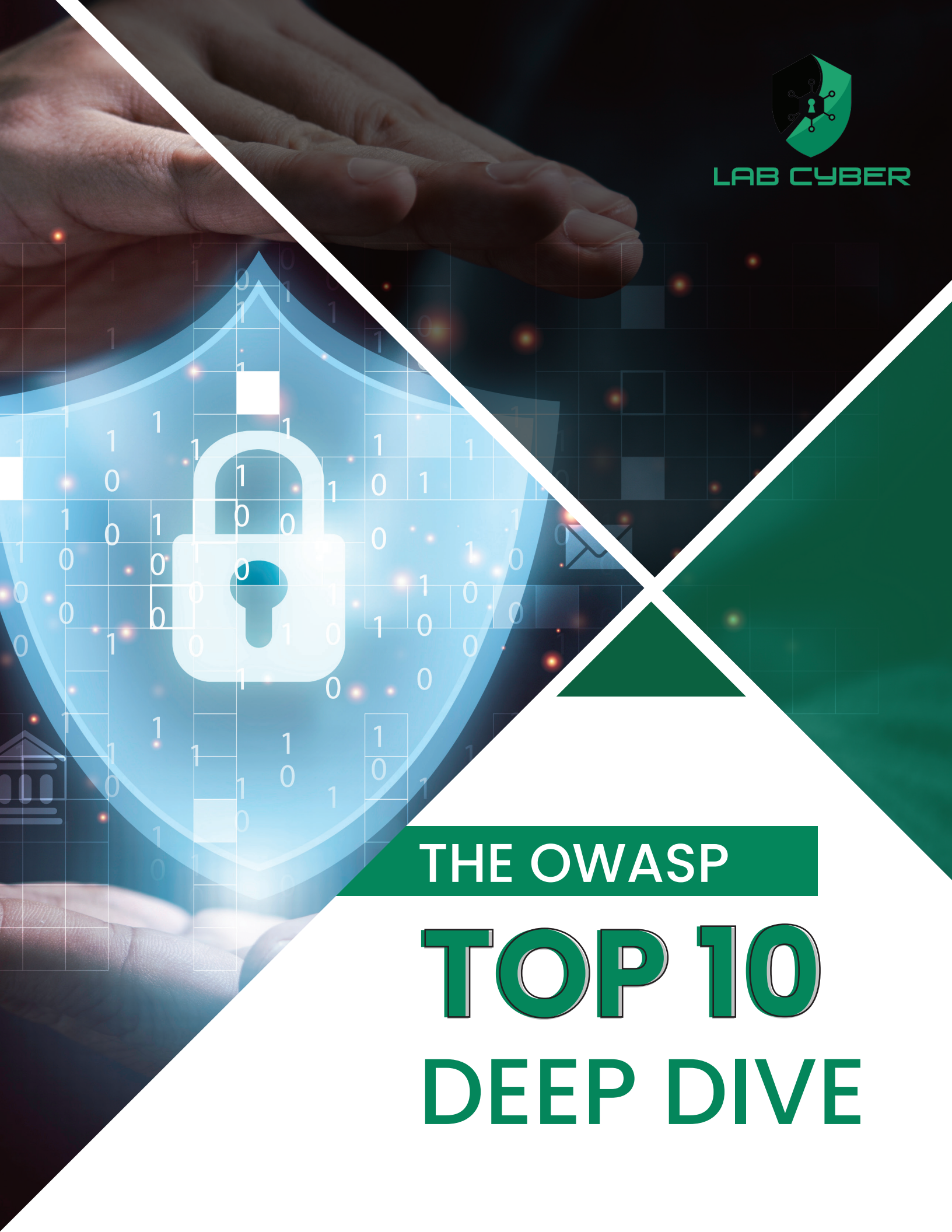


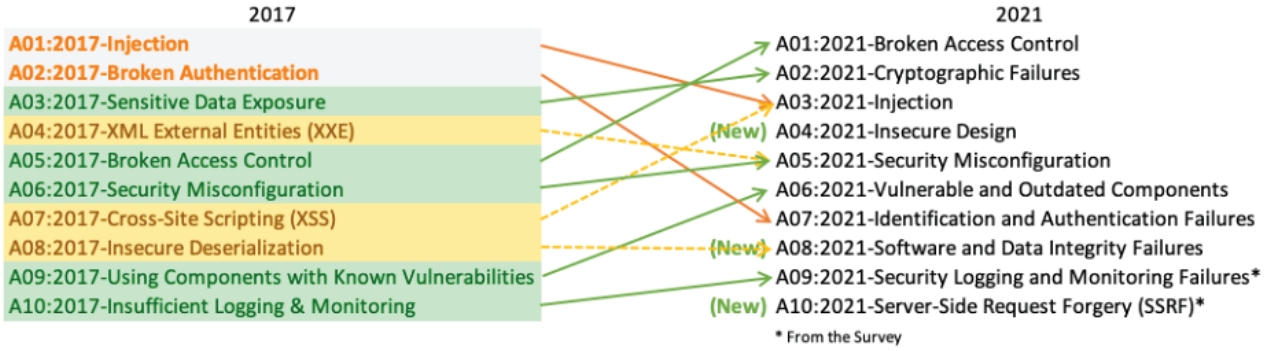
LAB CYBER

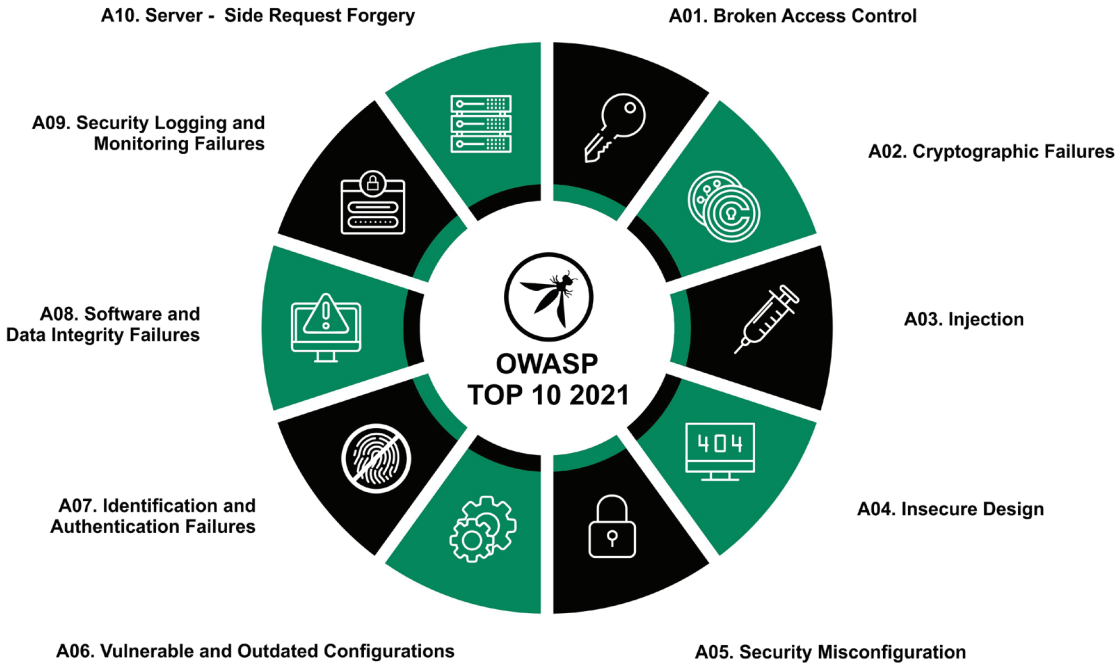
THE OWASP TOP 10 DEEP DIVE

Introduction

The OWASP Top 10 is a standard awareness document for developers and web application security. It represents a broad consensus about the most critical security risks to web applications.

OWASP stands for Open Web Application Security Project and they work to improve the security of software.





The list is updated every three to four years

OWASP #1 - BROKEN ACCESS CONTROL

Access control refers to a process for ensuring only authorized users have access to certain types of dataA process for ensuring only authorized users have access to certain types of data.

Broken access control typically leads to unauthorized information disclosure, modification, or destruction of all data or performing a business function outside the user's limits.

Elevation of privilege. Acting as a user without being logged in or acting as an admin when logged in as a user.

Horizontal privilege - The user gets access to different resources other than the one intended for them

Broken access control can also lead to the unauthorized access of sensitive links and web pages/ files on a website due to the fallacy of security through obscurity (the belief that anything that exists on a website that isn't linked or indexed cannot be found)

PREVENTION

- Deny by default i.e every user starts with the minimum privileged functions
- Enable RBAC (Role-Based Access Control)
- Disable web server directory listing and ensure file metadata and backup files are not present within web roots
- Constant Testing & Auditing of Access Controls

OWASP #2 - CRYPTOGRAPHIC FAILURES

Focuses on failures related to cryptography which often lead to exposure of sensitive data.

Attack Scenario

A site doesn't use or enforce TLS for all pages or supports weak encryption. An attacker monitors network traffic and steals the user's session cookie. The attacker then replays this cookie and hijacks the user's (authenticated) session, accessing or modifying the user's private data.

Transport Layer Security - A cryptographic protocol that ensures client-server communications on a network are secure.

HTTPS makes use of the TLS to ensure sensitive data like passwords and credit card details are encrypted

Prevention - Ensure website traffic is encrypted by making use of services like Let's Encrypt

OWASP #3 - INJECTION

SQL Injection Attacks

These occur when maliciously crafted inputs are submitted by an attacker causing an application to perform an unintended action.

SQL Injection Attack Methodology

- Extract sensitive information
- Enumerate login details of registered users on the website
- Deletetables of data
- Inject even more malicious code

SQL Injection Attacks can be prevented by:

- Parameterized statements (making sure the inputs passed into the statements are treated in a safe manner)
- Escaping inputs
- Sanitizing inputs (ability of the server to reject inputs that look suspicious)

Cross-Site Scripting Attack

Primarily a client side attack and targets the victim's browser with the use of malicious code

Very common on message boards, forums and webpages that allow comments

XSS attacks are possible with CSS, Flash but most common in JavaScript

XSS attacks are also considered to be less dangerous than SQL injection attacks

XSS Attacks can:

- Hijack the victim's cookies
- Gain access to the webcam, microphone and even the geolocation of the victim
- Combined with social engineering tactics, install Trojans, keyloggers etc

Method of Attack

- The attacker must find a way to inject the malicious code into the web page that the victim visits
- The web page needs to directly include user input
- XSS attacks can be prevented by validating user input

OWASP #4 - Insecure Design

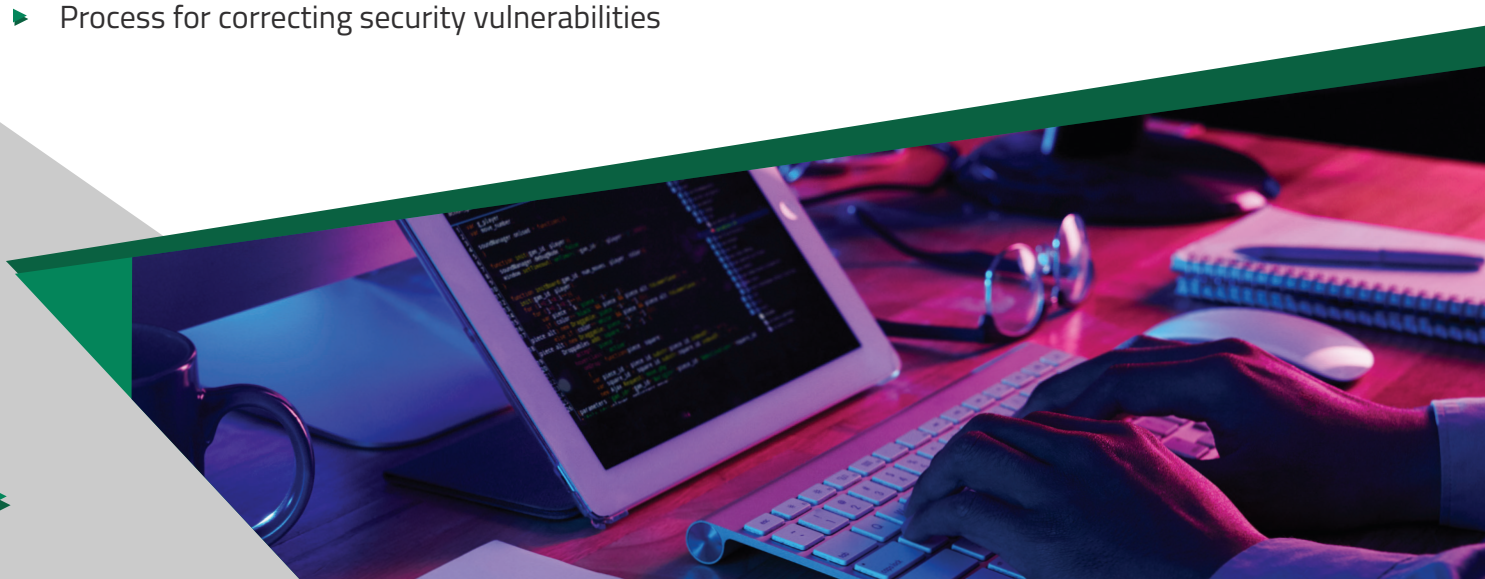
A new category for 2021 that focuses on risks related to design and architectural flaws with a call for more use of threat modeling and reference architectures.

Attack Scenario

A cinema chain allows group booking discounts and has a maximum of fifteen attendees before requiring a deposit. Attackers could threat model this flow and test if they could book six hundred seats and all cinemas at once in a few requests, causing a massive loss of income.

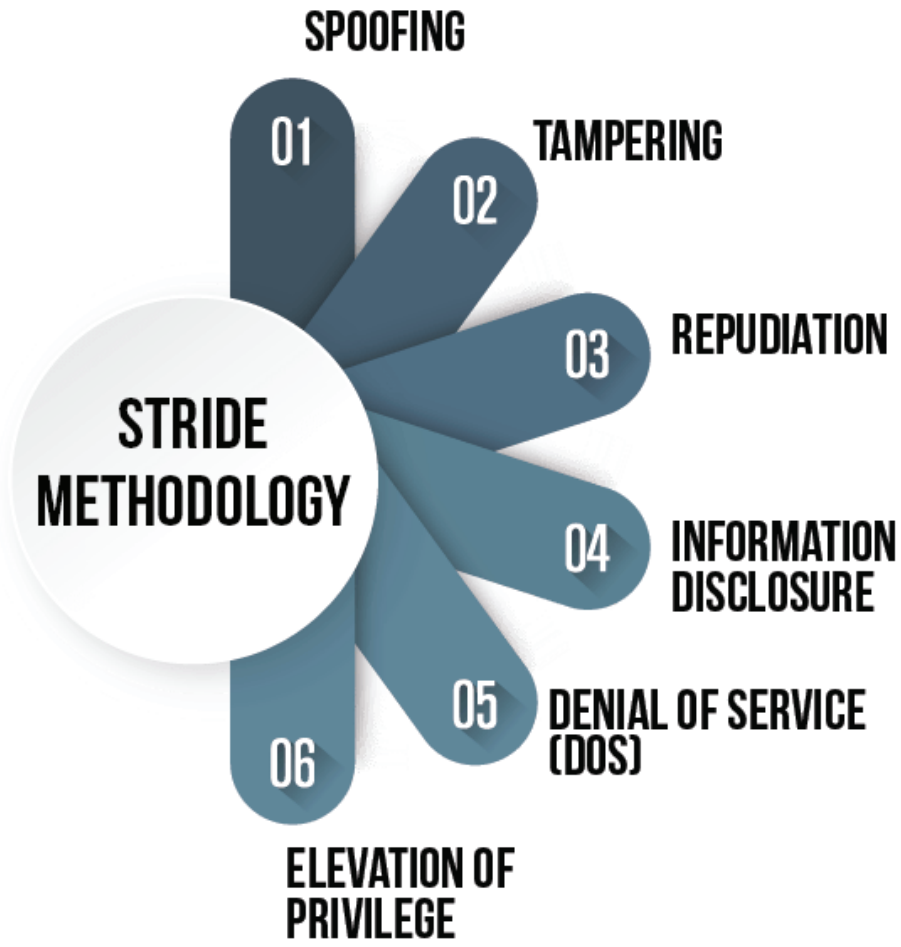
Building Secure Software

- Understand potential threats
- Identify where malicious scripts can be inserted
- Anticipate failure conditions
- Process for correcting security vulnerabilities



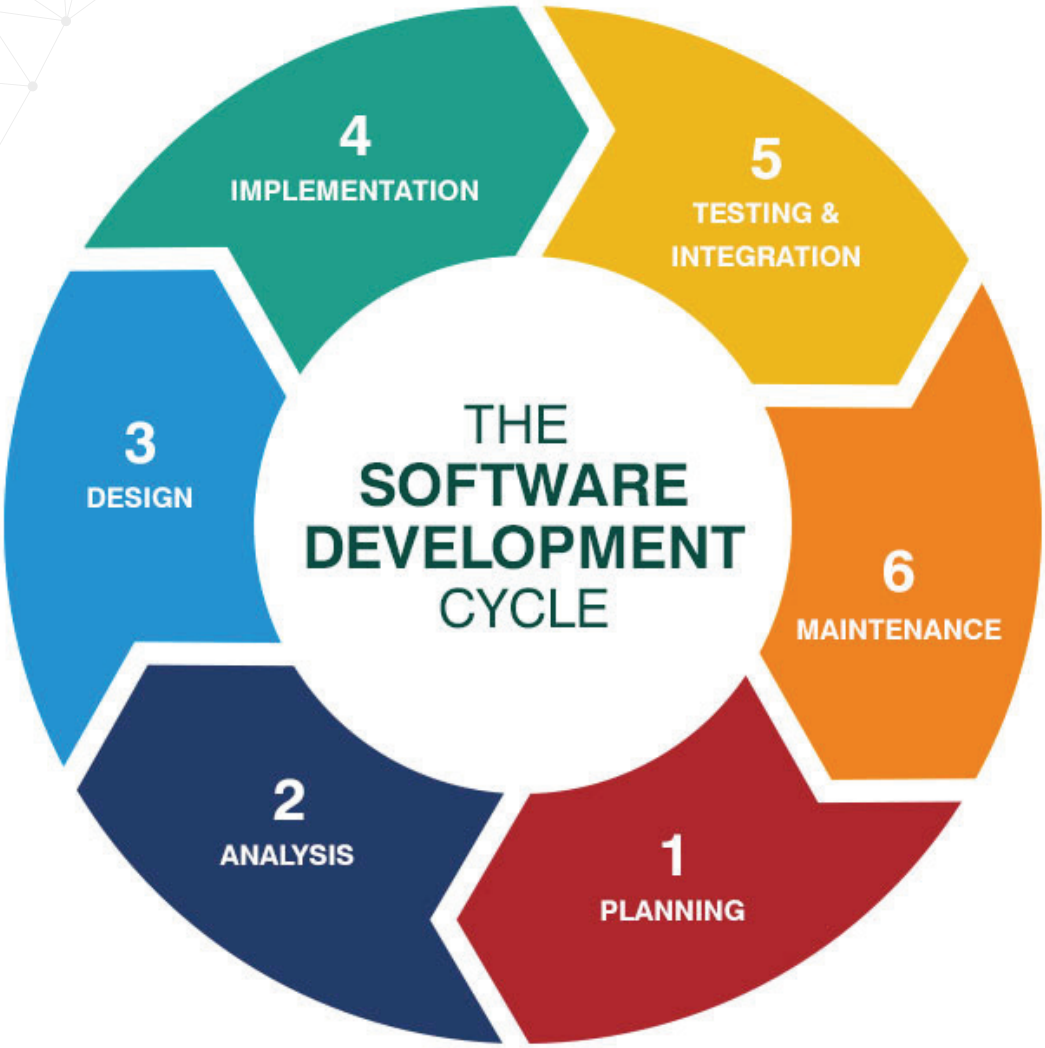
Threat Modeling

The process where potential threats and vulnerabilities or the absence of safe guards are identified, evaluated and mitigations can be prioritized



Data Classification

- Public - Accessible to anyone
- Private - Accessible only to authenticated users
- Restricted - Sensitive data owned by specific users such as account or profile information
- High Risk - Access tokens, passwords and other credentials that must never be shared



PREVENTION

- Principle of Least Privilege - Each program or user should operate only with the least amount of privilege required to achieve their goals.
- Validation of Input - Every input from the user is validated to ensure it is in the expected format and reject it otherwise
- Segregation of Tenants - Different environments such as live and test should be on separate networks and not share resources
- Data must be encrypted at all times including during the resting phase
- Fail Securely - Internal architectural details should be not be revealed in error messages
- Running code should issue logs that reveal data such as type and volume of traffic as well as available bandwidth being used.

OWASP #5 - SECURITY MISCONFIGURATION

Attack Scenario

The application server comes with sample applications not removed from the production server. These sample applications have known security flaws attackers use to compromise the server. Suppose one of these applications is the admin console, and default accounts weren't changed. In that case, the attacker logs in with default passwords and takes over.

Databases and applications typically come with default accounts making it easy for developers to get started quickly but must be disabled in the production phase.

PREVENTION

- Client-side error reporting should be turned off.
- Use of HTTPS should be enforced.
- All development tools like interactive consoles or debugging tools should be disabled.
- When working with a team, access to production data should be restricted to internal networks or require use of two factor authentication.

OWASP #6 - VULNERABLE & OUTDATED COMPONENTS

Most modern software are developed using pre-built code in libraries and frameworks written by other developers.

Such pre-built code might have vulnerabilities or even worse may contain code written with malicious intent.

Log4j - This vulnerability allowed attackers to perform remote code execution by exploiting insecure JNDI (Java Naming & Directory Interface) lookups in the library.

"Heartbleed" - Back in 2014, this vulnerability in OpenSSL cryptographic software allowed attackers to read large chunks of memory on the server.

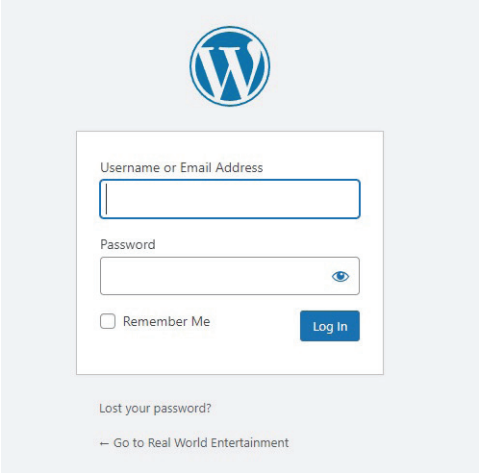
Solar Winds Attack - Attackers planted malware on the Orion software used by thousands of companies.

PREVENTION

- All unused dependencies, unnecessary features, components and files should be removed.
- Only obtain components from official sources over secure links preferably those with signed packages.
- Use dedicated tools to scan your dependency tree for security risks e.g Github security alerts and ShiftLeft
- Monitor security bulletins
- Perform regular code reviews & pen testing.

OWASP #7 -IDENTIFICATION & AUTHENTICATION FAILURES

Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords or session tokens and to exploit other implementation flaws to assume other users' identities temporarily or permanently.



Attack Scenario

An attacker harvests a list of usernames for a website and then uses brute force attacks to try and guess the password for each username.

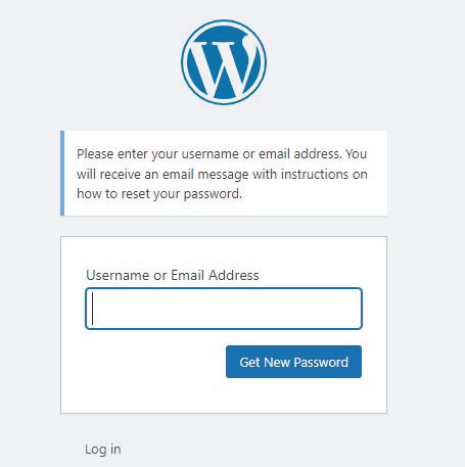
Authentication Weaknesses can be caused by:

- Permits default or weak passwords
- Permits brute force attacks
- Uses weak or ineffective credential recovery and forgot-password processes, such as "knowledge-based answers"
- Uses plain text, encrypted, or weakly hashed passwords data stores
- Has missing or ineffective multi-factor authentication.

Brute Force Attacks

An attacker could either harvest usernames or try logging in by using different types of usernames to see if the web server can confirm the existence of such usernames.

** Also "password-reset" pages could be exploited to reveal the existence of registered email addresses."



PREVENTION

Brute Force Attacks

- Enforce the use of strong passwords
- Time outs after 5 failed login attempts

Login & Password-Reset Pages

Generic messages should be displayed after each attempt to login or reset passwords.

"Either the username or password is incorrect"

"If the email address bob@gmail.com exists on this website, a password reset link will be sent to it"

Use two or multi factor authentication and do not ship or deploy with default credentials

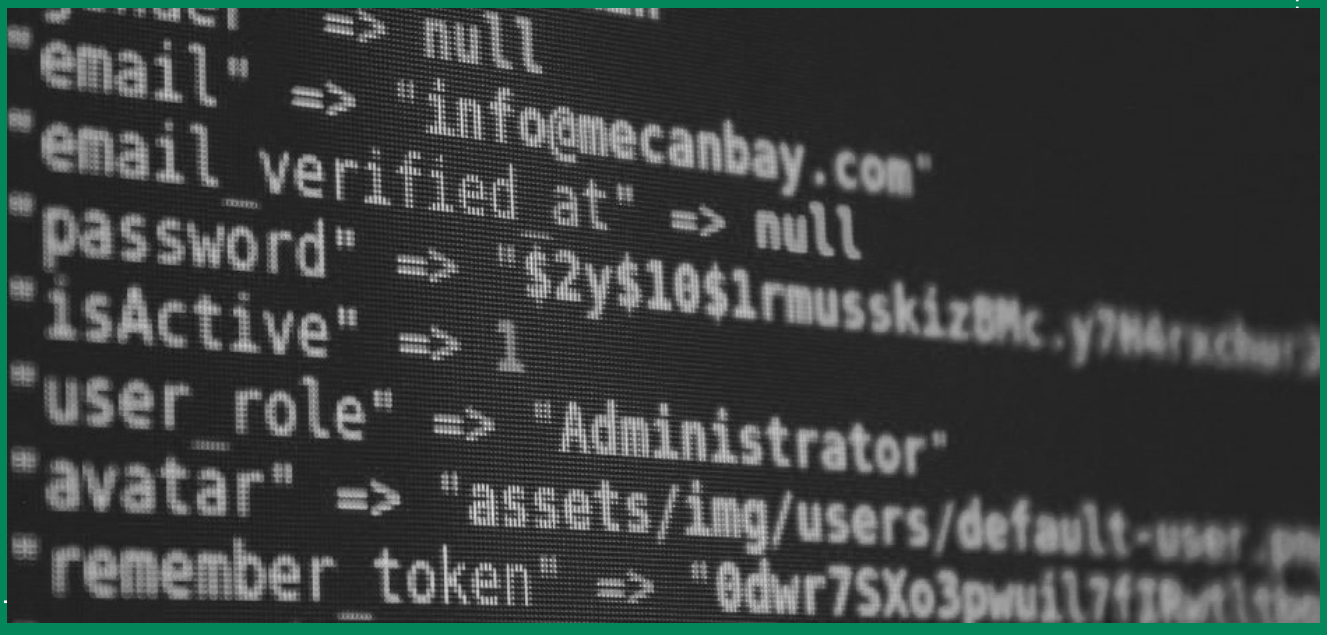
OWASP #8 - SOFTWARE & DATA INTEGRITY FAILURES

A new category for 2021 focuses on making assumptions related to software updates, critical data, and CI/CD pipelines without verifying integrity

Many applications now include auto-update functionality, where updates are downloaded without sufficient integrity verification and applied to the previously trusted application.

PREVENTION

- Use digital signatures to verify software or data is from the expected source and has not been altered.
- Software supply chain security tool such as OWASP Dependency Check should be used to verify that components do not contain vulnerabilities
- Ensure that there is a review process for code and configuration changes to minimize the chance that malicious code can be introduced into the software pipeline
- Ensure that unencrypted data is not sent to untrusted clients without some form of integrity check to prevent the risk of them tampering with the data.



OWASP #9 - SECURITY LOGGING & MONITORING FAILURES

This category is to help detect, escalate and respond to active breaches

Attack Scenario

A major Indian airline had a data breach involving more than ten years' worth of personal data of millions of passengers, including passport and credit card data. The data breach occurred at a third-party cloud hosting provider, who notified the airline of the breach after some time.

Insufficient Monitoring can be caused by:

- Auditable events like logins and transactions are not logged
- Warnings and errors generate no or unclear log messages
- Response escalation processes are not in place or effective
- Application cannot detect, escalate or alert for active attacks in real-time

PREVENTION

- Ensure all login, access control, and server-side input validation failures can be logged with sufficient user context to identify suspicious or malicious accounts
- All old logs should be kept for an extended period for delayed forensic analysis and investigations
- All high-value transactions must have an audit trail with integrity controls to prevent tampering or deletion
- Effective incident response, response escalation and recovery plans must be established

OWASP #10 - SERVER SIDE REQUEST FORGERY (SSRF)

Server-Side Request Forgery (SSRF) flaws occur whenever a web application fetches a remote resource without validating the user-supplied URL
Web applications can trigger requests between HTTP servers to fetch remote resources such as software updates or to import metadata.

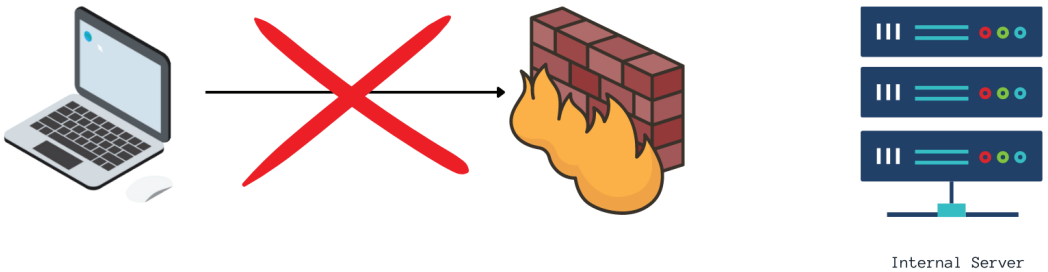
- Used to gain access to sensitive internal data
- Can be used to launch DDoS attacks

SSRF exploits could also be used to launch DDoS attacks against a third party website by using the vulnerable server.

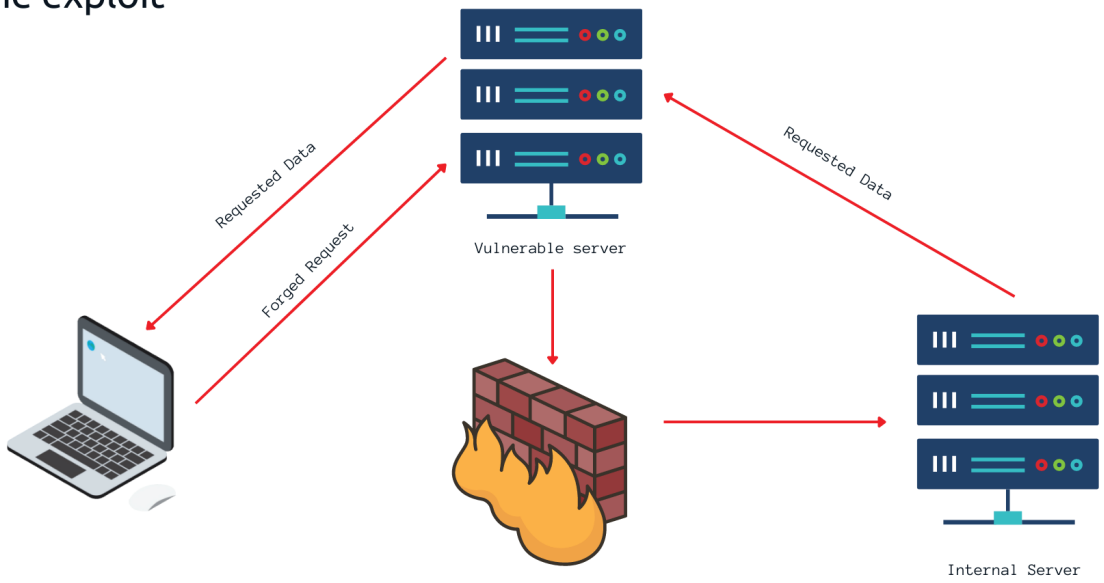
By spamming the vulnerable server with requests to fetch metadata from a third party website, the attacker can overwhelm the third party website while hiding behind the vulnerable web server.

In a typical SSRF attack, the attacker might cause the server to make a connection to internal-only services within the company infrastructure. In other cases, they may be able to force the server to connect to arbitrary external systems, potentially leaking sensitive data such as authorization credentials.

What happens if an attacker sends direct requests to access an internal server?



The exploit



PREVENTION

In the network layer:

- Enforce “deny by default” firewall policies or network access control rules to block all but essential intranet traffic.
- Segment remote resource access functionality in separate networks to reduce the impact of SSRF

In the application layer:

- Sanitize & validate all user input data
- Disable HTTP redirections
- Disable raw responses to clients

Only Make Outgoing HTTP Calls On Behalf of Real Users and limit the number of links a user can share in a given time frame