OWASP
Open Web Application Security Project

Talk:

OWASP Top 10 – 2017

Die 10 kritischsten Sicherheitsrisiken für Webanwendungen

- Neuerungen
- Hintergründe

OWASP
German Chapter

Aktuelles: Deutsche Version

NEU

Deutschsprachiges Top 10-Team:

- Christian Dresen
- Alexios Fakos
- Louisa Frick
- Torsten Gigler
- Tobias Glemser
- Dr. Frank Gut
- Dr. Ingo Hanke
- Dr. Thomas Herzog
- Dr. Markus Koegel
- Sebastian Klipper
- Jens Liebau
- Ralf Reinhardt
- Martin Riedel
- Michael Schaefer

Beim German OWASP Day 2018 und als Download:

https://www.owasp.org/index.php/Germany/Projekte/Top_10

OWASP
Open Web Application
Security Project

Unsere Mission [1]

- **Awareness**

  Für **Entwickler, Anwendungs-Verantwortliche, Sicherheitstester** und **Manager:**

  – **Sensibilisierung und kompakter Einstieg** in die Sicherheit für Webanwendungen

  – **Verstehen** von (gefundenen) Schwachstellen und **Hilfe** beim Beseitigen

- **Security-by-Default**

  – Motivation für das Programmieren von Tools und Bibliotheken, die bereits mit **Standard-Einstellungen robust gegen Schwachstellen** sind.

Unsere Mission [2]

- **Nutzung als ‚De-Facto-Sicherheitsstandard‘**
  - **Meine Meinung: kein Standard**
    → Vermittelt die Fähigkeit Risiken ‚zu sehen‘
    → ‚Good Practices‘ für die 10 kritischsten Risiken
    → **Guter erster Schritt** für mehr Anwendungssicherheit

OWASP
Open Web Application Security Project

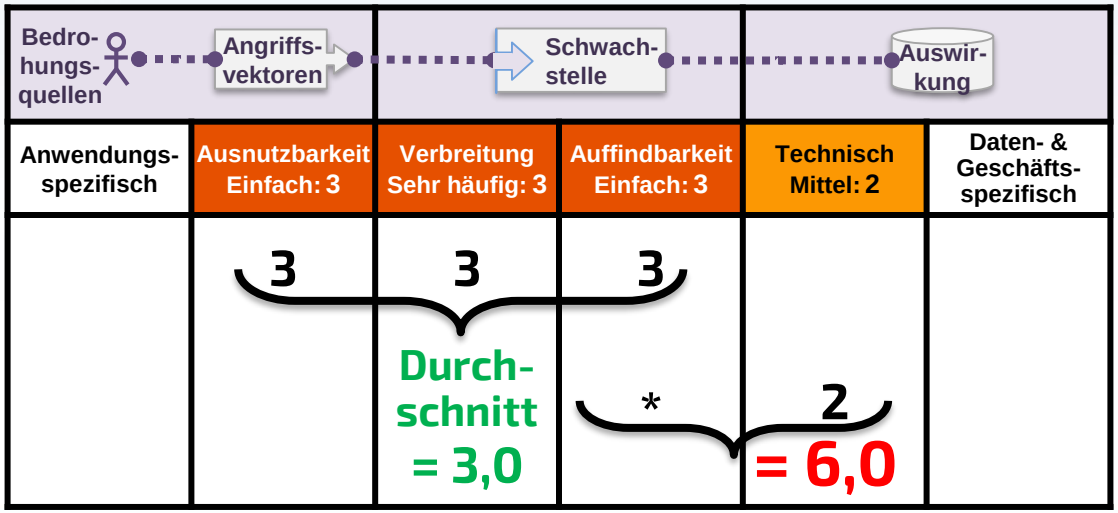
Neuerungen: Methodik [1]

- **Zusammensetzung der 10 Risiken:**
  - **8 Risiken** auf Basis einer **Datenerhebung** [+DAT]
    - **Rückschau**
    - **Häufigkeitsrate** auf Basis der **Anwendungen** mit einer bestimmten Schwachstelle (vorher: Anzahl der Schwachstellen)
    - **Rohdaten und Ergebnisse** sind öffentlich abrufbar
  - **2 Risiken** auf Basis einer **Expertenumfrage in der Community** [+DAT]
    - **Vorausschau**

Neuerungen: Methodik [2]

- **Berechnung der Risiken:**
  - **Risikofaktoren (1-Niedrig … 3-Hoch;** vorher: umgekehrt) [+R]
  - Angabe des **Werts** [+RF]

Bedro-hungs-quellen	Ausnutzbarkeit	Schwachstelle Verbreitung	Schwachstelle Auffindbarkeit	Technische Auswirkungen	Auswirkun-gen auf das Unternehmen
Anwen-dungs-spezifisch	Einfach: 3	Sehr häufig: 3	Einfach: 3	Schwer-wiegend: 3	Daten- & Geschäfts-spezifisch
	Durch-schnittlich: 2	Häufig: 2	Durch-schnittlich: 2	Mittel: 2	
	Schwierig: 1	Selten: 1	Schwierig: 1	Gering: 1	



Neuerungen: Risiken

OWASP Top 10 - 2017:

OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Fehler in Authentifizierung und Session-Mgmt.	→	A2:2017-Fehler in der Authentifizierung
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Verlust der Vertraulichkeit sensibler Daten
A4 – Unsichere direkte Objektreferenzen [mit A7]	U	A4:2017-XML External Entities (XXE) [NEU]
A5 – Sicherheitsrelevante Fehlkonfiguration	↘	A5:2017-Fehler in der Zugriffskontrolle [vereint]
A6 – Verlust der Vertraulichkeit sensibler Daten	↗	A6:2017-Sicherheitsrelevante Fehlkonfiguration
A7 – Fehlerhafte Autorisierung auf Anw.-Ebene [mit A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Unsichere Deserialisierung [NEU, Community]
A9 – Nutzung von Komponenten mit bekannten Schwachstellen	→	A9:2017-Nutzung von Komponenten mit bekannten Schwachstellen
A10 – Ungeprüfte Um- und Weiterleitungen	☒	A10:2017-Unzureichendes Logging & Monitoring [NEU, Community]

NEU

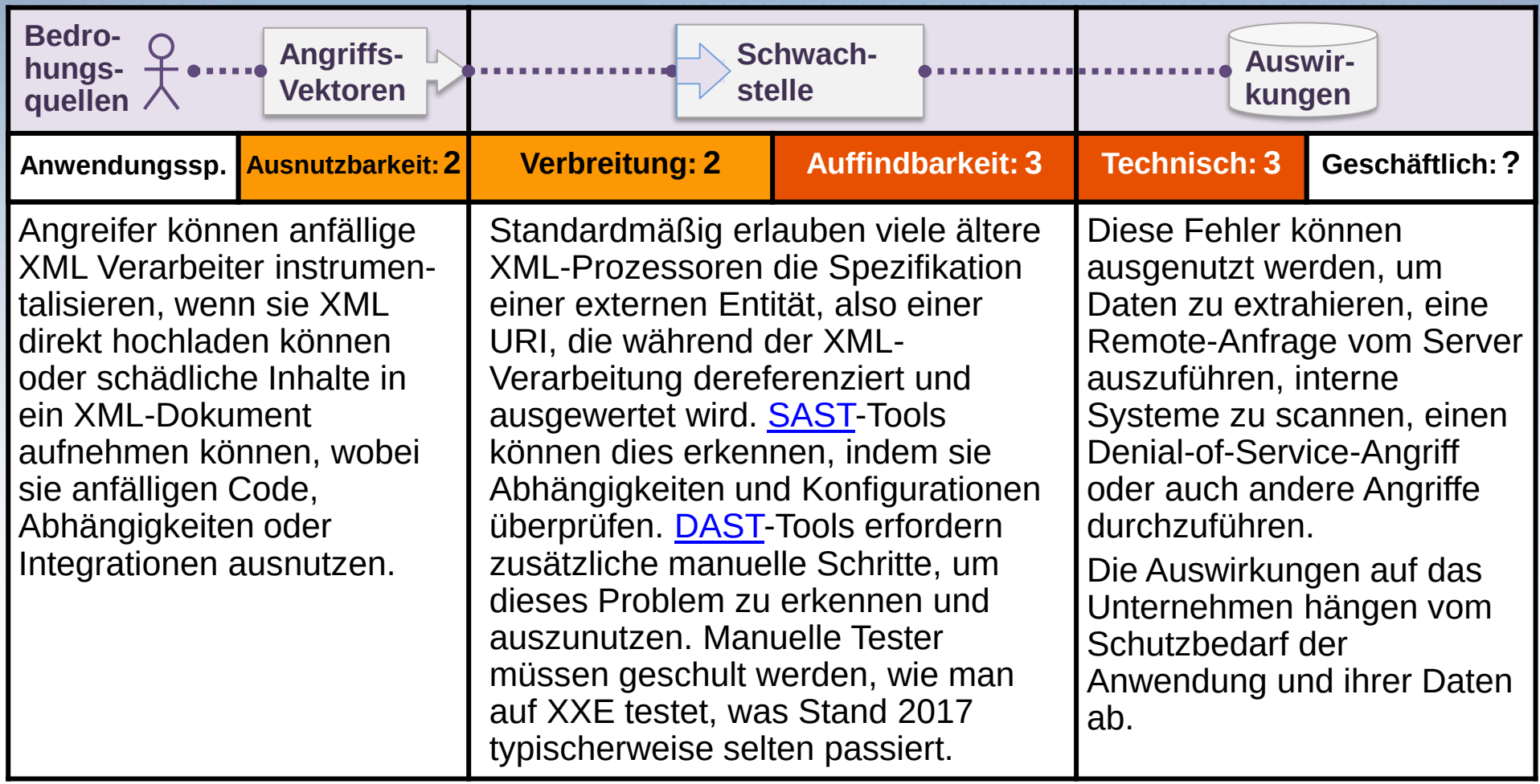
NEU, Community

NEU, Community

OWASP Open Web Application Security Project

A4 :2017

XML External Entities (XXE) [1]



OWASP Open Web Application Security Project

A4 :2017

XML External Entities (XXE) [2]

Ist die Anwendung verwundbar?

Anwendungen und insbesondere XML-basierte Webservices oder nachgelagerte Integrationen können in folgenden Fällen anfällig für Angriffe sein:

- Die Anwendung akzeptiert direkt XML oder XML-Uploads, insbesondere aus nicht vertrauens-würdigen Quellen oder fügt nicht vertrauenswürdige Daten in XML-Dokumente ein, die dann von einem XML-Prozessor analysiert werden.
- Die XML-Prozessoren in der Anwendung oder SOAP-basierte Webservices haben Document Type Definitions (DTDs) aktiviert. Da der genaue Mechanismus zum Deaktivieren der DTD-Verarbeitung je nach Prozessor variiert, ist es empfehlenswert, eine Referenz wie den OWASP Cheat Sheet 'XXE Prevention' zu konsultieren.
- Wenn Ihre Anwendung SAML für die Identitätsverarbeitung im Rahmen von föderierter Sicherheit oder für Single Sign On (SSO) Zwecke verwendet. SAML verwendet XML für Identitäts-bekundungen und kann daher anfällig sein.
- Wenn die Anwendung SOAP vor Version 1.2 verwendet, ist sie wahrscheinlich anfällig für XXE-Angriffe, wenn XML-Entitäten an das SOAP-Framework übergeben werden.
- Die Anfälligkeit für XXE-Angriffe bedeutet wahrscheinlich, dass die Anwendung anfällig für Denial-of-Service-Angriffe, einschließlich des sogenannten "Billion Laughs" Angriffs, ist.

OWASP Open Web Application Security Project

A4 :2017

XML External Entities (XXE) [3]

Mögliche Angriffsszenarien

Zahlreiche öffentliche XXE-Probleme wurden entdeckt, darunter auch Angriffe auf Embedded-Geräte. XXE tritt an vielen unerwarteten Stellen auf, einschließlich tief verschachtelter Abhängigkeiten. Der einfachste Weg, wenn möglich, ist das Hochladen einer bösartigen XML-Datei:

Szenario 1: Der Angreifer versucht, Daten vom Server zu extrahieren:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
  <foo>&xxe;</foo>
```

Szenario 2: Ein Angreifer durchsucht das private Netzwerk des Servers, indem er die obige ENTITY-Zeile ändert zu:

```
<!ENTITY xxe SYSTEM "https://192.168.1.1/private" >]>
```

Szenario 3: Ein Angreifer versucht einen Denial-of-Service-Angriff, indem er eine potenziell endlose Datei einfügt:

```
<!ENTITY xxe SYSTEM "file:///dev/random" >]>
```

OWASP Open Web Application Security Project

XML External Entities (XXE) [4]

Wie kann ich das verhindern?

Die Schulung von Entwicklern ist unerlässlich, um XXE zu identifizieren und zu beheben. Zusätzlich:

- Verwenden Sie möglichst weniger komplexe Datenformate, wie JSON und vermeiden Sie die Serialisierung sensibler Daten.
- Patchen oder aktualisieren Sie alle XML-Prozessoren und Bibliotheken, die von der Anwendung oder dem zugrunde liegenden Betriebssystem verwendet werden. Verwenden Sie Werkzeuge zur Prüfung von Abhängigkeiten. Aktualisieren Sie auf SOAP 1.2 oder höher.
- Deaktivieren Sie die Verarbeitung von externen XML-Entitäten und DTDs in allen XML-Parsern in der Anwendung, gemäß dem OWASP Cheat Sheet 'XXE Prevention'. Implementierung einer positiven serverseitigen Eingabevalidierung ("Whitelisting"), -filterung oder -bereinigung, um bösartige Daten in XML-Dokumenten, Headern oder Knoten zu verhindern.
- Vergewissern Sie sich, dass die Upload-Funktionalität für XML- oder XSL-Dateien eingehende XML-Daten mithilfe der XSD-Validierung oder ähnlichem validiert.
- SAST-Tools können helfen, XXE im Quellcode zu erkennen, jedoch ist die manuelle Codeüberprüfung die beste Alternative in großen, komplexen Anwendungen mit vielen Integrationen.
- Wenn dies nicht möglich ist, sollten Sie die Verwendung von virtuellen Patches, API-Sicherheitsgateways oder Web Application Firewalls (WAFs) in Betracht ziehen, um XXE-Angriffe zu erkennen, zu überwachen und zu blockieren.

OWASP Open Web Application Security Project

XML External Entities (XXE) [5]

Referenzen

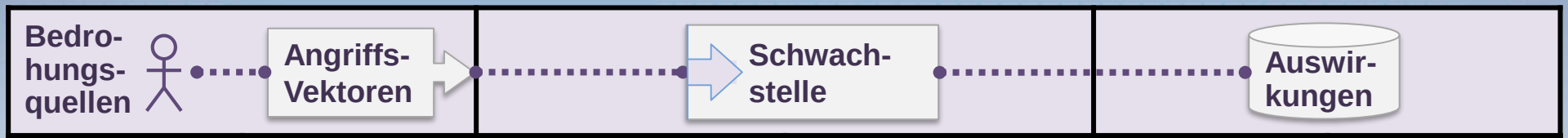
OWASP

- OWASP Application Security Verification Standard
- OWASP Testing Guide: Testing for XML Injection
- OWASP XXE Vulnerability
- OWASP Cheat Sheet: XXE Prevention
- OWASP Cheat Sheet: XML Security

Andere

- CWE-611: Improper Restriction of XXE
- Billion Laughs Attack
- SAML Security XML External Entity Attack
- Detecting and exploiting XXE in SAML Interfaces

A8 :2017 Unsichere Deserialisierung [1]



Anwendungssp.	Ausnutzbarkeit: 1	Verbreitung: 2	Auffindbarkeit: 2	Technisch: 3	Geschäftlich: ?
Das Ausnutzen von Fehlern in der Deserialisierung ist nicht trivial, zumal vorhandener Angriffscode selten ohne weitere Anpassungen einsetzbar ist.		Dieser Eintrag in den Top 10 basiert auf einer Expertenumfrage in der Community und nicht auf messbaren Fallzahlen. Einige Werkzeuge können Deserialisierungsschwachstellen entdecken, allerdings ist häufig eine manuelle Überprüfung des Fundes nötig. Es ist zu erwarten, dass belastbareres Zahlenmaterial zur Verfügung stehen wird, sobald die Tools zur Erkennung weiter entwickelt sind.		Die Auswirkungen von Deserialisierungsfehlern sollten nicht unterschätzt werden. Diese Schwachstelle kann durchaus zu "Remote-Code Execution" führen, einem der schwerwiegendsten Angriffe überhaupt. Die Auswirkungen auf das Unternehmen hängen vom Schutzbedarf der Anwendung und ihrer Daten ab.	

A8 :2017

Unsichere Deserialisierung [2]

Ist die Anwendung verwundbar?

Anwendungen oder APIs können verwundbar sein, wenn sie bösartige oder vom Angreifer manipulierte Objekte deserialisieren.

Dies kann zu zwei Hauptangriffsarten führen:

- Angriffe mittels Objekt- und Datenstrukturen, die es Angreifern ermöglichen, die Anwendungslogik zu verändern oder Programmcode auszuführen. Dies ist möglich, sofern die Anwendung auf Klassen zugreifen kann (inkl. Standardklassen), deren Verhalten während oder nach der Deserialisierung manipuliert werden kann.
- Übliche Angriffe mittels Datenmanipulation: dazu zählen Angriffe gegen die Zugriffskontrolle, wobei existierende Datenstrukturen genutzt und deren Inhalt manipuliert werden.

Serialisierung wird häufig eingesetzt bei:

- Remote- und Inter-Prozess Kommunikation (RPC/IPC)
- Wire-Protokollen, Webservices, Message-Brokern
- Caching/Persistenz
- Datenbanken, Cache-Servern, Dateisystemen
- HTTP-Cookies, HTML-Formular-Parameter oder API- Authentifizierungs-Token.

OWASP Open Web Application Security Project

A8 :2017

Unsichere Deserialisierung [3]

Mögliche Angriffsszenarien

Szenario 1: Eine React basierte Anwendung nutzt einige Spring Boot-Microservices. Die Programmierer dieser funktionalen Sprache haben darauf geachtet, dass ihr Programmcode „unveränderbar“ ist. Daher serialisieren Sie den Benutzerstatus und transferieren diesen so mit jeder Anfrage hin und her. Ein Angreifer entdeckt die „r00“-Base64-Signatur des Java-Objekts und nutzt das Werkzeug Java Serial Killer, um Remote-Code-Execution auf dem Anwendungsserver auszuführen.

Szenario 2: Ein PHP Forum nutzt die PHP-Objekt-Serialisierung um ein „Super-Cookie“ zu erzeugen, dieses enthält Angaben zur User-ID, Rolle, einen Passwort-Hash und weitere Informationen:

a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user";

i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}

Ein Angreifer verändert nun das serialisierte Objekt, um sich selbst Admin-Rechte zu verschaffen.

a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin";

i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}

OWASP Open Web Application Security Project

A8 :2017

Unsichere Deserialisierung [4]

Wie kann ich das verhindern?

Der einzig sichere Weg ist es keine serialisierten Objekte aus nicht vertrauenswürdigen Quellen anzunehmen oder nur serialisierte Datenstrukturen zu nutzen, die ausschließlich einfache Datentypen erlauben.

Andernfalls ziehen Sie folgende Empfehlungen in Betracht:

- Versehen Sie alle serialisierten Objekte mit einer digitalen Signatur, um so zu verhindern, dass bösartige Objekte erzeugt oder Daten manipuliert werden können.
- Achten Sie auf eine strikte Typisierung während der Deserialisierung und bevor Objekte erzeugt werden. Zumeist wird hier nur eine bekannte Menge an Klassen benötigt. Es wurde bereits gezeigt, das diese Maßnahme umgangen werden kann. Es ist daher nicht ratsam, sich alleine hierauf zu verlassen.
- Isolieren Sie den für die Deserialisierung zuständigen Programmcode und führen Sie ihn in einer eigenen Umgebung mit möglichst geringen Berechtigungen aus.
- Protokollieren Sie alle Ausnahmefehler, die bei der Deserialisierung auftreten (z.B. unerwartete Objekt-Typen).
- Begrenzen oder überwachen Sie ein- und ausgehende Netzwerkaktivitäten von Containern oder Servern, die Deserialisierungen ausführen.
- Überwachen und melden Sie, wenn ein Nutzer auffällig häufig eine Deserialisierung nutzt.

A8 :2017

Unsichere Deserialisierung [5]

Referenzen

OWASP

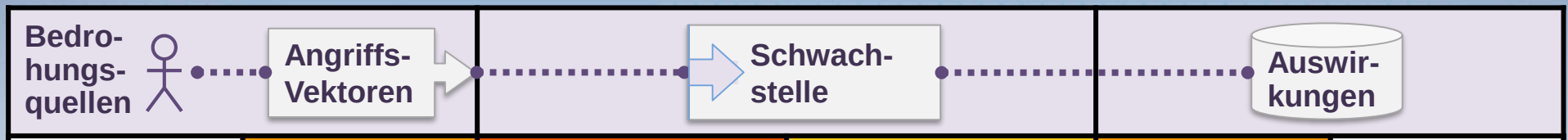
- OWASP Cheat Sheet: Deserialization
- OWASP Proactive Controls: Validate All Inputs
- OWASP Application Security Verification Standard
- OWASP AppSecEU 2016: Surviving the Java Deserialization Apocalypse
- OWASP AppSecUSA 2017: Friday the 13th JSON Attacks

Andere

- CWE-502: Deserialization of Untrusted Data
- Java Unmarshaller Security
- OWASP AppSec Cali 2015: Marshalling Pickles

OWASP Open Web Application Security Project

A10 :2017 Unzureichendes Logging & Monitoring [1]



Anwendungssp.	Ausnutzbarkeit: 2	Verbreitung: 3	Auffindbarkeit: 1	Technisch: 2	Geschäftlich: ?
Das Ausnutzen unzureichender Protokollierungs- und Monitoring-Maßnahmen ist der Ausgangspunkt fast aller größerer Sicherheitsvorfälle. Angreifer nutzen fehlendes Monitoring und verzögerte Antwortzeiten auf Vorfälle dazu aus, unentdeckt Angriffe durchzuführen.		Dieser Eintrag in den Top 10 basiert auf einer Umfrage unter Sicherheitsexperten. Eine mögliche Strategie, um herauszufinden, ob Ihre Monitoring-Maßnahmen ausreichend sind, ist es, die Logging-Einträge Ihres Systems nach einem Penetrationstest zu überprüfen. Die Aktivitäten des Testers sollten so protokolliert worden sein, das Sie daraus mögliche Schäden identifizieren können.		Den meisten erfolgreichen Angriffen gehen Schwachstellen-Scans voraus. Wenn solche Scans nicht abgewehrt werden, besteht ein fast 100 prozentiges Risiko für erfolgreiche Angriffe. Die Zeit bis zur Aufdeckung eines Einbruchs lag 2016 durchschnittlich bei 191 Tagen – viel Zeit, um Ihren Systemen Schaden zuzufügen.	

OWASP Open Web Application Security Project

A10 :2017

Unzureichendes Logging & Monitoring [2]

Ist die Anwendung verwundbar?

Unzureichende Protokollierungs-, Erkennungs- und Monitoring- Maßnahmen sowie fehlende aktive Reaktionen auf Vorfälle treten ständig auf:

- Auditierbare Ereignisse wie erfolgreiche oder fehlgeschlagene Logins oder wichtige Transaktionen werden nicht protokolliert.
- Warnungen und Fehler erzeugen keine, unzureichende oder uneindeutige Protokoll-Einträge.
- Protokolle von Anwendungen und Schnittstellen werden nicht ausreichend hinsichtlich verdächtiger Aktivitäten überprüft.
- Protokolle werden nur lokal gespeichert.
- Geeignete Alarmierungs-Schwellen und Eskalations-Prozesse als Reaktion auf (potentielle) Vorfälle liegen nicht vor oder sind nicht wirksam.
- Penetration-Tests und Scans mit DAST-Werkzeugen (wie OWASP ZAP) lösen keine Alarme aus.
- Die eingesetzten Überwachungsverfahren sind nicht in der Lage aktive Angriffe zu erkennen und in Echtzeit oder nahezu Echtzeit Alarm auszulösen.

Wenn Ihre Systeme Protokollierungs- und Alarmierungs-Nachrichten Benutzern oder Angreifern preisgeben, kann dies zum Abfluss von Daten führen (siehe **A3:2017-Verlust der Vertraulichkeit sensibler Daten**).

OWASP Open Web Application Security Project

Unzureichendes Logging & Monitoring [3]

Mögliche Angriffsszenarien

Szenario 1: Eine Open-Source Projektforums-Software, die von einem kleinen Team betrieben wird, wurde auf Grund eines Fehlers in der Software angegriffen. Die Angreifer konnten das interne Quellcode-Repository mit der nächsten Version und allen Inhalten löschen. Obwohl der Quellcode wiederhergestellt werden konnte, führte das Fehlen von Monitoring, Protokollierung und Warnmeldungen zu weit schwerwiegenderen Folgen. Als Konsequenz ist das Projektforum inzwischen nicht mehr aktiv.

Szenario 2: Angreifer scannen nach Nutzern mit häufig benutzten, einfachen Passwörtern. Sie können alle betroffenen Accounts übernehmen. Für alle anderen Nutzer hinterlässt dieser Angriff nur 1 falschen Loginversuch. Nach einiger Zeit könnte der Angriff mit anderen Passwörtern wiederholt werden.

Szenario 3: Ein großer US-Großhändler verfügte über eine Sandbox, die Mail-Anhänge auf Schadsoftware überprüfte. Die Sandbox entdeckte potenziell gefährliche Software, aber niemand reagierte auf diese Meldung. Der Sicherheitsvorfall wurde jedoch erst erkannt, als die Hausbank betrügerische Kreditkarten-Transaktionen meldete.

A10 :2017 Unzureichendes Logging & Monitoring [4]

Wie kann ich das verhindern?

Führen Sie für alle von Anwendungen gespeicherten oder prozessierten Daten folgende Maßnahmen durch:

- Stellen Sie sicher, dass alle erfolglosen Login- und Zugriffs-Versuche und Fehler bei der serverseitigen Eingabevalidierung mit aussagekräftigem Benutzerkontext protokolliert werden, um verdächtige oder schädliche Accounts zu identifizieren. Halten Sie diese Informationen ausreichend lange vor, um auch später forensische Analysen vorzunehmen zu können.
- Stellen Sie sicher, dass Protokollierungen in einem Format erstellt werden, die eine einfache Verarbeitung durch zentrale Protokollanalyse- und -managementwerkzeuge ermöglicht.
- Speichern Sie für wichtige Transaktionen Audit-Trails mit Integritätsschutz, um Verfälschung oder ein Löschen zu verhindern, z.B. durch Einsatz von Datenbanktabellen, die nur das Anhängen von Datensätzen zulassen.
- Richten Sie wirksame Monitoring- und Alarmierungs-Verfahren ein, damit verdächtige Aktivitäten zeitnah entdeckt und bearbeitet werden.
- Etablieren Sie Notfall- und Wiederherstellungspläne für Sicherheitsvorfälle, z.B. auf Basis von NIST 800-61 rev 2.

Es gibt kommerzielle o. Open-Source-Frameworks zum Schutz Ihrer Anwendungen, wie OWASP AppSensor, WebApp Firewalls wie ModSecurity mit dem OWASP ModSecurity Core Rule Set und geeignete Protokollanalyse-Werkzeuge inkl. Alarmierung.

OWASP Open Web Application Security Project

A10 :2017

Unzureichendes Logging & Monitoring [5]

Referenzen

OWASP

- OWASP Proactive Controls: Implement Logging and Intrusion Detection
- OWASP Application Security Verification Standard: V8 Logging and Monitoring
- OWASP Testing Guide: Testing for Error Code (OTG-ERR-001)
- OWASP Cheat Sheet: Logging

Andere

- CWE-223: Omission of Security-relevant Information
- CWE-778: Insufficient Logging

OWASP Open Web Application Security Project

Empfehlung: Nächste Schritte für…

- ‚Rollenbezogene‘ Seiten:
  - Software-Entwickler [+E]
  - Sicherheitstester [+T]
  - Organisationen [+O]
  - Anwendungs-Verantwortliche [+A] 

- Geben Hinweise auf
  - weitere Vorgehensweise
  - Prozesse
  - weitere Sicherheitsmaßnahmen und ‚Best Practices‘
  - zusätzliche Dokumente und Tools von OWASP

+A Nächste Schritte für Anwendungs-Verantwortliche [1]

Regeln Sie den vollständigen Lebenszyklus von Anwendungen

Anwendungen gehören zu den komplexesten Systemen, die Menschen regelmäßig erschaffen und betreiben. Das IT-Management von Anwendungen sollte von IT-Spezialisten erfolgen, die für den vollständigen Lebenszyklus einer Anwendung verantwortlich sind. Wir empfehlen, die Rolle des Anwendungs-Verantwortlichen (Application Manager) als technisches Pendant zum Anwendungs-Eigentümer (Application Owner) zu etablieren. Der Anwendungs-Verantwortliche ist für den gesamten Lebenszyklus der Anwendung bezüglich der IT-Belange zuständig, von der Erhebung bis hin zur Außerbetriebnahme der Systeme. Letzteres wird häufig übersehen.

Anforderungs- und Ressourcen-Management

- Fachliche Anforderungen mit dem Fachbereich aufnehmen und vereinbaren, inkl. dem Schutzbedarf aller Daten-Assets in Bezug auf Vertraulichkeit, Authentizität, Integrität und Verfügbarkeit, sowie der erwarteten Anwendungslogik.
- Zusammenstellen der technischen Anforderungen inkl. funktionalen und nicht-funktionalen Anforderungen an die Sicherheit.
- Planen und vereinbaren des Budgets, das alle Aspekte abdeckt, vom Design, Entwicklung, Testen bis hin zum Betrieb sowie die Sicherheitsmaßnahmen.

OWASP Open Web Application Security Project

+A Nächste Schritte für Anwendungs-Verantwortliche [2]

Ausschreibung und Vergabe

- Die Anforderungen mit internen oder externen Entwicklern vereinbaren, inkl. Richtlinien, Sicherheits-Vorgaben und -Prozesse, wie z.B. sicherer Software-entwicklungsprozess (SDLC), Best Practices.
- Bewerten Sie den Erfüllungsgrad der technischen Anforderungen inkl. Planungs- und Design-Phase.
- Vereinbaren Sie alle technischen Anforderungen inkl. Design, Sicherheit und Service-Level-Agreements (SLAs).
- Nutzen Sie Vorlagen und Checklisten, z.B. den OWASP Secure Software Contract Annex (deutsch).
  Hinweis: Das Dokument ist ausschließlich als Orientierungshilfe anzusehen, es bezieht sich auf US-Recht. Konsultieren Sie in jedem Fall einen spezialisierten Anwalt, bevor Sie es benutzen.

Planung und Design

- Vereinbaren Sie die Planung und das Design der Anwendung mit den Entwicklern und internen Stakeholdern, z.B. Sicherheits-Spezialisten.
- Definieren Sie, unterstützt von Sicherheits-Spezialisten, die Sicherheits-Architektur, allgemeine vorbeugende Maßnahmen und gezielte Gegenmaßnahmen entsprechend dem Schutzbedarf und dem erwarteten Gefährdungsniveau.
- Stellen Sie sicher, dass der Anwendungseigentümer Restrisiken akzeptiert oder zusätzliches Budget bereitstellt.
- Stellen Sie sicher, dass es in jedem Sprint Sicherheits-Stories enthalten sind, die Auflagen für nicht-funktionale Anforderungen enthalten.

OWASP Open Web Application Security Project

+A Nächste Schritte für Anwendungs-Verantwortliche [3]

Deploy-ment, Testen und Rollout

- Automatisieren Sie das Deployment von Anwendungen, Schnittstellen und allen benötigten Komponenten mit sicheren Konfigurationsvoreinstellungen, inkl. der benötigen Berechtigungen.
- Testen Sie die technischen Funktionen und die Integration in die IT-Architektur, koordinieren Sie fachliche Tests.
- Erzeugen Sie “Use-" und “Abuse-Testfälle” aus technischer und fachlicher Sicht.
- Koordinieren Sie Sicherheits-Tests gemäß den internen Prozessen, dem Schutzbedarf und dem angenommenen Gefährdungsniveau der Anwendung.
- Nehmen Sie die Anwendung in Betrieb und übernehmen Sie ggf. Daten aus Altanwendungen.
- Vervollständigen Sie die Dokumentation, inkl. in der Configuration Management Data Base (CMDB) und die Sicherheitsarchitektur.

Betrieb und Change-Manage-ment

- Das Betriebshandbuch muss Vorgaben für den sicheren Betrieb der Anwendung enthalten, z.B. Patchmanagement.
- Sensibilisieren Sie die Anwender für Sicherheitsaspekte und lösen Sie Konflikte zwischen Benutzbarkeit und Sicherheit.
- Planen und begleiten Sie Changes, z.B. Versionswechsel der Anwendung oder anderer Komponenten wie das Betriebssystem, Middleware und Bibliotheken.
- Aktualisieren Sie die vollständige Dokumentation, inkl. der CMDB, der Sicherheitsarchitektur, vorbeugende Maßnahmen, Gegenmaßnahmen und das Betriebshandbuch.

+A Nächste Schritte für Anwendungs-Verantwortliche [4]

Außer-betrieb-nahme von Anwen-dungen

- Weiterhin benötigte Daten sollten archiviert werden, alle anderen Daten sollten sicher gelöscht werden.
- Nehmen Sie die Anwendung auf sichere Weise außer Betrieb, inkl. dem Löschen von nicht mehr benötigten Benutzerkonten, Rollen und Rechten.
- Ändern Sie den Zustand der Anwendung in der CMDB auf "außer Betrieb“.

OWASP Open Web Application Security Project

Dein/Ihr Einsatz

OWASP
Open Web Application Security Project