OWASP
Open Web Application
Security Project

Leitlinien und Werkzeuge von der OWASP-Community

TechDays München, 03. Juni 2019
Symposium: Software Security
Torsten Gigler

Über mich: Torsten Gigler

- **Interner IT-Sicherheitsberater bei einer Bank**
  spezialisiert auf IT-Infrastruktur- und Anwendungs-Sicherheit (>20 Jahre)

- **OWASP-Volunteer seit über 6 Jahren, z. B.:**
  – OWASP Top 10 – 2017 (Co-Leader, Mitarbeit bei der deutschen Version)
  – OWASP Top 10 – 2013 (Contributor, Mitarbeit bei der deutschen Version)
  – OWASP Stammtisch München (Mitorganisator seit >4 Jahren)
  – OWASP-Germany (Mitglied im Chapter-Board seit >1 Jahr)
  – O-Saft – OWASP SSL Advanced Forensic Tool (Co-Entwickler seit >5 Jahren)

Email: **torsten.gigler@owasp.org**

Inhalt

- **Über OWASP**
- **Leitlinien (Beispiele)**
- **Werkzeuge (Beispiele)**
- **Übersicht über Leitlinien und Werkzeuge der OWASP-Community**
- **OWASP Deutschland**

Über OWASP

- Open Web Application Security Project
- **unabhängige, weltweite Community (seit 2001)**
- **OWASP Foundation**: gemeinnützige Organisation (US-Recht)
- **Ziele:**
  - Bedeutung der **Sicherheit von (Web-)Anwendungen** »sichtbar machen«.
  - **Know-How** zur Entwicklung und den Betrieb sicherer (Web-)Anwendungen verbreiten.
  - OWASP Werkzeuge, Dokumente, Videos, Präsentationen und Chapter sind **frei verfügbar**.

Links: https://www.owasp.org, OWASP-Deutschland: https://www.owasp.de

Leitlinien (Beispiele)

OWASP Top 10

- **Die 10 kritischsten Sicherheitsrisiken für Webanwendungen**
- **Sensibilisierung/Awareness**

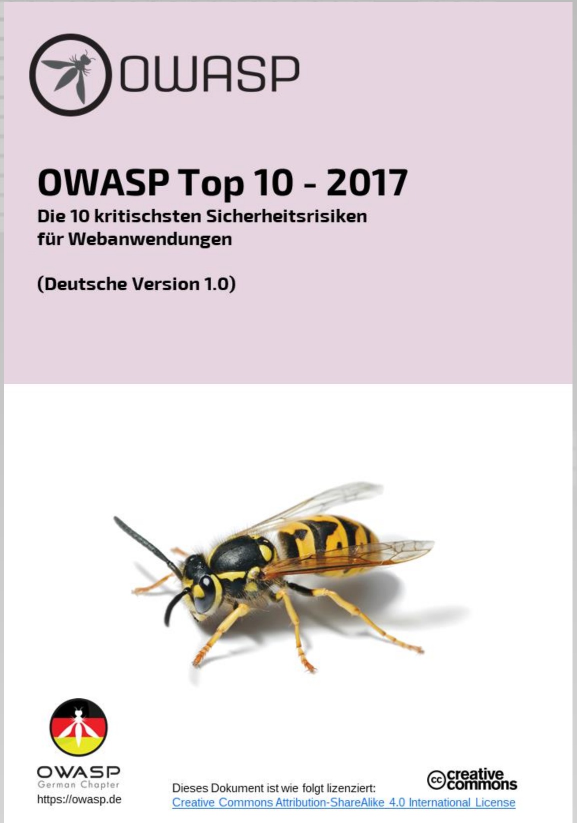
  Für **Entwickler, Anwendungs-Verantwortliche, Sicherheitstester** und **Manager:**

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

  – **Verstehen** von (gefundenen) Schwachstellen und **Hilfe** beim Beseitigen
- **Nutzung als ‚De-Facto-Sicherheitsstandard'?**

  – **Guter, erster Schritt** für mehr Anwendungssicherheit

  – **Keine** ‚Checkliste', kein ‚Sicherheitsstandard'!



OWASP

OWASP Top 10 - 2017

Die 10 kritischsten Sicherheitsrisiken für Webanwendungen

(Deutsche Version 1.0)

OWASP German Chapter https://owasp.de

Dieses Dokument ist wie folgt lizenziert: Creative Commons Attribution-ShareAlike 4.0 International License

creative commons

Link: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

OWASP Top 10

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]	∪	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]	∪	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

Link: https://www.owasp.org/index.php/Germany/Projekte/Top_10 (Deutsche Version)

OWASP Top 10

Empfehlung: ‚Rollenbezogene' Seiten ‚Nächste Schritte für'…

- Software-Entwickler
- Sicherheitstester
- Organisationen
- Anwendungs-Verantwortliche

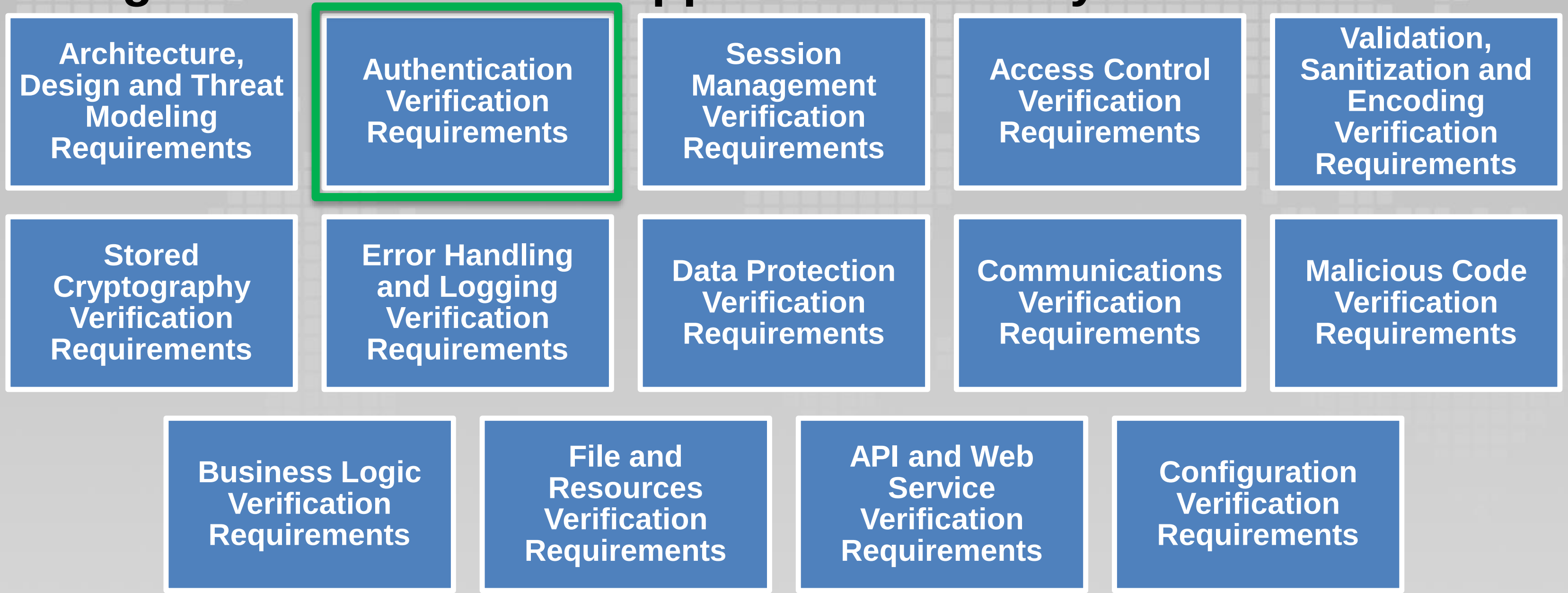
Geben Hinweise auf

- weitere Vorgehensweise
- Prozesse
- weitere Sicherheitsmaßnahmen und ‚Best Practices'
- zusätzliche Richtlinien und Werkzeuge von OWASP

Link: https://www.owasp.org/index.php/Germany/Projekte/Top_10 (Deutsche Version)

Application Security Verification Standard (ASVS)

- **Designed to be an actual application security standard:**



Architecture, Design and Threat Modeling Requirements

Authentication Verification Requirements

Session Management Verification Requirements

Access Control Verification Requirements

Validation, Sanitization and Encoding Verification Requirements

Stored Cryptography Verification Requirements

Error Handling and Logging Verification Requirements

Data Protection Verification Requirements

Communications Verification Requirements

Malicious Code Verification Requirements

Business Logic Verification Requirements

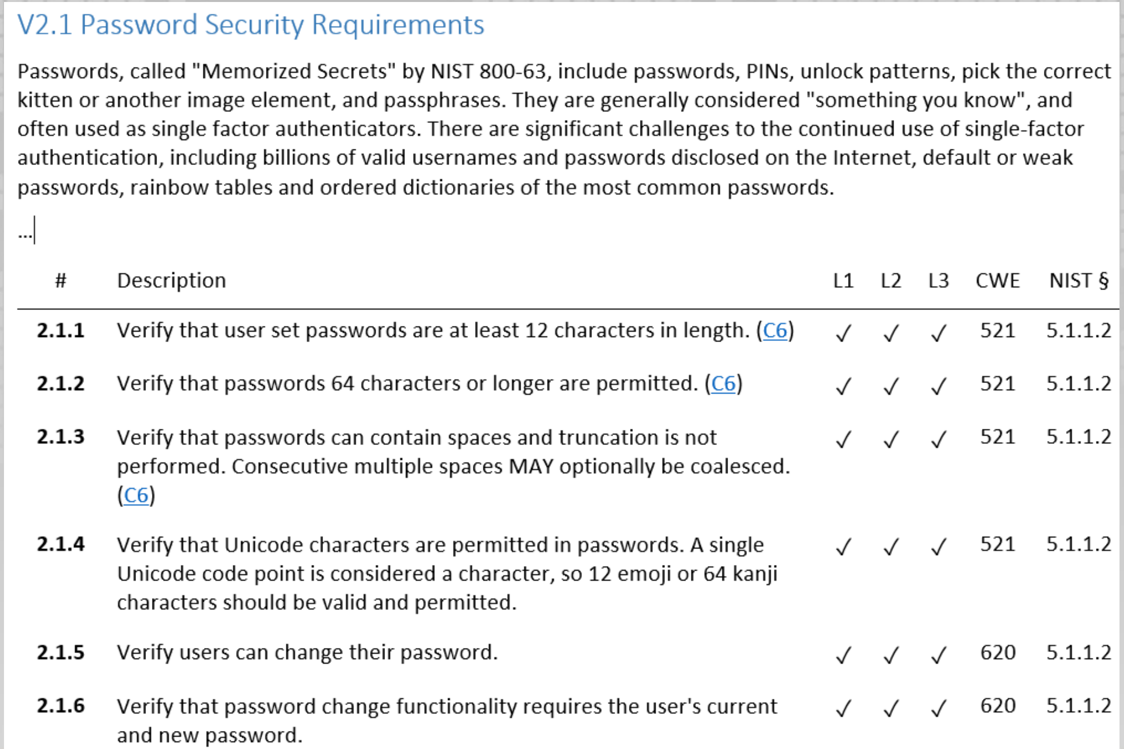
File and Resources Verification Requirements

API and Web Service Verification Requirements

Configuration Verification Requirements

Link: https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project

Application Security Verification Standard (ASVS)



V2.1 Password Security Requirements

Passwords, called "Memorized Secrets" by NIST 800-63, include passwords, PINs, unlock patterns, pick the correct kitten or another image element, and passphrases. They are generally considered "something you know", and often used as single factor authenticators. There are significant challenges to the continued use of single-factor authentication, including billions of valid usernames and passwords disclosed on the Internet, default or weak passwords, rainbow tables and ordered dictionaries of the most common passwords.

...

#	Description	L1	L2	L3	CWE	NIST §
2.1.1	Verify that user set passwords are at least 12 characters in length. (C6)	✓	✓	✓	521	5.1.1.2
2.1.2	Verify that passwords 64 characters or longer are permitted. (C6)	✓	✓	✓	521	5.1.1.2
2.1.3	Verify that passwords can contain spaces and truncation is not performed. Consecutive multiple spaces MAY optionally be coalesced. (C6)	✓	✓	✓	521	5.1.1.2
2.1.4	Verify that Unicode characters are permitted in passwords. A single Unicode code point is considered a character, so 12 emoji or 64 kanji characters should be valid and permitted.	✓	✓	✓	521	5.1.1.2
2.1.5	Verify users can change their password.	✓	✓	✓	620	5.1.1.2
2.1.6	Verify that password change functionality requires the user's current and new password.	✓	✓	✓	620	5.1.1.2

Link: https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project

Cheat Sheet Series

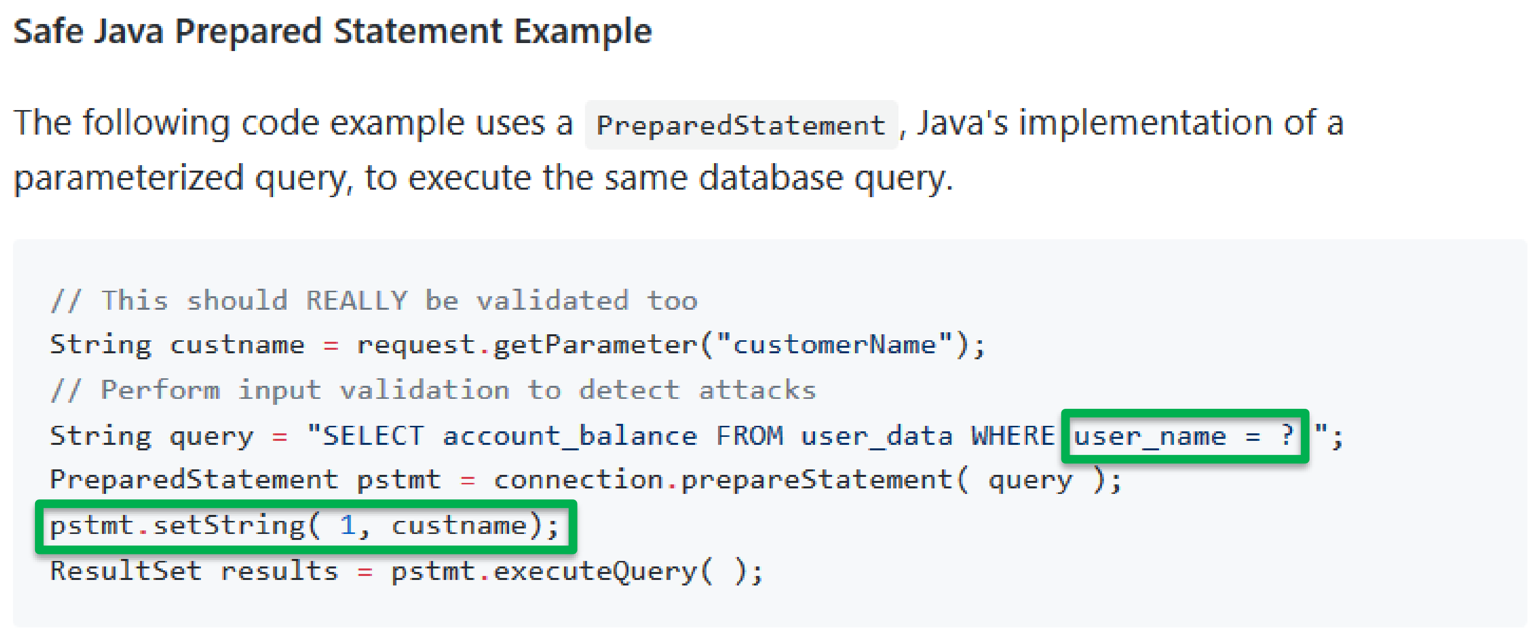
- **Aktuell sind 61 technische Spickzettel (Cheat Sheets) verfügbar:**

Abuse Case	Access Control	Cross Site Scripting Prevention	LDAP Injection Prevention	Logging
Password Storage	REST Security	Session Management	SQL Injection Prevention	Transport Layer Protection
Vulnerable Dependency Management	Web Service Security	XML External Entity Prevention	XML Security	…

Link: https://www.owasp.org/index.php/OWASP_Cheat_Sheet_Series

Cheat Sheet Series

- **Beispiel: SQL Injection Prevention Cheat Sheet**



Safe Java Prepared Statement Example

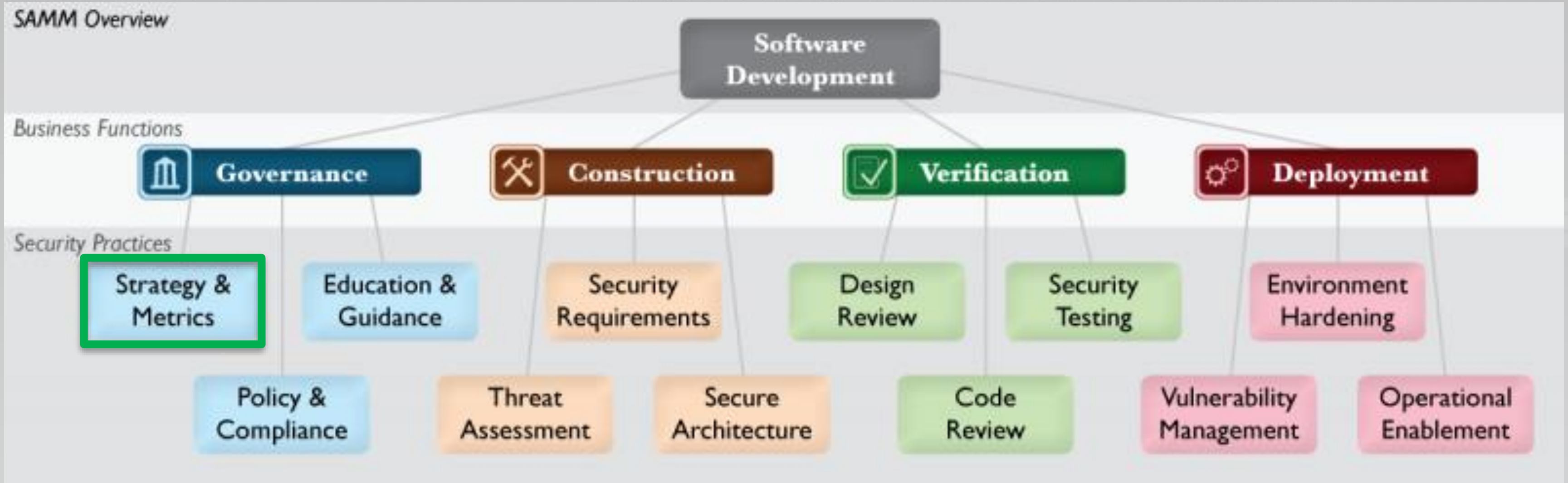
The following code example uses a `PreparedStatement`, Java's implementation of a parameterized query, to execute the same database query.

```
// This should REALLY be validated too
String custname = request.getParameter("customerName");
// Perform input validation to detect attacks
String query = "SELECT account_balance FROM user_data WHERE user_name = ? ";
PreparedStatement pstmt = connection.prepareStatement( query );
pstmt.setString( 1, custname);
ResultSet results = pstmt.executeQuery( );
```

Link: https://www.owasp.org/index.php/OWASP_Cheat_Sheet_Series

Software Assurance Maturity Model (SAMM)

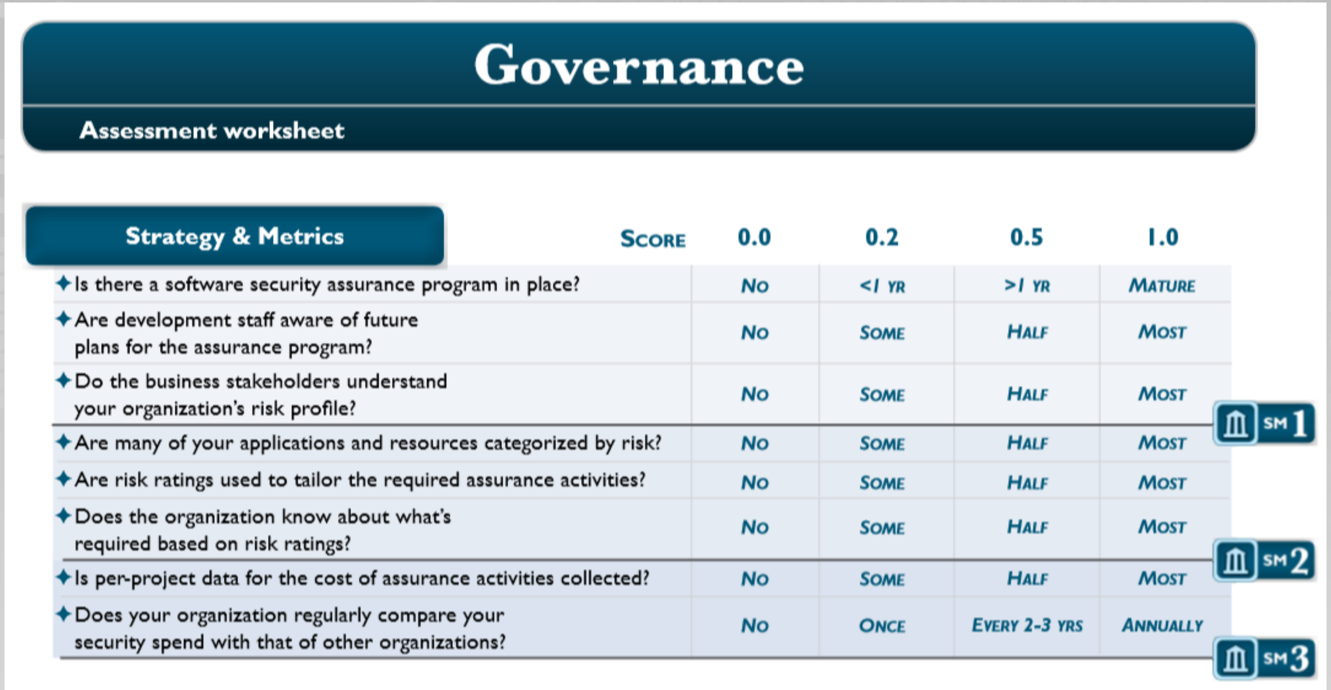
- **Sicherer Software Development Lifecycle mit OWASP SAMM:**



Link: https://www.owasp.org/index.php/OWASP_SAMM_Project

Software Assurance Maturity Model (SAMM)

- **Beispiel:**



Governance

Assessment worksheet

Strategy & Metrics	Score 0.0	0.2	0.5	1.0	
✦Is there a software security assurance program in place?	No	<1 yr	>1 yr	Mature	
✦Are development staff aware of future plans for the assurance program?	No	Some	Half	Most	
✦Do the business stakeholders understand your organization's risk profile?	No	Some	Half	Most	SM 1
✦Are many of your applications and resources categorized by risk?	No	Some	Half	Most	
✦Are risk ratings used to tailor the required assurance activities?	No	Some	Half	Most	
✦Does the organization know about what's required based on risk ratings?	No	Some	Half	Most	SM 2
✦Is per-project data for the cost of assurance activities collected?	No	Some	Half	Most	
✦Does your organization regularly compare your security spend with that of other organizations?	No	Once	Every 2-3 yrs	Annually	SM 3

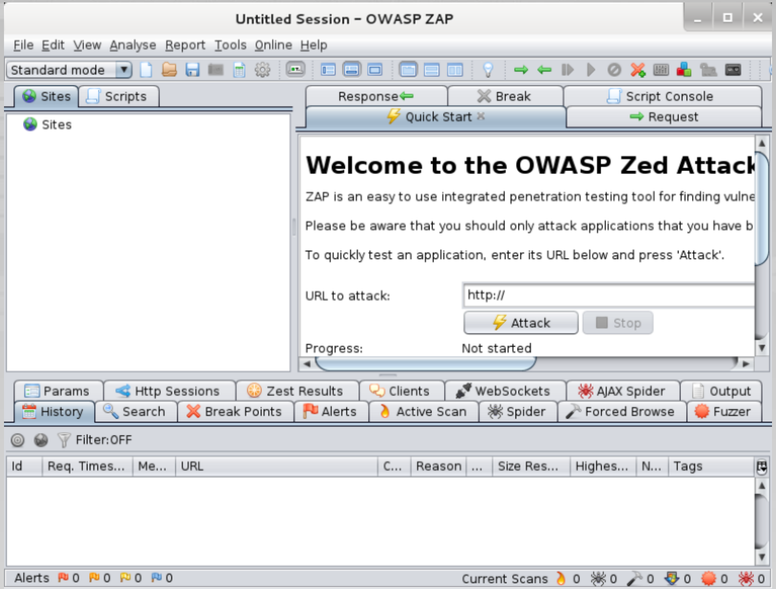
Link: https://www.owasp.org/index.php/OWASP_SAMM_Project

Werkzeuge (Beispiele)

Zed Attack Proxy (ZAP)

Schwachstellenscanner für Webanwendungen

- Man-in-the-middle Proxy
- Traditional and AJAX spiders
- Automated scanner, passive scanner
- Forced browsing, Fuzzer
- Dynamic SSL certificates
- Smartcard and Client Digital Certificates support
- Web sockets support
- Support for a wide range of scripting languages
- Authentication and session support
- Powerful REST based API
- …



Welcome to the OWASP Zed Attack

ZAP is an easy to use integrated penetration testing tool for finding vulne

Please be aware that you should only attack applications that you have b

To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://

Attack Stop

Progress: Not started

Link: https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

O-Saft: OWASP SSL Advanced Forensic Tool

Transport-Verschlüsselung mit TLS/SSL testen:

- Auch für **interne Netze** geeignet
- Tests von **beliebigen**, sogar unbekannten **Ciphern** möglich (bis 65536)
- **Unabhängig** vom **Betriebssystem** und von installierten TLS-Bibliotheken
- Http-**Proxy-Support**
- **STARTTLS**-Protokolle: SMTP, POP3, IMAP, LDAP, RDP, XMPP, eigene, ...
- **Prüfung/Hinweis** auf (mögliche) bekannte **Schwachstellen**
- Tests von **mehreren Servern** gleichzeitig, **automatisierbare** Tests
- Formatierung und Nachbearbeitung der Ergebnisse

Link: https://www.owasp.org/index.php/O-Saft

Weitere Informationen

41 OWASP-Leitlinien & Werkzeuge (‚Flagship‘ oder ‚Lab‘)

Werkzeuge:

- Zed Attack Proxy
- Web Testing Environment Project
- OWTF
- Dependency Check
- Security Shepherd
- DefectDojo Project
- Juice Shop Project
- Security Knowledge Framework
- Dependency Track Project
- O-Saft
- EnDe Project
- Mobile Security Project
- O2 Platform
- Passfault
- WebGoat Project
- Xenotix XSS Exploit Framework
- Code Pulse Project
- SeraphimDroid Project
- Glue Tool Project

Leitlinien:

- Application Security Verification Standard Project
- Software Assurance Maturity Model (SAMM)
- AppSensor Project
- Top Ten Project
- Testing Project
- Cheat Sheet Series
- Code Review Guide Project
- Cornucopia
- Podcast Project
- Proactive Controls
- Internet of Things Top Ten Project
- Top 10 Privacy Risks Project
- Snakes and Ladders Project
- Automated Threats to Web Applications
- Mobile Security Testing Guide

Code:

- ModSecurity Core Rule Set Project
- CSRFGuard Project
- Enterprise Security API
- Security Logging Project
- Benchmark

Contest:

- University Challenge
- CTF Project

Link: https://www.owasp.org/index.php/Category:OWASP_Project#tab=Project_Inventory

OWASP-Community

OWASP-Deutschland:

- German OWASP-Day (Konferenz)
- Mailing-Liste
- Stammtische:

Dresden | Frankfurt | Hamburg | Heilbronn-Franken | Karlsruhe

Köln | München | Ruhrpott | Stuttgart

Nächster Termin: Di, 25.06.2019, 19^{00} Uhr Hackerhaus, München; bitte anmelden

Links: https://www.owasp.de, https://www.owasp.org/index.php/OWASP_Chapter

Vielen Dank für Ihre Aufmerksamkeit!

Fragen?

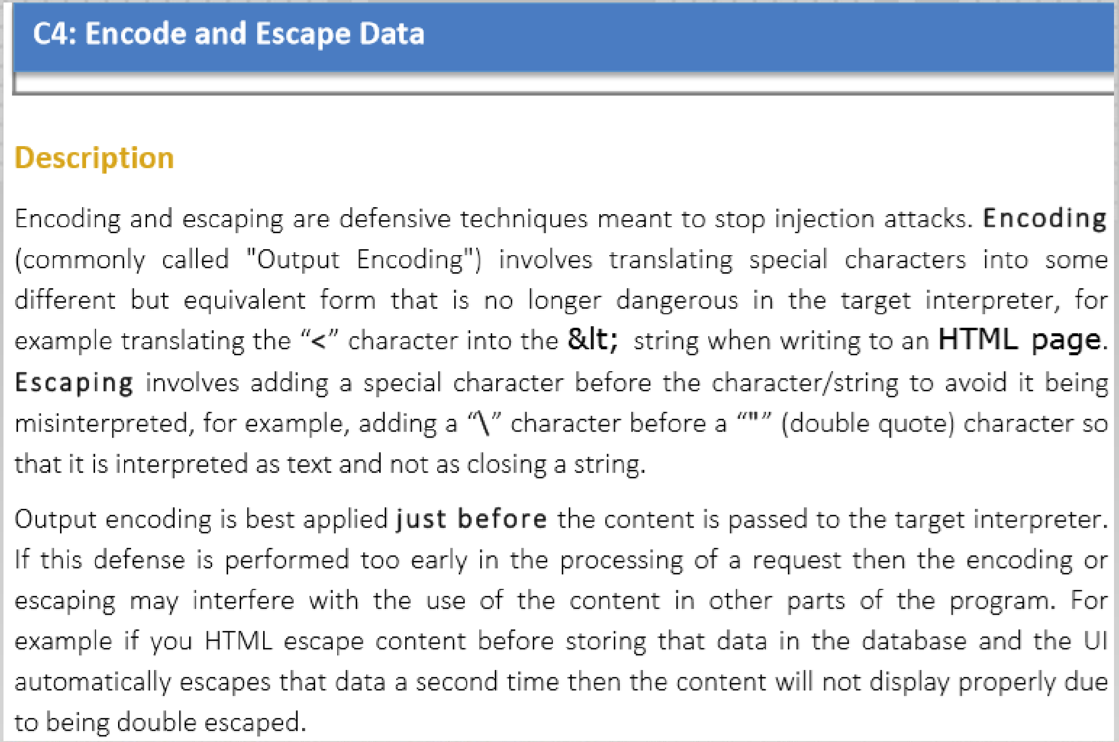
Zusätzliche Folien

OWASP Proactive Controls

- C1 Define Security Requirements
- C2 Leverage Security Frameworks and Libraries
- C3 Secure Database Access
- C4 Encode and Escape Data
- C5 Validate All Imputs
- C6 Implement Digital Identity
- C7 Enforce Access Control
- C8 Protect Data Everywhere
- C9 Implement Security Logging and Monitoring
- C10 Handle All Errors and Exceptions

Link: https://www.owasp.org/index.php/OWASP_Proactive_Controls

OWASP Proactive Controls



C4: Encode and Escape Data

Description

Encoding and escaping are defensive techniques meant to stop injection attacks. **Encoding** (commonly called "Output Encoding") involves translating special characters into some different but equivalent form that is no longer dangerous in the target interpreter, for example translating the “<” character into the &lt; string when writing to an **HTML page**. **Escaping** involves adding a special character before the character/string to avoid it being misinterpreted, for example, adding a “\” character before a “"” (double quote) character so that it is interpreted as text and not as closing a string.

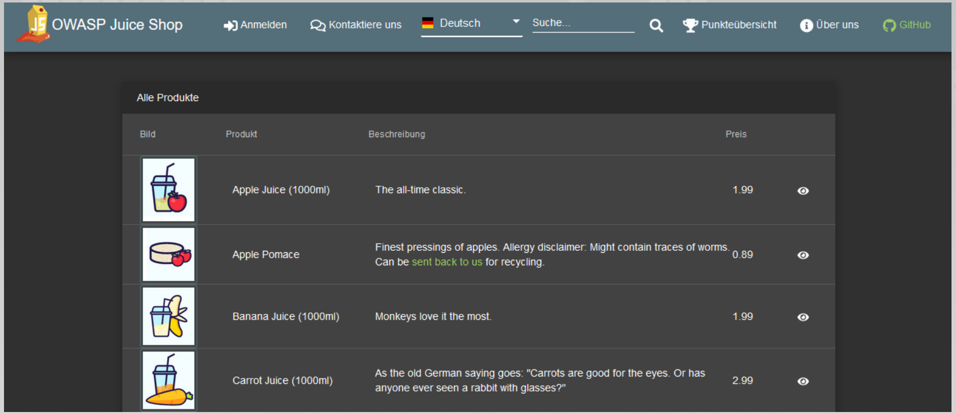
Output encoding is best applied **just before** the content is passed to the target interpreter. If this defense is performed too early in the processing of a request then the encoding or escaping may interfere with the use of the content in other parts of the program. For example if you HTML escape content before storing that data in the database and the UI automatically escapes that data a second time then the content will not display properly due to being double escaped.

(zurück zur Übersicht)

Juice Shop: Ein echter ‚Saftladen‘

Sicherheitslücken spielerisch entdecken:

- ‚Trainings-Anwendung‘ mit absichtlichen Schwachstellen
- ≥73 Übungen/Challenges
- Hacking-Wettbewerbe (Capture-the-Flag)
- Auch als Docker-Image
- Online-Demo
- Installations- und Lösungsbuch



Alle Produkte

Bild	Produkt	Beschreibung	Preis
	Apple Juice (1000ml)	The all-time classic.	1.99
	Apple Pomace	Finest pressings of apples. Allergy disclaimer: Might contain traces of worms. Can be sent back to us for recycling.	0.89
	Banana Juice (1000ml)	Monkeys love it the most.	1.99
	Carrot Juice (1000ml)	As the old German saying goes: "Carrots are good for the eyes. Or has anyone ever seen a rabbit with glasses?"	2.99

(zurück zur Übersicht)

Link: https://www.owasp.org/index.php/OWASP_Juice_Shop_Project