OWASP

Introduction

OWASP - The Open Web Application Security Project is a non-profit foundation which works to improve the security of network. It provides tools and resources to help build more secure application. It also holds conferences and training where we can learn the lastest security trends.

OWASP Top 10

One major project that OWASP is working is OWASP Top 10. The foundation gathers information and feedback from a team of security experts all over the world; and produces a referential document outlining the 10 most critical security concerns for web application security. The latest version of OWASP Top 10 is finished in 2021. Below are a few categories are listed on the OWASP Top 10 2021.

A01 - Broken Access Control

Access control means the application will not allow users to access resources or APIs outside of their intended permissions. Broken access control could allow attacker to view, modify or even delete the data in application system. Common access control vulnerabilities includes:

- URL missing access control mechanism
- Not following the least privileges principle or deny access by default
- Metadata manipulation to elevate privileges

Example attack

The application has the following two URLs:

```
https://example.com/app/getAppInfo
https://example.com/app/adminPortal  <-  this should only be allowed if user is authorized as admin
```

If proper access control is missing, attacker may simply modify the URL to get access to webiste's admin portal.

How to prevent

- Implement access control mechanism and re-use it throughout the application
- Except for public resources, deny access by default
- Log access failure and alert when appropriate
- Stateful session identifiers should have time-out mechanism. Stateless token should be rather short-lived.

A02 - Cryptographic Failure

For data which needs protection in transit or at rest according to local regulation, for example - PII or credit card number, we will need to use cryptographic tools to encrypt, transfer and store them. If the cryptographic tools are broken or the data is not encrypted at all, we may have data breach risk. Common vulnerabilities includes:

- Transit sensitive data in plaintext. Like using HTTP, SMTP, FTP protocol is dangerous for external internet traffic and even internal traffic.
- Usage of outdated or vulnerable cryptographic algorithm or protocol. (e.g. SHA-1 is outdated and proven not secure anymore)
- The server certificate or its trust chain not properly validated

Example attack

An application uses database's automatic encryption to store credit card information. However this data will be decrypted when retrieved, so attacker can use SQL injection to retrieve credit card in plain text.

(We have also covered a lot of methods to crack cryptographic tools in this course such as length extension attack)

How to prevent

- Identify which data is sensitive according to law and regulation
- Don't store sensitive data unnecessarily
- Encrypt all sensitive data at rest
- Check all cryptographic algorithms or protocols are up-to-date in a regular basis

A08 - Software and Data Integrity Failures

This relates to code or infrastructure that does not protect against integrity violation. A common case is an insecure CI/CD pipeline where the CI/CD tool does not verify the signature of application code and attacker may upload their own code to application server without being noticed. Another example is objects or data being serialized to a structure that attacker can modify, which leads to insecure deserialization.

Example attack

Many routers or device firmware doesn't verify updates with their signature, and this is a growing target for attackers.

Another well-known attack is SolarWinds Orion attack. Attacker was able to bypass SolarWinds company's integrity check process and inject their backdoor code to SolarWinds Orion product. This malicious product then got distributed to over 18000 companies and was used to monitor and steal data from these companies which includes many government organization. This is also by-far the most serious supply-chain attack.

How to prevent

- Use digital signature to verify the source
- Ensure libraries and dependencies are from trusted repositories
- Use supply-chain security tool to verify libraries don't have known vulnerabilities
- Enforce code review before any code can be pushed to repositories

There are also other categories in OWASP Top 10. The official report covers their description, how to prevent these kinds of attack, and provides several example scenarios for illustration. For each category, it also has a list of references for how attacks are being done in details and how we can configure our applciation to avoid these attacks. OWASP also has API Top 10 and Mobile Top 10 whose interests are more focused on API development or Mobile development.

OWASP tools

Apart from OWASP Top 10, it also has several projects that provides security related tools. We can use them to improve our application's security.

Dependency-Track

As we mentioned in the SolarWinds example in last section, supply-chain attack can have severe impact and it is also very difficult to discover.

Dependency-Track is a tool that helps organization to analyze their libraries and dependencies and reduce supply-chain risks. It leverages the capabilities of SBOM(Software Bill of Material, it's like a tree structure describing what libraries are used in the application, and recurisively what dependencies are used in these libraries) and scans through all dependencies to check for vulnerabilities and identify risks.

Juice Shop

Juice shop is a sample insecure applicaiton which can be used in security training, competitions or just a sandbox for security experiments. It has all the vulnerabilities from OWASP Top 10 and many other security flaws found in real world application. You can try to hack it to get better understanding of each type of attack.

ZAP

ZAP is a web app scanner which can be used to monitor web app's vulnerabilities and perform auto-testing. According to its documentation, the security testing is usually broken out to:

- **Vulnerability Assessment** - scanned the system and analyzed it for security issues
- **Penetration Testing** - simulate attack from malicious attackers and monitor the system
- **Runtime Testing** - security testing from end-users
- **Code Review** - review the system code in detail looking specifically for security vulnerabilities

ZAP is mainly a tool for automated penetration testing (also called pentesting). It acts as a "man-in-middle-proxy". It can intercept and modify the requests sent from tester's browser to the web applicaiton, performing automated attacks based on tester's request. ZAP also has a scanning tool which scans for common vulnerabilities (e.g. HTTP protocol usage, CRLF injection) on all web pages.

Web Security Testing Guide

This project is to create a comprehensive guide and testing framework framework of how to test if a web applicaiton is secure or not. It covers many aspects of the application including: authentication, deployment management, session management, input validation, etc.

In the appendix of the guide, it lists many tools that can be used for security testing. Some tools which may relate to JVM are:

- **SpotBugs** - is a bug finder for Java programs. It looks for bug patterns that are likely to have errors
- **SonarQube** - is another code quality tool which can analyze the code and identify potential risks. Dependency-track and ZAP reports can also be integrated into SonarQube so the organization can have one place to monitor all risks
- **Selenium** - is a multi-language web driver. It enable developers and testers to emulate interaction with web pages and execute scripts using GUI-free code. It not only supports Java, but also other languages like Python, Ruby, Kotlin

Summary

OWASP is an organization which provides security development and testing standards for web applications. On top of that, it implements its open-source testing and monitoring tools like ZAP or Dependency-Track. We can definitely make use of these tools and guides to help our web application become more secure and avoid potential attacks.