sanitization.
- **Input Validation:** Ensure strict validation on user inputs. For example, reject unexpected input types, lengths, or patterns.
- **Least Privilege:** Limit the database permissions for the web application. For example, an application user shouldn't have admin privileges if it's not necessary.
- **Web Application Firewall (WAF):** A WAF like **ModSecurity** can detect and block SQLi attempts by inspecting HTTP requests for malicious patterns.

2. Cross-Site Scripting (XSS)

XSS allows attackers to inject malicious scripts into a web page, potentially leading to session hijacking, defacement, or redirection to malicious websites.

Detection Techniques:

- **Manual Testing:** Inspect the areas of the application that reflect user input back to the browser (e.g., comment fields, search bars). Check whether the input is being sanitized or encoded before rendering in the browser.
- **Automated Vulnerability Scanners:**
  - **OWASP ZAP and Burp Suite:** Both tools can be used to identify XSS vulnerabilities by automatically injecting scripts into input fields and observing if they get executed.
  - **Arachni:** This tool also provides a comprehensive scan for XSS and other web application vulnerabilities.

Prevention Techniques:

- **Output Encoding:** Use context-specific output encoding to ensure that user inputs are treated as data and not executable code. For example:
  - **HTML encoding** for user inputs displayed in HTML (<, >, `"`, `'` become `&lt;`, `&gt;`, `&quot;`, `&#x27;`).
  - **JavaScript encoding** for data inside JavaScript.
  - **CSS encoding** for inputs used within CSS styles.
- **Input Validation and Sanitization:** Ensure inputs are validated for expected types and sanitize any untrusted input by removing or escaping dangerous characters.
- **Content Security Policy (CSP):** Implement CSP headers to restrict the sources from which scripts can be executed. For example:

  http

```
Content-Security-Policy: default-src 'self'; script-src 'self'
```

This ensures that only scripts from trusted sources (such as the same domain) are executed.

- HTTP-**Only Cookies:** To protect against session hijacking, set cookies as `HttpOnly` and `Secure`, preventing JavaScript access to session tokens:

```
Set-Cookie: sessionID=abc123; HttpOnly; Secure;
```

3. Tools and Methods for Both Vulnerabilities

a) Security Testing Frameworks:

- **OWASP Zap and Burp Suite (Pro version):** These tools provide both automated and manual testing capabilities to detect vulnerabilities like SQL injection and XSS. They allow for crawling the application, intercepting requests, and injecting payloads to test for weaknesses.

b) Source Code Scanners:

- **SonarQube:** A static analysis tool that can be integrated into CI/CD pipelines to scan the source code for vulnerabilities, including SQLi and XSS.
- **Checkmarx:** A tool that automatically scans code repositories for security flaws and suggests remediation techniques.

c) Continuous Monitoring and Patching:

- **Intrusion Detection Systems (IDS):** Deploying IDS tools like **Snort** or **Suricata** can help monitor traffic for patterns of SQL injection or XSS attempts, enabling quick identification and response to attacks.
- **Regular Patching:** Ensure that the web application framework, database management systems, and third-party libraries are regularly patched to protect against newly discovered vulnerabilities.

4. Real-World Examples

- **SQL Injection Example:** The 2014 breach of **Nando's** where attackers exploited SQL injection to extract sensitive customer data, highlighting the importance of proper input handling and database security.
- **XSS Example:** In 2019, a major **eBay** vulnerability was discovered that allowed attackers to inject malicious scripts via listing descriptions, compromising user sessions. This emphasizes the need for proper input sanitization and output encoding.

Reconnaissance and its Types

Penetration testing scenario, gathering information about a target is crucial for identifying vulnerabilities and understanding the security posture of the organization. This process typically involves two main phases: passive reconnaissance and active reconnaissance.

1. Passive Reconnaissance

Passive reconnaissance involves gathering information without directly interacting with the target systems. The goal is to collect as much data as possible without alerting the target.

Techniques and Tools:

- **WHOIS Lookup:**
  - **Description:** This tool provides information about domain registration, including the owner's contact details, registration dates, and nameservers.
  - **Steps:**
    1. Perform a WHOIS lookup on the target domain to retrieve registration details.
    2. Analyze the data to identify potential weaknesses (e.g., outdated registration info).
- **Shodan:**
  - **Description:** Shodan is a search engine for internet-connected devices. It helps in finding devices that are publicly accessible.
  - **Steps:**
    1. Search for the target's IP address or domain name in Shodan.
    2. Review the results to identify any exposed devices or services.
- **Google Dorking:**
  - **Description:** This technique uses advanced search queries on Google to find specific information about the target.
  - **Steps:**
    1. Use specific search operators (e.g., `site:targetdomain.com filetype:xls`) to uncover sensitive documents.
    2. Collect information from indexed pages that may reveal security weaknesses.
- **Social Media and OSINT (Open Source Intelligence):**
  - **Description:** Collecting information from social media, news articles, and forums.
  - **Steps:**
    1. Search platforms like LinkedIn, Twitter, and Facebook for employee information, organizational structure, and technologies in use.
    2. Use tools like Maltego for visualizing relationships and gathering intelligence.

Benefits of Passive Reconnaissance:

- Leaves no footprint or direct interaction with the target.
- Provides valuable insights into the organization's structure, technologies, and potential vulnerabilities.

2. Active Reconnaissance

Active reconnaissance involves direct interaction with the target systems to gather information. This phase can be more intrusive and may alert the target to the testing activities.

Techniques and Tools:

- **Nmap**:
  - **Description:** Nmap is a powerful network scanning tool used to discover hosts and services on a network.
  - **Steps:**
    1. Conduct a network scan using Nmap to identify live hosts and open ports

       ```
       nmap -sS -sV -O target-ip
       ```

    2. Analyze the output for open ports and running services to determine potential attack vectors.

Metasploit:

- **Description:** Metasploit is a penetration testing framework that includes various tools for vulnerability scanning and exploitation.
- **Steps:**
  1. Use Metasploit to perform service version detection and vulnerability scanning:

     ```
     use auxiliary/scanner/http/http_version
     set RHOSTS target-ip
     run
     ```

  2. Review the findings to identify known vulnerabilities associated with the detected services.

Nikto:

- **Description:** Nikto is a web server scanner that detects vulnerabilities, outdated software, and configuration issues.
- **Steps:**
  1. Scan the target web application using Nikto

     ```
     nikto -h http://target-website.com
     ```

Examine the report for potential vulnerabilities, such as misconfigurations or exposed sensitive files.

Benefits of Active Reconnaissance:

- Provides detailed insights into the target's systems and services.

- Enables the identification of specific vulnerabilities that can be exploited.