

OWASP

Owasp stands for Open Web Application Security Project. It is a non-profit organization that develops free open-source materials for enhancing and research in the field of web security. Their most recent project is OWASP Top 10 which outlines the 10 most prevalent web security issues. The OWASP resources can be used by individuals and companies to secure their applications against cyber attacks. Using the OWASP 10 helps in identifying the risks, prioritizing them, and defining a strategy around them.

The Top 10 list is created using four factors:

1. What is the frequency of the threat?
2. How exploitable is the threat?
3. How easily can the threat be detected?
4. Will the threat impact the business and technology of the company?

The Top 10 security risks include:

1. Injection

An injection is when an intended command is input into a form field and the field isn't secured. This may result in the command getting executed and outputting sensitive data. This can be prevented by implementing validation and data sanity checks at the fields in the backend. Access to sensitive data can only be set for the admin and not for a generic user. This will help in averting data exposure. Popular example is **SQL Injection**.

2. Broken Authentication

Lack of proper authentication can result in hackers accessing user account data and comprising the whole system. User credentials can be leaked. Attackers can even use brute force to predict the passwords to barge into the systems. This can be prevented using two-factor authentication, limiting login attempts, and limiting logged-in devices or applications simultaneously.

3. Sensitive Data Exposure

Applications should encrypt sensitive data like passwords, health records, and credit card information, and store them as in case of a breach, it should almost be impossible for the attacker to decrypt them. Web browsers can turn off caching of any critical data.

4. XML External Entities (XEE)

A web application can be compromised if an improper XML parser is implemented. An attacker can force an input referencing an external entity and try to exploit the vulnerability in the parser. To avoid this JSON type of data is

used in web applications for smoothly passing data between pages of the application.

5. Broken Access Control

Broken Access Control is when attackers exploit the weak authorization of the application and bypass it by passing some kind of information in the URL to pretend to be an authorized user. The use of authorization tokens helps in saving from such attacks and identifying the correct user.

6. Security Misconfiguration

One of the most common vulnerabilities often occurs due to improper configurations or configurations not being updated with modern protocols. Excessive information is also revealed in the errors. Unused code should be removed from the application to avoid unexpected execution.

7. Cross-Site Scripting

A hacker could trick an application into supplying another user with a malicious script that might be executed in the user's browser using a funny popup and can result in the account getting taken over or information being stolen. For example, an attacker could send an email to a victim that appears to be from a trusted bank, with a link to that bank's website. This link could have some malicious JavaScript code tagged onto the end of the URL. If the bank's site is not properly protected against cross-site scripting, then that malicious code will be run in the victim's web browser when they click on the link. Mitigation strategies for cross-site scripting include escaping untrusted HTTP requests as well as validating and/or sanitizing user-generated content. Using modern web development frameworks like ReactJS and Ruby on Rails also provides some built-in cross-site scripting protection.

8. Insecure Deserialization

Serializing data means converting objects from code to a specific format for further use and deserializing means converting it back to code-usable objects. Insecure Deserialization happens when deserialization happens from an untrusted source. This can lead to DDoS attacks and remote code execution attacks. The most important way to prevent this is to block deserializing data from untrusted web services.

9. Using Components With Known Vulnerabilities

Applications should avoid open-source component libraries that have known vulnerabilities as it may lead to those vulnerabilities being embedded in the code and compromising the security of the application. Libraries should be thoroughly scanned before being used and should be updated.

10. Insufficient Logging And Monitoring

Logging and monitoring reduce the time between the data breach and the action being taken to rectify the issues. This can also help the application become more aware of the usage and strategize a response plan in case an attack happens.

How to avoid the worst cybersecurity breaches and test the software that is cyber-secure

- Thorough testing of applications can avoid security breaches and make the application more robust against attacks.
- The following resources are provided by the OWASP community:
 - [OWASP ZAP](#)
 - [Burp Proxy](#)
 - [Webstretch Proxy](#)
 - [Firefox HTTP Header Live](#)
 - [Firefox Tamper Data](#)
 - [Firefox Web Developer Tools](#)
 - [Grendel-Scan](#)
 - [W3af](#)
 - [Skipfish](#)
 - [Web Developer toolbar](#) (Chrome extension)
 - [HTTP Request Maker](#) (Chrome extension)
 - [Cookie Editor](#) (Chrome extension)
 - [Cookie swap](#) (Chrome extension)
 - [Session Manager](#) (Chrome extension)
 - [Subgraph Vega](#)

How to Hack a Website?

A website can be hacked using two ways:


- Online SQL Injection
Add "inurl:.php?id=" in the website URL to check its vulnerability. If the response comes out to be "you have an error in your SQL syntax", the site is likely to be hacked. Further, more complicated steps are given in the blog shared in the references.
- HTML Coding
Enter the wrong credentials to get an error page pop-up. Right-click on the error page to view the source. If the Javascript code is accessible, remove the portion

of the code that authenticates a user. This way the authentication can be bypassed and the website can be hacked.

Libraries and tools are available or recommended that are relevant to the JVM platform

- <https://owasp.org/www-project-java-encoder/>
- <https://owasp.org/www-project-enterprise-security-api/>
- <https://owasp.org/www-project-java-html-sanitizer/>
- <https://owasp.org/www-project-dependency-check/>

References:

1. <https://www.cloudflare.com/learning/security/threats/owasp-top-10/>
2. https://owasp.org/www-project-web-security-testing-guide/v41/6-Appendix/A-Testing_Tools_Resource
3.  What is the OWASP Top 10? | AppSec 101
4. <https://cwatch.comodo.com/blog/website-security/hack-this-site-in-7-steps/>