



OWASP

Open Web Application
Security Project

German OWASP Day 2015



Nachlese

Münchener OWASP-Stammtisch, 19.04.2016

Torsten Gigler
Thomas Herzog

Vorträge (1)



Name	Vortrag
Christian Rhino (Commerzbank)	Keynote: Muss das alles so kompliziert sein? Die Digitalisierung der Finanzindustrie zwischen Kundenerwartungen, Regulierung und FinTechs
Marcus Niemietz, Juraj Somorovsky and Christian Mainka	Not so Smart: On Smart TV Apps
Sebastian Lekies and Ben Stock	Your Scripts in My Page - What Could Possibly Go Wrong?
Michael Spreitzenborth and Jennifer Bombien	Why Organisations should rely on Mobile AppTesting
Volker Hammer	Löschen können! Die DIN 66398 und die Entwicklung von Anwendungen

Vorträge (2)



Name	Vortrag
Christoph Kern (Google)	Technical Keynote: Robuste und Praktikable Ansätze zur Verhinderung von Sicherheitsdefekten
Juraj Somorovsky	Practical Invalid Curve Attacks on TLS-ECDH
Christian Dresen and Sebastian Schinzel	Webschwachstellen im Internet of Things
Alexios Fakos	IT-Sicherheitsgesetz und mögliche Effekte für die Software-Industrie
Amir Alsbih	Praktische Erfahrungen aus der Applikationssicherheit

Lightning Talks (1)



OWASP
Open Web Application
Security Project

Name	Vortrag
Ralf Reinhardt	Ralf Reinhardt: OWASP Secure Software Contract Annex auf Deutsch
Björn Kimminich	Hacking the Juice Shop "So ein Saftladen!"
Christian Mainka, Vladislav Mladenov and Tim Guenther	EsPReSSO oder eine Erfrischung auf der Suche nach Single Sign-On
Christian Mainka and Juraj Somorovsky	How to Evaluate Web Services with WS-Attacker

Keynote: Muss das alles so kompliziert sein? (1)

[Christian Rhino]



OWASP
Open Web Application
Security Project

- Die Digitalisierung der Finanzindustrie zwischen Kundenerwartungen, Regulierung und FinTechs –

Einflüsse auf die Bank



FinTechs



Digitalisierung



Regulierung



Kunden

Keynote: Muss das alles so kompliziert sein? (2)

[Christian Rhino]



OWASP
Open Web Application
Security Project

Der Kunde erwartet ein
Maximum an Sicherheit
bei einem **Minimum an Aufwand!**



Passwort:
meineBank9999



„Der neue Safe ist die IT“
„make security suck less“

Not so Smart: On Smart TV Apps (1)

[Marcus Niemietz,
Juraj Somorovsky and Christian Mainka]



OWASP
Open Web Application
Security Project

Smart TV apps

- Social networks like Facebook
- Streaming services like Netflix or Watchever
- Games like Angry Birds
- Browsing websites



NETFLIX



Not so Smart: On Smart TV Apps (2)

[Marcus Niemietz,
Juraj Somorovsky and Christian Mainka]



OWASP
Open Web Application
Security Project

- **Attacker Model**
 - Eavesdropper → Sniffing HTTP Traffic
 - Attached Storage Attacker → e.g. USB flash drive
 - Malware Attacker → Device's app store or attacker controlled website
- **Conclusions**
 - New IoT technologies like Smart TVs are a valuable target for Attackers
- **Lessons learned**
 - Use TLS
 - Do not save sensitive data (unencrypted)
 - Do not run with root privileges
 - Use a sandboxing mechanism
 - Todo: Protect your webcam

Your Scripts in My Page (1)

– What Could Possibly Go Wrong?

[Marcus Niemietz,

Juraj Somorovsky and Christian Mainka]

- ✓ The Same-Origin Policy restricts communication of active content to objects that share the same origin

- Host
- Protocol
- Port



Your Scripts in My Page (2)

– What Could Possibly Go Wrong?

[Marcus Niemietz,

Juraj Somorovsky and Christian Mainka]

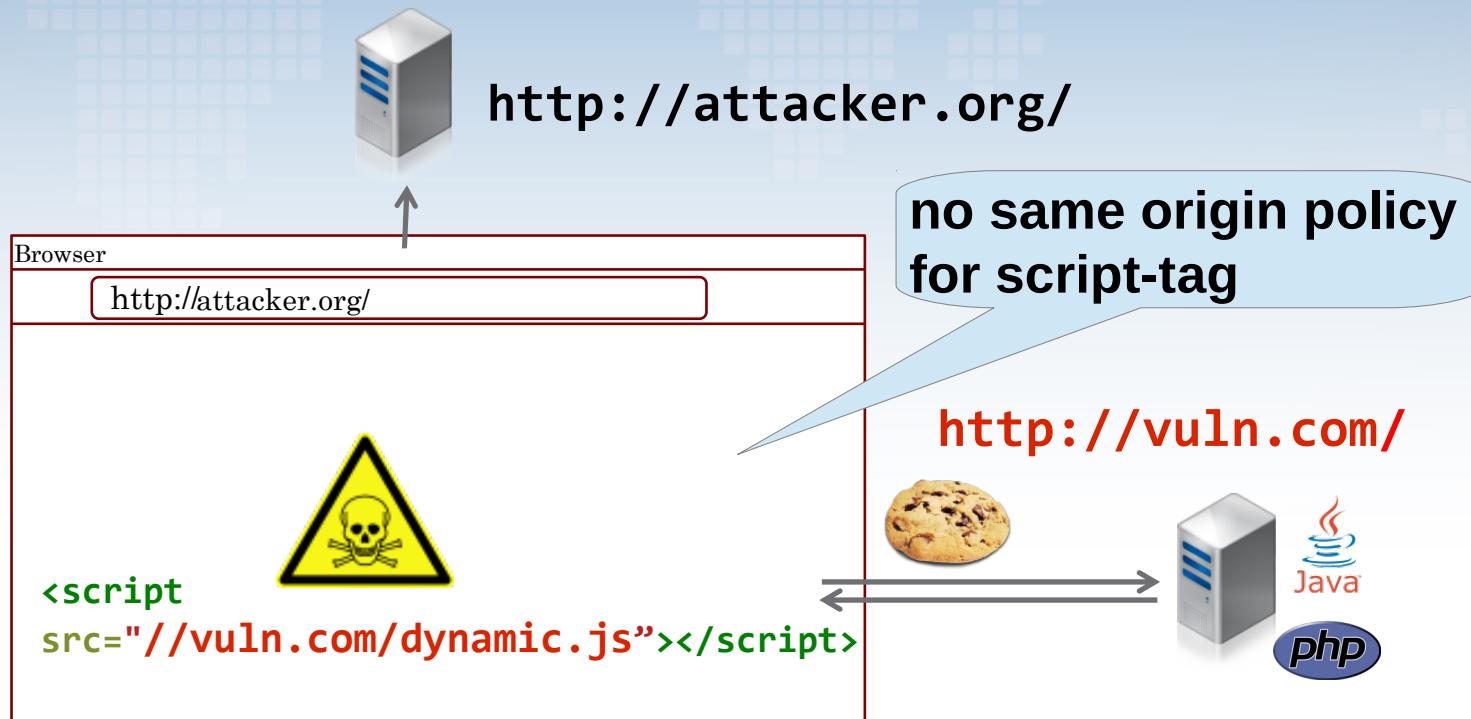


OWASP

Open Web Application
Security Project

The Unexpected Dangers of Dynamic JavaScript:

Cross-Site Script Inclusion (XSSI)



Your Scripts in My Page (3)

– What Could Possibly Go Wrong?

[Marcus Niemietz,

Juraj Somorovsky and Christian Mainka]



OWASP

Open Web Application
Security Project

- Security of dynamic Java Script files
 - Dynamic generation of Java Script is wide-spread
 - Many dynamic Java Script files include information based on a user's session, e.g. email-address
 - Data contained inside script files can be accessed across origins (e.g. if included in external scripts)
 - stored in global variables
 - via global functions
 - ... → see paper: [The Unexpected Dangers of Dynamic JavaScript](#)

=> Lösungsansätze:

- Schutz der Skript-Files vor dem Einbinden durch Dritte (z.B. sichere Token)
- Statische Java Skripte und personalisierte Datenfiles nutzen
- Importierbarkeit der Datenfiles in (externe) Skripte verhindern
 - Schutz der Datenfiles über Single-Origin-Poly
 - am Anfang der Datenfiles verhindern, dass sie als Skript ausgeführt werden können, z.B. Java-Script-Exception einbauen)

Why Organisations should rely on Mobile AppTesting (1)

Michael Spreitzenbarth
und Jennifer Bombien]



OWASP

Open Web Application
Security Project

- Beruflich genutzte Apps sollten getestet werden, wenn diese auf Firmendaten zugreifen (= speichern, bearbeiten)
- Die häufigsten Schwachstellen (2015: 180 Apps):



vortrag ist für 31.5.2016 geplant

- Insufficient Transport Layer Protection
- Lack of Binary Protections
- Insecure Data Storage
- Poor Authorization and Authentication
- Other Issues

Löschen können! (1)

– Die DIN 66398 und die
Entwicklung von Anwendungen
[Volker Hammer]



OWASP
Open Web Application
Security Project

- Beschreibt Elemente eines Löschkonzepts
- Dokumentationsstruktur
- Definiert Begriffe
- Vorgehensweise zur Bildung von Löschregeln
- Inhalt von Umsetzungsvorgaben
- Notwendige Verantwortlichkeiten

Löschen können! (2)

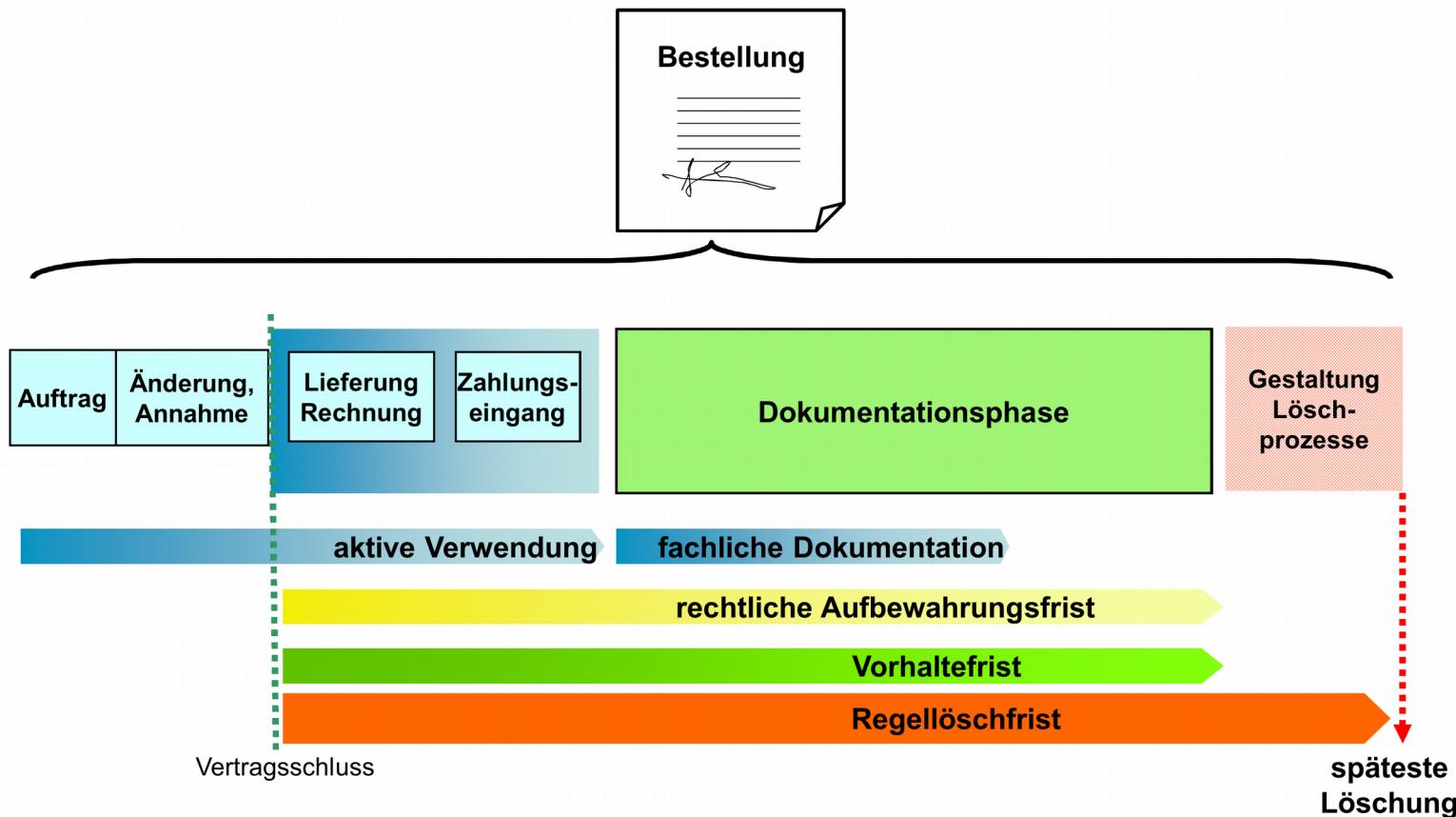
– Die DIN 66398 und die Entwicklung von Anwendungen

[Volker Hammer]



OWASP
Open Web Application
Security Project

Begriffe für die Fristableitung



Löschen können! (3)

– Die DIN 66398 und die Entwicklung von Anwendungen

[Volker Hammer]



OWASP
Open Web Application Security Project

Matrix der Löschklassen

Standardlöschfristen

		Sofort	42 T	120 T	1J	4J	7J	12J
Startzeitpunkte	Erh			Maut-daten	Mautd. mit bes. Analysebedarf			
	EdV	Web-Logs, nmF	Kurzzeit-Doku, Betriebs-Logs	Voll erstattete Reklamationen	Vorgänge ohne Dokupflicht	Rekla- und Forde-rungsd.	Handels-briefe	Buch-haltungs-daten
	EBB				ergän-zende Stamm-data		Verträge	Kern-stamm-data.

Löschklassen am Beispiel von Toll Collect

(Legende: Fett gelb unterlegt = allgemeine Gesetze, blau unterlegt = spezifische Gesetze, grün unterlegt = frei gewählt
Erh: ab Erhebung; EdV: Ende eines Vorgangs; EBB: Ende der Beziehung zum Betroffenen)

Technical Keynote: Robuste und praktikable Ansätze zur Verhinderung von Sicherheitsdefekten (1) [Christoph Kern]



Ansatz: Vermeidung von Sicherheitslücken ist Verantwortung des API-Designers, nicht des Anwendungsentwicklers

Beispiel für ein 'gehärtetes' API

- **Sicheres API mit statischem Kontrakt**

```
public class QueryBuilder {  
    private StringBuilder query;  
    /** ... Only call with compile-time-constant arg!!! ... */  
    public QueryBuilder append(@CompileTimeConstant String s) { query.append(s); }  
    public String getQuery() { return query.build(); }  
}
```

- Durchsetzen der Benutzung von statischen Werten, z.B. via javac-integrated checker
 - z.B.: [error-prone](#), vgl [Aftandilian et al, SCAM '12](#)

Technical Keynote: Robuste und praktikable Ansätze zur Verhinderung von Sicherheitsdefekten (2) [Christoph Kern]



OWASP
Open Web Application
Security Project

Beispiel für ein strukturelles Verbessern der APIs/Templates:

- “Don't Track 'Taint', make or track 'Safe'”
- Striktes Kontext-sensitives HTML Template

```
{template .profilePage autoescape="strict"}  
...  
    <div class="name">{$profile.name |escapeHtml}</div>  
    <div class="homepage">  
        <a href="{$profile.homePage |sanitizeUrl|escapeHtml}"> ...  
    <div class="about">  
        {$profile.aboutHtml |escapeHtml}  
    ...  
{/template}
```

=> Ergebnisse:

- manuelle Code-Analyse auf spezifische Module reduziert
- weniger Schwachstellen
- mehr Vertrauen in APIs

Vertiefender Artikel: [Securing the Tangled Web](#), Video: [AppSec California 2016](#)

Practical Invalid Curve Attacks on TLS-ECDH (1)

[Juraj Somorovsky]



OWASP
Open Web Application
Security Project

Vorwort: Beispiele für Elliptische Kurven

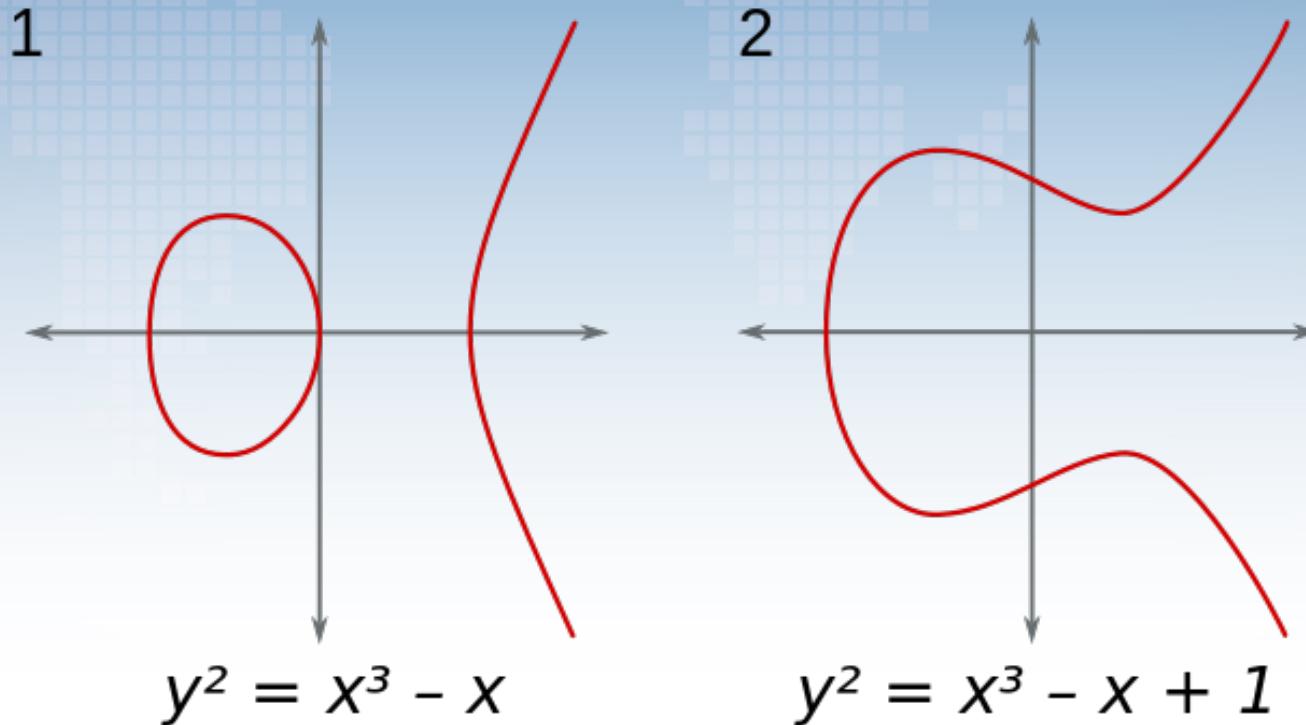


Bild-Quelle: <https://upload.wikimedia.org/wikipedia/commons/thumb/d/d0/ECCLines-3.svg/640px-ECCLines-3.svg.png>

Practical Invalid Curve Attacks on TLS-ECDH (2)

[Juraj Somorovsky]

Elliptic Curve Diffie Hellman (ECDH) → point P' outside of curve E



OWASP

Open Web Application
Security Project

RUHR-UNIVERSITÄT BOCHUM

RUB

Invalid Curve Attack

- What is the problem?
- Shared secret has only 5 possible values!

- Example

Server Secret s = 13

- Server attempts to multiply sP

$$3 = s \bmod 5$$

5P = infinity
10P = infinity

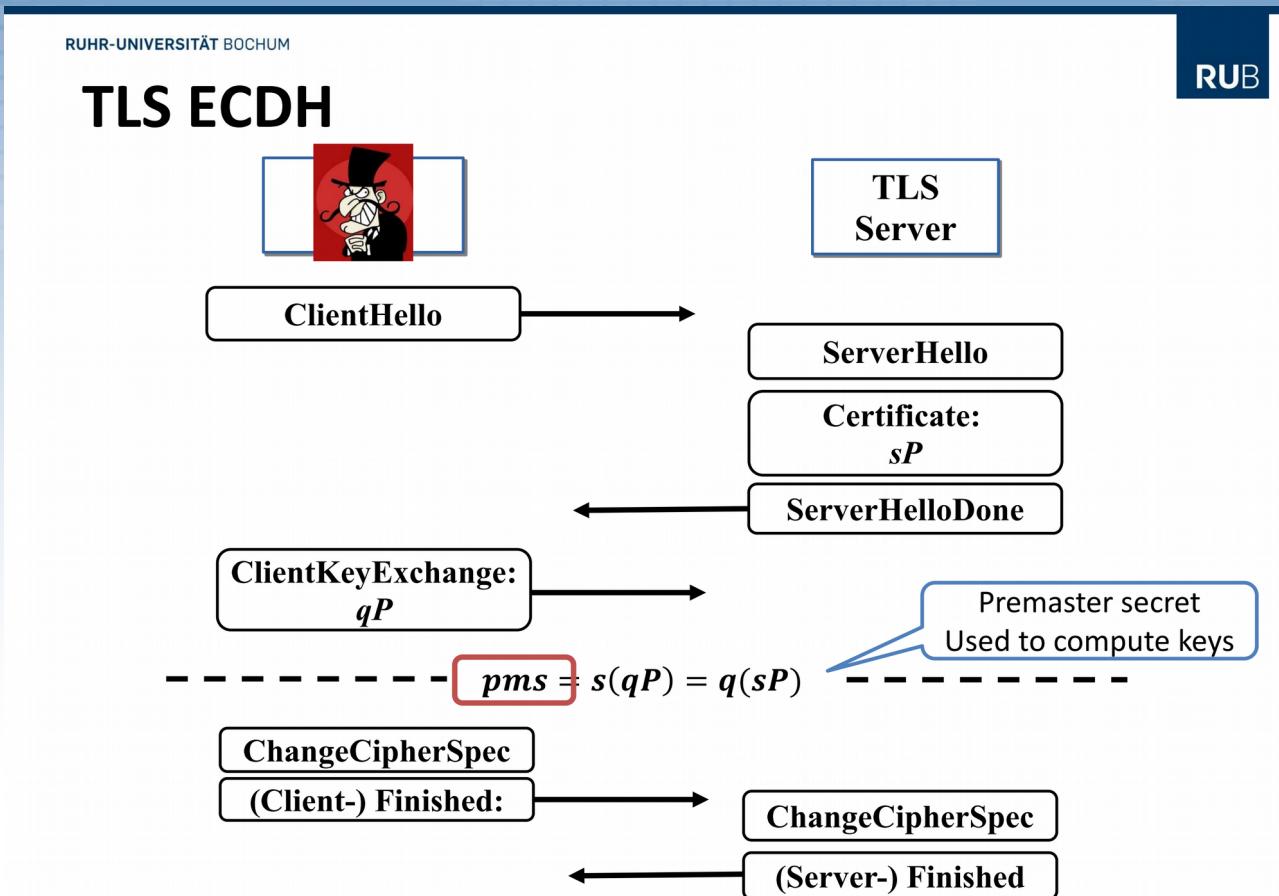


Practical Invalid Curve Attacks on TLS-ECDH (3)

[Juraj Somorovsky]



OWASP
Open Web Application Security Project



Practical Invalid Curve Attacks on TLS-ECDH (4)

[Juraj Somorovsky]



Evaluation

- 8 libraries → 2 vulnerable:
Bouncy Castle v1.50, Bouncy Castle v1.52, MatrixSSL, mbedTLS, OpenSSL, Java NSS Provider, Oracle JSSE, WolfSSL
- FIPS-zertifiziertes HSM-Modul (CVE 2015-6924)
- Attacks extract server private keys!
- Weitere Informationen:
<http://web-in-security.blogspot.de/2015/09/practical-invalid-curve-attacks.html>

Webschwachstellen im Internet of Things

[Christian Dresen,
Sebastian Schinzel]

– keine Präsentation verfügbar –



OWASP

Open Web Application
Security Project

- Verschiedenste, aus dem Internet erreichbare Geräte weisen teilweise haarsträubende Sicherheitslücken auf
 - z.B. Router, Drucker, Telefone, Kameras, Fernseher, Handys
 - Software (binäres, proprietäres Firmware-Image) ist oft über Hersteller-Webseiten zugänglich
 - Häufig Konfiguration via Webschnittstelle, die meist jegliche Best-Practices missachtet (z.B. OWASP-Richtlinien)
 - Im Vortrag wurde ein neues Werkzeug vorgestellt
 - Damit wurden 1000 Firmwares automatisiert auf Sicherheits-schwachstellen untersucht
- => gefundene, kritische (und teilweise kuriose)
Sicherheitslücken

IT-Sicherheitsgesetz und mögliche Effekte für die Software-Industrie

[Alexios Fakos]



OWASP
Open Web Application
Security Project

- keine Präsentation verfügbar –**
- **IT-Sicherheitsgesetz trifft in erster Linie Betreiber kritischer Infrastrukturen (KRITIS)**
 - Verpflichtung, ein Mindestniveau an IT-Sicherheit einzuhalten**
 - Meldung von IT-Sicherheitsvorfällen an das Bundesamt für Sicherheit in der Informationstechnik (BSI)**
- **IT-Sicherheitsgesetz betrifft jedoch auch Hard- und Software-Hersteller**
 - Mitwirkungspflicht von Software-Herstellern hinsichtlich Beseitigung von Sicherheitslücken**
 - Vorstellung denkbarer Handlungsmöglichkeiten seitens Betreiber**

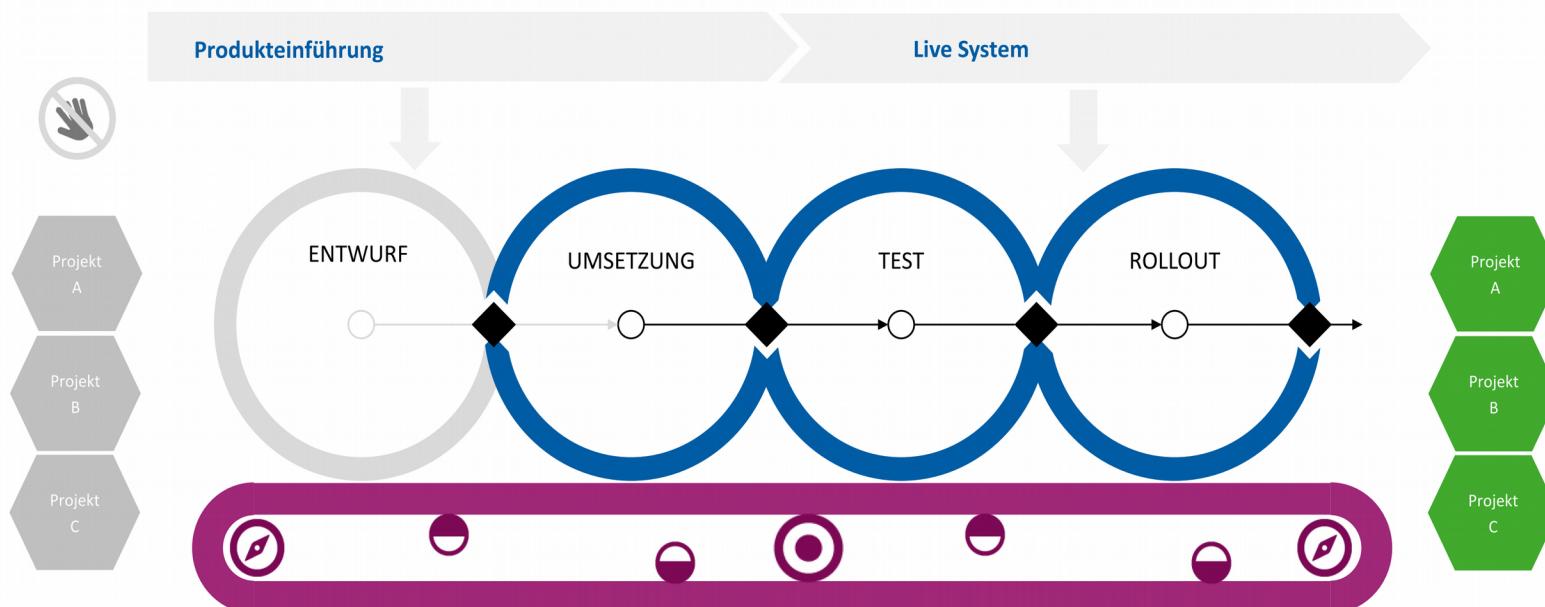
Praktische Erfahrungen aus der Applikationssicherheit (1)

[Amir Alsbih]



OWASP
Open Web Application
Security Project

Lösung DevOps – Automatisierter Prozess



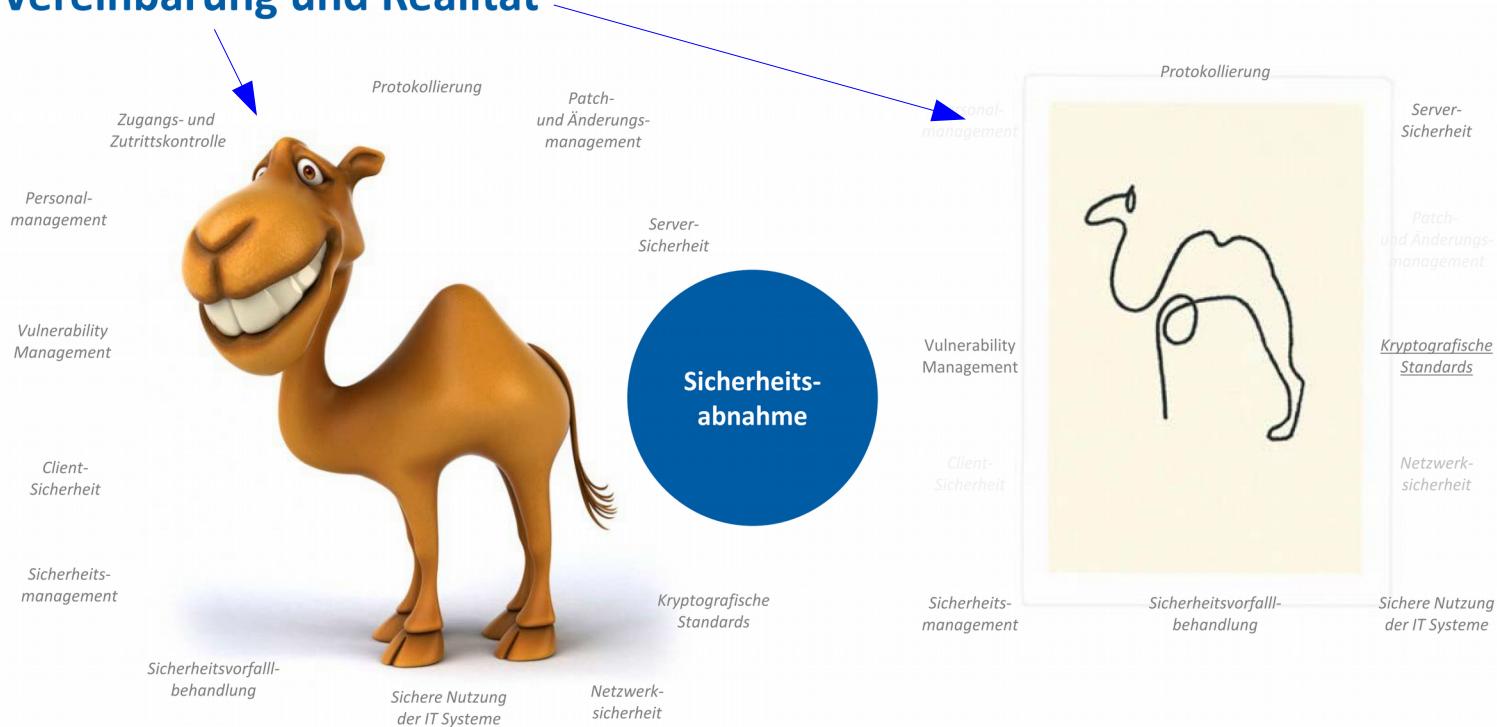
Praktische Erfahrungen aus der Applikationssicherheit (2)

[Amir Alsbih]



OWASP
Open Web Application Security Project

Erfahrung mit Dienstleistern Vereinbarung und Realität



Lightning Talks (1)



OWASP
Open Web Application
Security Project

- **OWASP Secure Software Contract Annex auf Deutsch**

https://www.owasp.org/images/b/b6/OWASP_Secure_Software_Contract_Annex_auf_Deutsch - Ralf_Reinhardt.pdf

[Ralf Reinhardt]

=> vgl Ralfs Vortrag

- **Hacking the Juice Shop "So ein Saftladen!"**

https://www.owasp.org/images/8/8d/Juice_Shop_An_intentionally_insecure_Javascript_Web_Application - Björn_Kimminich.pdf

[Björn Kimminich]

=> vgl Ralfs Vortrag

Lightning Talks (2)

EsPReSSO oder eine Erfrischung

auf der Suche nach Single Sign-On

[Christian Mainka, Vladislav Mladenov und Tim Guenther]



OWASP

Open Web Application
Security Project

Analyzing SSO: Protocols

- SAML 2.0
- OpenID
- OAuth 2.0
 - MS Account
 - Facebook Connect
- OpenID Connect 1.0
- BrowserID



- Automated analysis is insufficient (using existing tools before)
- Security report: "No issues found"? (so far unclear results)
- EsPReSSO is an extension for the Burp Suite

=> To the best of our knowledge, EsPReSSO is the first tool capable to detect, display and modify multiple different SSO protocols at the same time.

Lightning Talks (3)

How to Evaluate Web Services with WS-Attacker

[Christian Mainka und Juraj Somorovsky]



OWASP
Open Web Application
Security Project

– keine Präsentation verfügbar –

WS-Attacker is a modular framework for Web Services penetration testing. It is a free and easy to use software solution, which provides an all-in-one security checking interface with only a few clicks.

- Download: <https://github.com/RUB-NDS/WS-Attacker>
- Documentation: https://github.com/RUB-NDS/WS-Attacker/raw/master/doc/_main.pdf
- Last Changes:
 - New Intelligent Denial-of-Service Plugin
 - Fully automatic detection of DoS Weaknesses
 - Detects Thresholds
 - Improved time measurement ("last byte sent" till "first byte received")
 - New XML-Encryption Attack Plugin
 - Automatically detects XML-Encryption Attack countermeasures
 - Attacks symmetric and asymmetric XML-Encryption