proficom
we make IT work

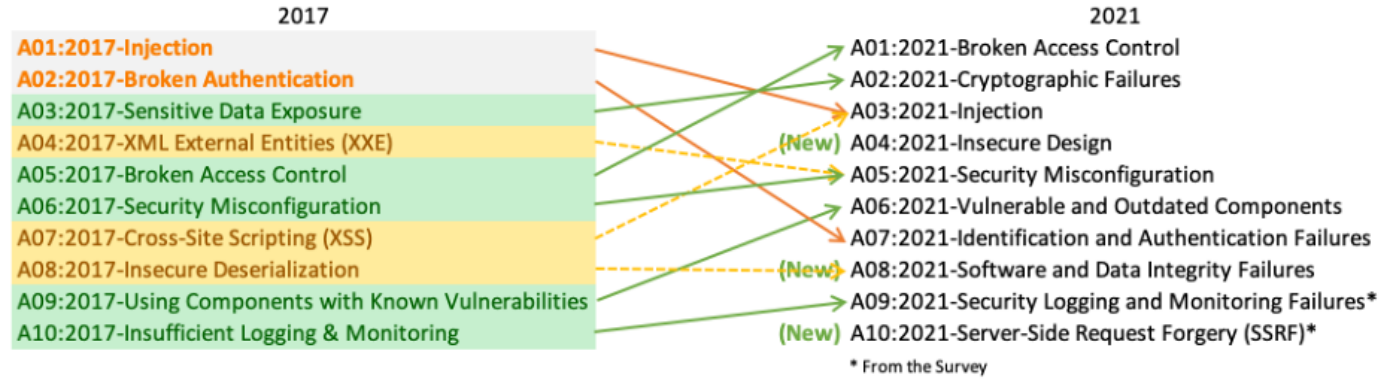
OWASP® TOP10
und seine Änderungen 2021

Die Anzahl der Angriffe auf IT-Systeme steigt stetig an. Entwickler:innen müssen ihre Software vor Cyberattacken schützen. Die OWASP TOP10 fasst die wichtigsten Bedrohungen zusammen und bietet einen der effektivsten ersten Schritte, um Softwareentwicklung und Secure Code miteinander zu vereinen.

Das Open Web Application Security Project (OWASP) ist eine offene, nicht profitable Community und sammelt und evaluiert Schwachstellen in Webapplikationen und APIs in ihrem TOP10 Projekt. Ziel des Projekts ist es, Organisationen und Projekte zu ermöglichen, sichere und vertrauenswürdige Webapplikationen und APIs zu entwickeln und einzusetzen. Die gefundenen Schwachstellen werden alle vier Jahre veröffentlicht, die neueste Version ist 2021 erschienen. Weitere Informationen finden Sie unter: https://owasp.org/Top10.

Unterschiede zu 2017

Wie im unteren Bild zu sehen ist, hat sich die Reihenfolge geändert und drei Punkte sind neu, beziehungsweise vorherige Punkte zusammengefasst worden.



Quelle: https://owasp.org/www-project-top-ten

Liste der Top10 Schwachstellen

1. Broken Access Control

Sind Zugangsberechtigungen für Systeme falsch konfiguriert oder zu großzügig gesetzt, kann es zu unbefugten Zugriffen auf das System bzw. Teile davon kommen. Dabei können Daten ungewollt durch Nutzer:innen eingesehen, verändert oder sogar zerstört werden, bspw. durch Manipulation einer URL, einer API-Anfrage oder eines Tokens. Um dieses Problem zu vermeiden, gilt das Prinzip der minimalen Rechtevergabe.

2. Crypthographic Failures

Passwörter, Kreditkarteninformationen, Geschäftsgeheimnisse und weitere sensible Informationen gehören zu besonders schützenswerten Daten und brauchen eine adäquate Verschlüsselung. Ist diese Verschlüsselung unzureichend oder nicht vorhanden, können Daten in die falschen Hände gelangen und bspw. bei einer Datenübertragung durch Dritte problemlos gelesen werden. Es ist unvermeidbar, dass angemessene kryptografische Maßnahmen ergriffen und regelmäßig evaluiert werden, da sich auch kryptografische Methoden weiterentwickeln.

3. Injection

Die Injection umfasst eine ganze Gruppe an Schwachstellen (z.B. SQL, OS command, LDAP injection), die durch das Verarbeiten von Informationen durch einen Interpreter entstehen können. Werden Schnittstellen nicht abgesichert, können z.B. von Nutzer:innen eingegebene Kommandos durch den Service als solche interpretiert und ausgeführt werden. So kann es u.a. zum ungberechtigten Zugriff auf Datenbankinhalte und Funktionen kommen, die von außen nicht erreichbar sein sollten. Jegliche Eingaben und API-Anfragen sollten stets zuerst validiert, gefiltert bzw. bereinigt werden, um Injections zu verhindern.

4. Insecure Design
Durch Mängel im Design einer Software, d.h. durch fehlende oder ineffektive Kontrollen, entstehen Schwachstellen, die leicht ausnutzbar sind. Bei einer Passwortwiederherstellung bspw., die auf Sicherheitsfragen und -antworten basiert, besteht die Gefahr, dass Antworten auch Dritten bekannt oder durch diese leicht ermittelbar sind. Es ist essenziell, dass bereits beim Softwaredesign Sicherheit eine gebührende Rolle zugesprochen wird und schon früh ein Austausch mit Sicherheitsexpert:innen stattfindet.

5. Security Misconfiguration
Fehlkonfigurationen von Sicherheitseinstellungen sind häufig vertreten und äußerst problematisch. Beispiele hierfür sind nicht geänderte Standardpasswörter und unnötig informative Fehlermeldungen. Auch angebundene Features und Systeme, die nicht (mehr) benötigt werden, bieten eine zusätzlich Angriffsfläche. Regelmäßige Reviews und Aktualisierungen der Sicherheitskonfiguration helfen, dieses Einfallstor klein zu halten.

6. Vulnerable and Outdated Components
Selten kommt eine Software ganz ohne externe Softwarebibliotheken oder Drittsysteme aus. Handelt es sich dabei um veraltete oder sicherheitsschwache Komponenten, wirkt sich das auch auf die Sicherheit des Systems aus, das diese Komponenten integriert. Eine regelmäßige Überprüfung dieser Komponenten auf Aktualität und derzeit bekannte Sicherheitslücken ist dringend zu empfehlen. Auch bei der Aktualisierung auf neuere Versionen lohnt sich eine Analyse, um zu verhindern, dass neue Schwachstellen aufgeworfen werden.

7. Identification and Authentication Failures
Ist die Benutzer:innen-Identifikation und -Authentifizierung mangelhaft oder gar fehlerhaft, entsteht ein Einfallstor für unbefugtes Eindringen. Ausnutzbar sind hier beispielsweise schwache Passwörter, die mit der Brute-Force-Methode leicht zu knacken sind. Unsichere Cookie- und Session-Identifikatoren, die beispielsweise für mehrere Sessions wiederverwendet werden, oder eine schlecht umgesetzte Zwei-Faktor-Authentifizierung können Angreifer:innen ebenfalls Zutritt gewähren. Hilfreich können dagegen unter anderem Sicherheitsvorgaben für die Passworterstellung, eine starke Multi-Faktoren-Authentifizierung und effektives Session-Management sein.

8. Software and Data Integrity Failures
Fehler im Bereich der Software- und Datenintegrität können unter anderem entstehen, indem Daten durch unsichere Infrastruktur, d.h. Zuliefer- und Umsysteme, manipuliert werden. Beispielsweise eine nicht sichere CI/CD-Pipeline kann verschiedene anfällige Endpunkte öffnen. Auch die Installation unsignierter Software und unzureichende Verschlüsselung kann gefährlich sein. Um diese Schwachstellen zu minimieren, muss eine sichere Infrastruktur und Datenübertragung gewährleistet sein. Das ist z.B. durch Verwendung digitaler Signaturen, regelmäßige Überprüfung der Sicherheitskonfigurationen von Umsystemen und Vorgaben hinsichtlich Verschlüsselung und Zugangskontrolle möglich.

9. Security Logging and Monitoring Failures
Je schlechter ein System überwacht wird, desto schwerer wird es, vorhandene Sicherheitslücken aufzudecken und verdächtige Aktivitäten zu erkennen. Problematisch sind hier bspw. lückenhaftes Logging und fehlende Scan- und Alarmmechanismen, wodurch Angriffe zu spät oder gar nicht bemerkt werden. Durch die Protokollierung wichtiger Ereignisse (z.B. Logins und bestimmte Transaktionen), ordentliche Verschlüsselung und Ablage von Logs sowie adäquate Alarmmechanismen kann im Ernstfall schnell reagiert werden.

10. Server Side Request Forgery (SSRF)
Bei der SSRF entsteht die Schwachstelle, wenn eine Anwendung sich von einer anderen Stelle Informationen holt, ohne diese zu validieren. Ein Angreifer kann die URL so verändern, dass die eigentlichen Schutzmaßnahmen wie VPN und Firewall nicht greifen.