Merkblatt 01

Generelle Sicherheitsanforderungen an Web Applikationen

1. Sicherheits Grundsätze

Folgende Sicherheitsgrundsätze sollen durch die Einhaltung der Anforderungen erreicht werden:

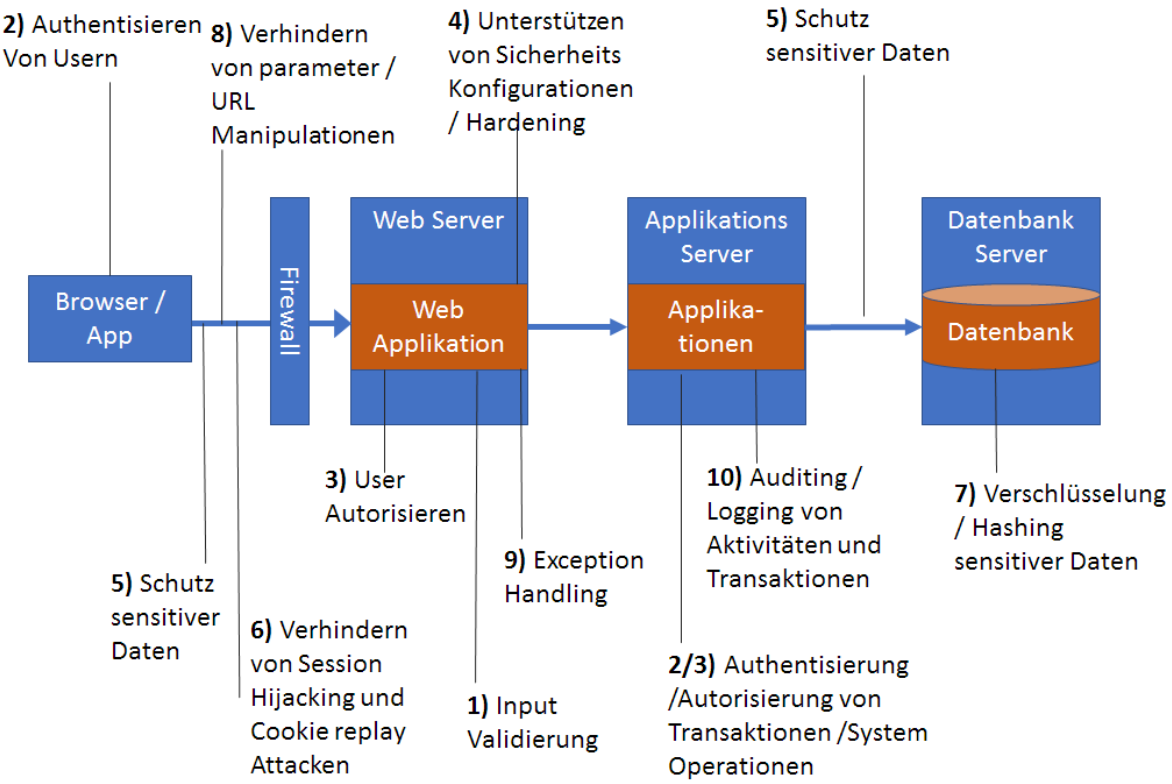
- Erstellung von Eingabe Validierungsstrategien in Web Applikationen
  Partitionierung von Websites in offene und eingeschränkte Bereiche

- Implementierung effektiver Account Management-Praktiken
- Entwicklung effektiver Authentifizierungs- und Berechtigungsstrategien
- Schutz sensibler Daten
- Schutz von Benutzersitzungen
- Vermeidung von Parametermanipulationen
- Sicherer Umgang mit Ausnahmen resp. Fehlermeldungen an Systemen / Applikationen
- Sichern der Konfigurationsverwaltungsfunktionen einer Anwendung / Server's
- Aufzählung von Audit- und Logging-Überlegungen

2. Sicherheitsarchitektur

Überblick

Die nachfolgende Grafik zeigt auf, wo die geforderten Sicherheitsanforderungen greifen.



Beschreibung

1) Eingabe Validierung	Attacken durch Einbettung von bösartigen Strings in Abfragen, Web Form Felder, Cookies, URL und HTTP headers. Diese beinhalten auch die Ausführung von cross-site scripting (XSS), SQL injection und buffer overflow Attacken.
2) Authentisierung	Nachahmen von bestehenden Identitäten, password cracking, unberechtigtes Erlangen von höheren Systemrechten sowie Unautorisierter Zugriff auf Systeme.

3) Autorisierung	Zugriff auf vertrauliche Informationen oder über Zugriffsrechte eingeschränkter Zugriff auf Daten, Manipulation von Daten sowie Ausführung von unautorisierten Transaktionen oder Systemoperationen
4) Konfigurations Management	Unbefugter Zugriff auf Administrationsschnittstellen, Möglichkeit Konfigurationsdaten zu aktualisieren und unberechtigten Zugriff auf Benutzerkonten und Kontoprofile zu erlangen.
5) Sensitive Data	Offenlegung vertraulicher Informationen sowie Datenmanipulationen.
6) Session Management	Schwache oder keine Session Identifier führen zu session hijacking und Identitäts Nachahmung.
7) Verschlüsselung	Einschränken des Zugriffes auf vertrauliche Daten oder Kontoanmeldeinformationen oder beides
8) Parameter Manipulation	Pfad-Traversal-Attacken, Befehlsausführung und Umgehung von Zutrittskontrollmechanismen, was zur Offenlegung von Informationen, zur Erhöhung von Privilegien (Systemrechten) und zur Denial-of-Service führen kann.
9) Exception Management	Denial-of-Service und Offenlegung von sensiblen System-Details.
10) Auditing and Logging	Ignorieren von Anzeichen einer Intrusion, Unfähigkeit, die Handlungen eines Benutzers zu beweisen sowie Schwierigkeiten bei der Problemdiagnose.

3. Anforderungen

Nachfolgend sind die wichtigsten Anforderungen bezüglich Entwicklung von Web Applikationen enthalten. Die nachfolgenden Anforderungen aus den OWASP Top Ten stellen nur die am meisten ausgenutzten Schwachstellen dar. Generell muss auch ein Leistungsanbieter / Auftragnehmer entsprechende Vorkehrungen in seinem Entwicklungsprozess analog ISO27001 / NIST treffen um eine, den heutigen Ansprüchen, entsprechend sichere Applikation entwickeln zu können.

Owasp Top 10 Kategorien – Anforderungen zur minmierung von Zugriffen durch nicht autorisierte / authentisierte Personen auf ein System

Gesamtkatalog: OWASP Top 10 Guide

#	**Anforderung** (Schutz vor...)	**Beschreibung**	**Kategorie**	**Referenz (direct Link)**
1.	A1 – Injection	Injektionsfehler, wie SQL-, NoSQL-, OS- und LDAP-Injektion, treten auf, wenn nicht vertrauenswürdige Daten als Teil eines Befehls oder einer Abfrage an einen Interpreter gesendet werden. Die feindlichen Daten des Angreifers können den Interpreter dazu verleiten, unbeabsichtigte Befehle auszuführen oder auf Daten ohne entsprechende Berechtigung zuzugreifen.	Webapplikation/ Webserver	**Injection**
2.	A2 – Broken Authentication	Anwendungsfunktionen im Zusammenhang mit der Authentifizierung und Sitzungsverwaltung sind oft fehlerhaft implementiert, so dass Angreifer Passwörter, Schlüssel oder Sitzungs-Tokens kompromittieren oder andere Implementierungsfehler ausnutzen können, um die Identität anderer Benutzer vorübergehend oder dauerhaft anzunehmen.	Browser/ Webserver / / Webapplikation	**Broken Authentication**
3.	A3 – Sensitive Data Exposer	Viele Webanwendungen und APIs schützen sensible Daten wie Finanz-, Gesundheits- und personenbezogene Daten nicht	Browser/ Webapplikation	**SensitiveData Exposure.**

#	Anforderung (Schutz vor...)	Beschreibung	Kategorie	Referenz (direct Link)
		richtig. Angreifer können solche schwach geschützten Daten stehlen oder verändern, um Kreditkartenbetrug, Identitätsdiebstahl oder andere Verbrechen durchzuführen. Sensible Daten können ohne zusätzlichen Schutz, wie z. B. Verschlüsselung im Ruhezustand oder bei der Übertragung, kompromittiert werden und erfordern besondere Vorsichtsmaßnahmen, wenn sie mit dem Browser ausgetauscht werden.		
4.	A4 – XML Externe Entitäten (XXE).	Viele ältere oder schlecht konfigurierte XML-Prozessoren werten externe Entitätsreferenzen innerhalb von XML-Dokumenten aus. Externe Entitäten können dazu verwendet werden, interne Dateien mithilfe des Datei-URI-Handlers, interne Dateifreigaben, internes Port-Scanning, Remotecodeausführung und Denial-of-Service-Angriffe offenzulegen.	Browser/ Applikation / Datenbank server	**XML External Entities (XXE).**
5.	A5 - Broken Access Control	Beschränkungen, was authentifizierte Benutzer tun dürfen, werden oft nicht richtig durchgesetzt. Angreifer können diese Schwachstellen ausnutzen, um auf nicht autorisierte Funktionen und/oder Daten zuzugreifen, z. B. auf Konten anderer Benutzer zuzugreifen, sensible Dateien anzuzeigen, Daten anderer Benutzer zu ändern, Zugriffsrechte zu ändern usw.	Webserver / Browser/ Datenbank server	**Broken Access Control.**
6.	A6 – Security Misconfiguration.	Fehlkonfiguration der Sicherheit. Eine falsche Sicherheitskonfiguration ist das am häufigsten auftretende Problem. Dies ist häufig eine Folge von unsicheren Standardkonfigurationen, unvollständigen oder Ad-hoc-Konfigurationen, offenem Cloud-Speicher, falsch konfigurierten HTTP-Headern und ausführlichen Fehlermeldungen, die sensible Informationen enthalten. Alle	Datenbank server / Applikation	**Security Misconfiguration.**

#	**Anforderung** (Schutz vor...)	**Beschreibung**	**Kategorie**	**Referenz (direct Link)**
		Betriebssysteme, Frameworks, Bibliotheken und Anwendungen müssen nicht nur sicher konfiguriert, sondern auch zeitnah gepatcht/aktualisiert werden.		
7.	A7 –Cross-Site Scripting (XSS).	XSS-Schwachstellen treten immer dann auf, wenn eine Anwendung nicht vertrauenswürdige Daten ohne ordnungsgemäße Validierung oder Escaping in eine neue Webseite einfügt oder eine bestehende Webseite mit vom Benutzer bereitgestellten Daten unter Verwendung einer Browser-API aktualisiert, die HTML oder JavaScript erstellen kann. XSS ermöglicht es Angreifern, Skripte im Browser des Opfers auszuführen, die Benutzersitzungen entführen, Websites verunstalten oder den Benutzer auf bösartige Websites umleiten können.	Webapplikation / Browser	**Cross-Site Scripting (XSS)**
8.	A8 – Insecure Deserialization	Eine unsichere Deserialisierung führt häufig zu Remotecodeausführung. Selbst wenn Deserialisierungsfehler nicht zu Remotecodeausführung führen, können sie für Angriffe wie Replay-Angriffe, Injektionsangriffe und Angriffe zur Privilegienerweiterung genutzt werden..	Webserver / Webapplikation Code	**Insecure Deserialization.**

#	**Anforderung** (Schutz vor...)	**Beschreibung**	**Kategorie**	**Referenz (direct Link)**
9.	A9 – Nutzung von Komponenten mit bekannten Schwachstellen	Komponenten wie z.B. Bibliotheken, Frameworks oder andere Softwaremodule werden meistens mit vollen Berechtigungen ausgeführt. Wenn eine verwundbare Komponente ausgenutzt wird, kann ein solcher Angriff zu schwerwiegendem Datenverlust oder bis zu einer Serverübernahme führen. Applikationen, die Komponenten mit bekannten Schwachstellen einsetzen, können Schutz-maßnahmen unterlaufen und so zahlreiche Angriffe und Auswirkungen ermöglichen.	Alle involvierten Systeme	**Using Components with Known Vulnerabilities.**
10.	A10 – Insufficient Logging & Monitoring	Unzureichende Protokollierung und Überwachung, gepaart mit fehlender oder ineffektiver Integration mit der Incident Response, ermöglicht es Angreifern, weitere Systeme anzugreifen, die Persistenz aufrechtzuerhalten, auf weitere Systeme überzugehen und Daten zu manipulieren, zu extrahieren oder zu zerstören. Die meisten Studien zu Sicherheitsverletzungen zeigen, dass die Zeit bis zur Entdeckung einer Sicherheitsverletzung mehr als 200 Tage beträgt und in der Regel durch externe Parteien und nicht durch interne Prozesse oder Überwachung entdeckt wird.	Entdecken & Reagieren	**Insufficient Logging & Monitoring**

4. Weitere funktionale Anforderungen

#	Anforderung	Beschreibung
11.	Rollenbasierendes Zugriffskonzept	Die Applikation muss ein Zugriffskonzept implementieren /umsetzen können welches nach dem need to know Prinzip Zugriffe auf Funktionen / Informationen einschränken kann. Das System muss ebenfalls eine Gruppenverwaltung sowie Rollenkonzepte unterstützen um nebst der Authentisierung auch die Autorisierung steuern zu können.
12.	Autorisierung	Die Autorisierung auf die Applikation muss durch die Applikaiton erzwungen werden
13.	Authentisierung	Die Applikation muss die Möglichkeit bieten weitere Authentisierungsmechnismen einzubinden(AAI) ohne die Autorisierungsvorgaben zu unterwandern.
14.	Nicht erfolgreiche Authentisierungsversuche	• Die Applikation unterstützt eine Limitierung der Anzahl nicht erfolgreicher Login Versuche • Die Applikation unterstützt das Locken eines Accounts nach X fehlerhaften Login Versuchen. ➔ Gelockte Accounts können via User Administration freigeschalten werden.
15.	Fehlerinformationen der Applikation	Fehlerinformationen der Applikation werden in einer try / Catch Funktion abgefangen. In der Catch Funktion muss eine neutrale Fehlerinformation auf der Benutzeroberfläche angezeigt werden. →Keine Standardfehler wie bspw. von Datenbanken / Webserver etc. welche Version und Produkt oder das Aufzeigen von fehlerhaften SQL Queries sollen dabei auf der Benutzeroberfläche angezeigt werden.
16.	Letztes erfolgreiches Login	Das System zeigt nach einem Login an, wann sich der benutzte

#	Anforderung	Beschreibung
		Account zum letzten Mal erfolgreiche authentisiert hat.
17.	Keine gleichzeigten / parallelen Sessions des gleichen Accounts	Das System verhindert gleichzeitige Sessions von unterschiedlichen Browserinstanzen.
18.	Session Time out / Session Management / Session Authentizität	➔ Die Applikation unterstützt ein Session Time out den man konfigurieren kann (Standardwerte 180 Min ohne Aktivität, danach erfolgt ein Log out) ➔ Sessions welche unterbrochen werden (bspw. Ausfall Clientseitiger Internet Zugang) müssen geschlossen werden. Keine hängende Sessions / re – etablierte Sessions. ➔ Session sind pro Login über einen eindeutigen Identifyer referenziert. Keine Doubletten möglich.
19.	Logout	Die Applikation stellt beim erfolgreichen Logout sicher, dass alle clientseitigen und schützenswerten Informationen (u.a. Browser Cache / Cookies) gelöscht werden. ➔ Bspw. ein erneutes Login nach erfolgreichem abmelden via der Backfunktion des Browsers darf nicht zu einem automatischem Re – Login führen.
20. .	Logging / Auditierung des Systemes	Die Applikation schreibt Logfiles welche sicherheitsrelevanten Inhalt haben in ein Logfile welches an ein SIEM weitergeleitet werden kann (Syslog Format)
21.	Business Contingency	Das System unterstützt eine redundante Auslegung **bei Bedarf** sowie ein failover von einer aktiven Instanz auf eine Hot by Instanz ohne Datenverlust.
22.	Verschlüsselung	Die Lösung unterstützt eine Verschlüsselung von sensitiven Inhalten (bspw. DB Inhalt) bei Bedarf (vorbereitet).
23.	Applikations Partitionierung	Die Applikation separiert User Funktionen von

#	Anforderung	Beschreibung
		Applikationsmanagement Funktionen
24.	Isolierung von sicherheitsrelevanten Funktionen	Die Applikation trennt sicherheitsrelevante Funktionen von nicht sicherheitrelevanten Funktionen.
25.	Logging	Die Applikation zeichnet wichtige User / System Transaktionen auf und hinterlegt diese in einem Logfile (Syslog).
26.	WAF – Webapplication Firewall	Die Applikation kann mit Webproxies und Webapplication Firewalls umgehen.