OWASP TOP 10 VULNERABILITIES

The Open Web Application Security Project **(OWASP)** is an open-source application security community whose goal is to spread awareness surrounding the security of applications, best known for releasing the industry standard OWASP Top 10.

The OWASP community is powered by security knowledgeable volunteers from corporations, educational organizations, and individuals from around the world. This community works to create freely-available articles, methodologies, documentation, tools, and technologies. The OWASP Foundation is a 501(c)(3) charitable organization that supports and manages OWASP projects and infrastructure.

Although the community supports the informed use of security technology, OWASP is not affiliated with any technology company, allowing them to provide high quality information without bias. OWASP advocates approaching application security by considering the people, process, and technology dimensions.

What is the OWASP Top 10?

Every few years, OWASP releases the OWASP Top 10, a list of the Top 10 most critical application security risks faced by developers and organizations, with a goal of helping developers and security teams better secure the applications they design and deploy. Because the risks to applications are always evolving, The OWASP Top 10 list is revised each time to reflect these changes, along with the techniques and best practices for avoiding and remediating the vulnerabilities. In addition to the OWASP Top 10 for web applications, OWASP has also created similar lists for Internet of Things vulnerabilities, as well as mobile security issues. The list is compiled by evaluating the overall threat as well as the regularity of the threats faced. Some risks may be rare but when exploited could be fatal, while others are common but easy to guard against. Here's a quick overview of the list.

The OWASP Top 10 Vulnerabilities

1. **SQL Injection Attacks**

SQL Injections are at the head of the OWASP Top 10, and occur when a database or other areas of the web app where inputs aren't properly santized, allowing malicious or untrusted data into the system to cause harm. SQL injection attacks are simply when data is sent to any form of code interpreter that can be run as a command or in the case of a database – a query. The idea is that the data fools the interpreter into either handing over data that the attacker wants or it executes commands that may be hostile in the environment.

More about SQLi:

Knowledge Base: SQL Injection

SQL Injection Tutorial

1. **Broken Authentication & Session Management**

Broken Authentication and Session Management vulnerabilities allow anonymous attacks aimed at attempting to steal valuable data, especially Personally Identifiable Information. If authentication or session management protocols have not been implemented properly, they may enable a hostile to steal passwords, session keys or tokens or otherwise assume or exploit a user's identity.

1. **Cross-Site Scripting (XSS) Attacks**

Cross-Site Scripting, often shortened as XSS, attempts to trick a browser into accepting data that isn't from a trusted source. Applications that allow user input but don't have control over output are highly vulnerable to XSS. If successful, XSS allows the attacker to take over a user session, cause damage to a website or force the user to visit another site (often a website hosting further hostile code). There are three different kinds of XSS attacks, referred to as another site (often a website hosting further hostile code). There are three different kinds of XSS attacks, referred to as Stored XSS, DOM Based XSS, and Reflected XSS.

More about XSS:

Knowledge Base: Cross-Site Scripting

The Definitive Guide to Cross-Site Scripting Prevention

1. **Insecure Direct Object References**Insecure Direct Object References occur when authentication isn't properly executed. If an application is vulnerable, malicious users may be able to gain administrative access to the application. If no access control check or other protection is in place, an attacker could manipulate that type of reference to access data they're not authorized for.
2. **Security Misconfiguration**When security processes and practices aren't correctly followed or implemented, Security Misconfigurations can easily be used by attackers to detect weak areas that would allow them to access privileged data. Configuration of the whole application environment including servers, platforms, etc. needs to be properly defined, implemented and controlled or it can lead to security holes.
3. **Sensitive Data Exposure**When security controls like SSL and HTTPS are not properly implemented, data can be leaked or stolen through a Sensitive Data Exposure vulnerability. Sensitive data such as Personally Identifiable Information, including financial and banking details, tax IDs, and passwords can be at risk if not correctly secured. Applications should ensure that they authenticate access, encrypt data and ensure the integrity of data in the transport layer. A failure to do so may allow for weak (and exploitable) algorithms and might allow access from expired or forged certificates, leading to a privacy violation.
4. **Missing Function Level Access Control** This risk is posed when web applications don't correctly verify function level access rights before making available functionality that shouldn't be granted.
5. **Cross Site Request Forgery Attacks (CSRF)**Cross-Site Request Forgery attacks, often shortened to CSRF, allow the attacker to forge an HTTP request from the victim, and may include data such as cookies or authentication information. The victim's browser may then be used to generate additional requests that appear legitimate to the object of the attack.*More about CSRF:*

   Knowledge Base: Cross-Site Request Forgery

   The Ultimate Guide to Understanding & Preventing CSRF
6. **Using Components with Known Vulnerabilities** Components, especially libraries and frameworks derived from the open source community, should never be used when there are known vulnerabilities in the code. Doing so undermines the application and possibly the entire organization, as an attacker could easily leverage an SQL injection, XSS attack or similar to attempt an application takeover.
7. **Unvalidated Redirects and Forwards**Unvalidated Redirects and Forwards can be used with a bit of social engineering to mimic an already existing site and trick visitors into downloading malware or giving up Personally Identifiable Information.

It is important to note that the OWASP Top 10 isn't a complete list of vulnerabilities, but rather a starting place from which security experts and developers together can build off of.

Further reading:

- Visit CheckMarx' Vulnerability Knowledge Base
- Read about OWASP Mobile Top Ten: Avoiding The Most Common Mobile Vulnerabilities
- Download FREE WhitePaper "OWASP Top 10 for IoT Explained"

SIGN UP FOR A FREE DEMO

R E Q U E S T A D E M O