Juventus Technikerschule HF

Hacking Exposed

# TLS & Internet PKI

Ahmet Inci

# Wichtiger Hinweis

- Dies ist eine Lehrveranstaltung.
- Die im Rahmen der Hacking-Exposed-Vorlesung vermittelten Kenntnisse sollen dazu beitragen, dass Sie Informationssicherheitsaspekte beachten und in Ihren Projekten berücksichtigen.
- Die HE-Vorlesung ist keineswegs als Anstiftung zum Hacken zu verstehen.

# Inhalt heute

- TLS Grundlagen, Funktionsweise und Anwendungszwecke
- Internet Public Key Infrastructure

# Ziele

- Sie kennen die grundlegende Funktionsweise von TLS und können wichtige Komponenten eines TLS Handshakes benennen.
- Sie können TLS-Scanning beschreiben und kennen die Voraussetzungen, um dieses durchzuführen.
- Sie wissen, was eine Internet PKI ist und wie eine Vertrauenskette funktioniert und können deren Funktionsweise beschreiben.
- Sie können X.509 Zertifikate interpretieren und erstellen.

# #01 Transportverschlüsselung

# Typische Transportverschlüsselungen: IPSec

1. **Verschlüsselung**: IPSec kann den Inhalt der Datenpakete verschlüsseln, so dass sie nur vom Empfänger entschlüsselt und gelesen werden können.

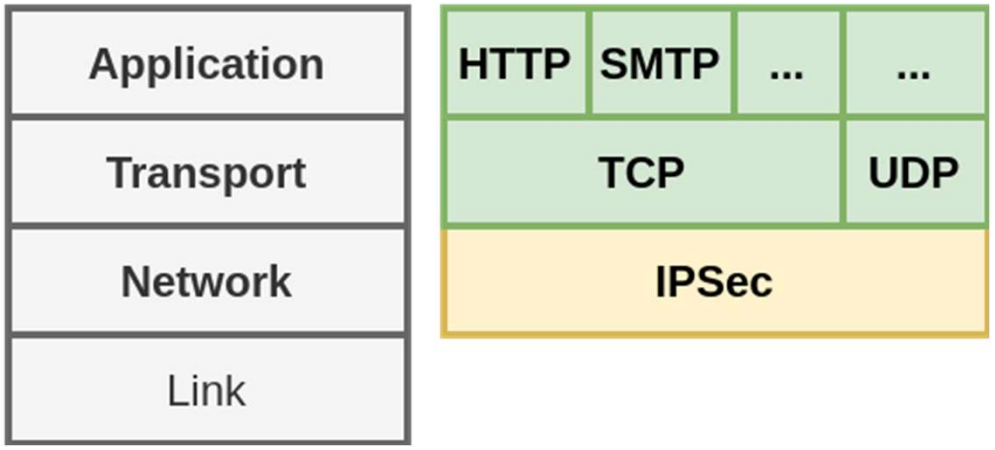
2. **Authentifizierung**: Es stellt sicher, dass die Datenpakete tatsächlich von den angegebenen Sendern stammen.

3. **Integrität**: IPSec sorgt dafür, dass die Daten während der Übertragung nicht verändert wurden. Dies geschieht durch Hash-Funktionen und digitale Signaturen.

4. **Schlüsselverwaltung**: IPSec verwaltet und tauscht Schlüssel aus, die für Verschlüsselung und Authentifizierung verwendet werden, mittels des Internet Key Exchange (IKE)-Protokolls.

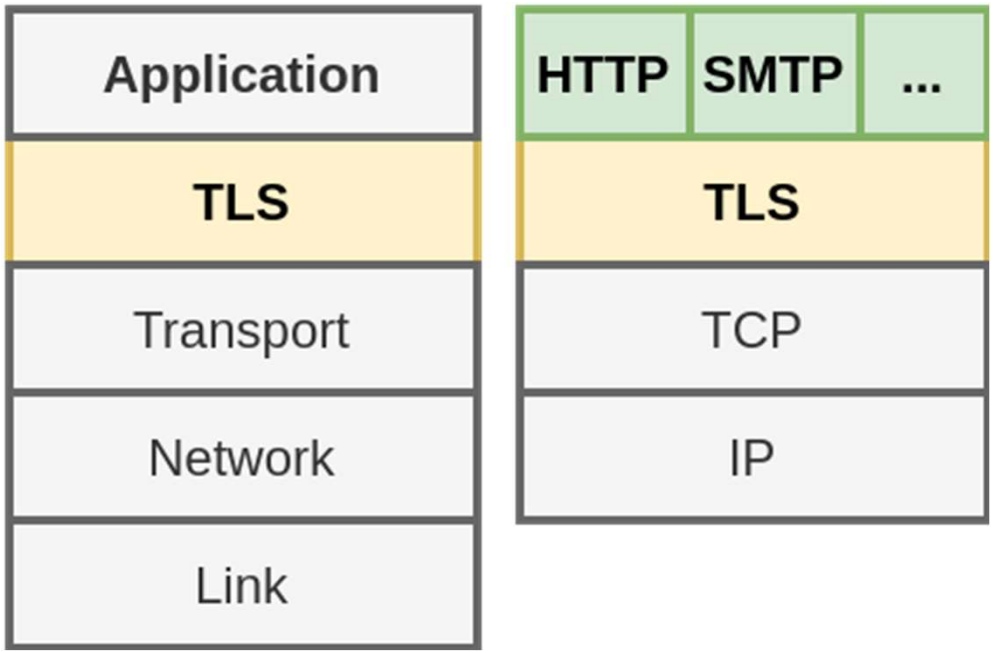
5. **Protokollmodi**: IPSec kann im Transportmodus, der nur den Payload der IP-Pakete verschlüsselt, oder im Tunnelmodus, der das gesamte IP-Paket verschlüsselt, betrieben werden.

- Side-to-Side VPN
- Road-Warrior Szenario



# Typische Transportverschlüsselungen: TLS

- Protokollschicht auf TCP
- **TLS** ist Application-Protokoll unabhängig / agnostisch (SMTPS, HTTPS). TLS wird typischerweise verwendet, um eine sichere Kommunikation zwischen zwei Endpunkten auf Anwendungsebene zu gewährleisten
- **IPSec** wird oft in Netzwerkumgebungen verwendet, um den gesamten Datenverkehr zwischen zwei Standorten zu verschlüsseln, wie z. B. zwischen zwei Zweigstellen eines Unternehmens oder zwischen einem Benutzer und einem Unternehmensnetzwerk im Falle eines VPN-Zugangs

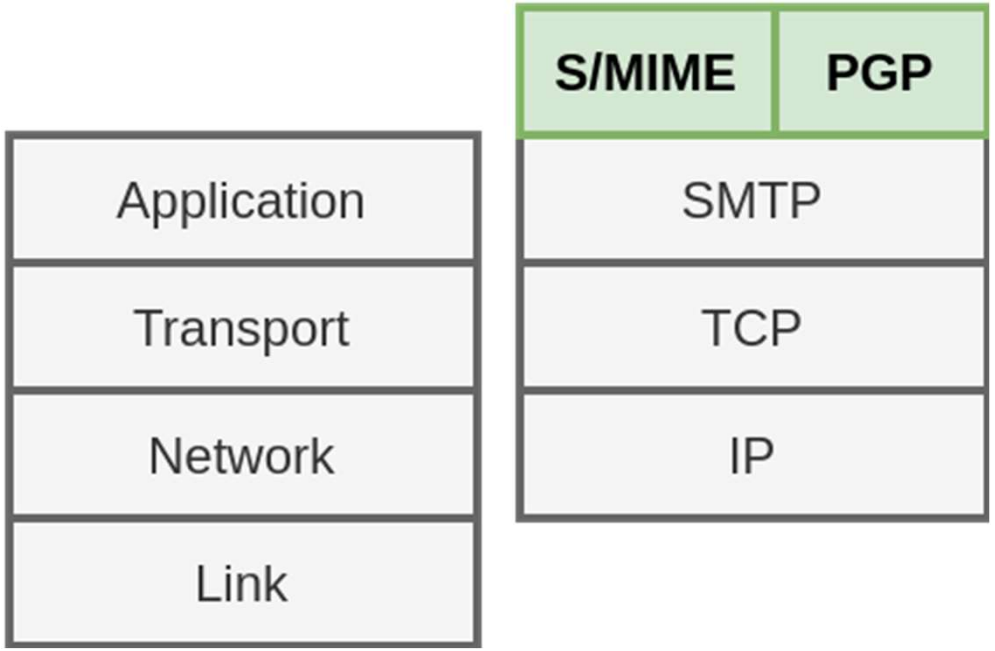


# Inhaltsverschlüsselung

1. **Inhaltsverschlüsselung**: Dies bezieht sich auf die Verschlüsselung der eigentlichen Nachrichteninhalte, nicht nur des Übertragungswegs, wie es bei TLS der Fall ist.

2. **Keine Transportverschlüsselung**: Im Gegensatz zu TLS, das eine sichere Verbindung auf Transportebene bietet, bieten S/MIME und PGP Sicherheit auf der Anwendungsebene, indem sie die Inhalte (den Payload) der E-Mail selbst verschlüsseln.

3. **E-Mail Verschlüsselung, die sich auf den Inhalt (Payload) bezieht**: S/MIME und PGP verschlüsseln den Inhalt der E-Mail-Nachrichten, um die Vertraulichkeit der Nachricht zu gewährleisten, selbst wenn sie über ungesicherte Transportwege übertragen wird.

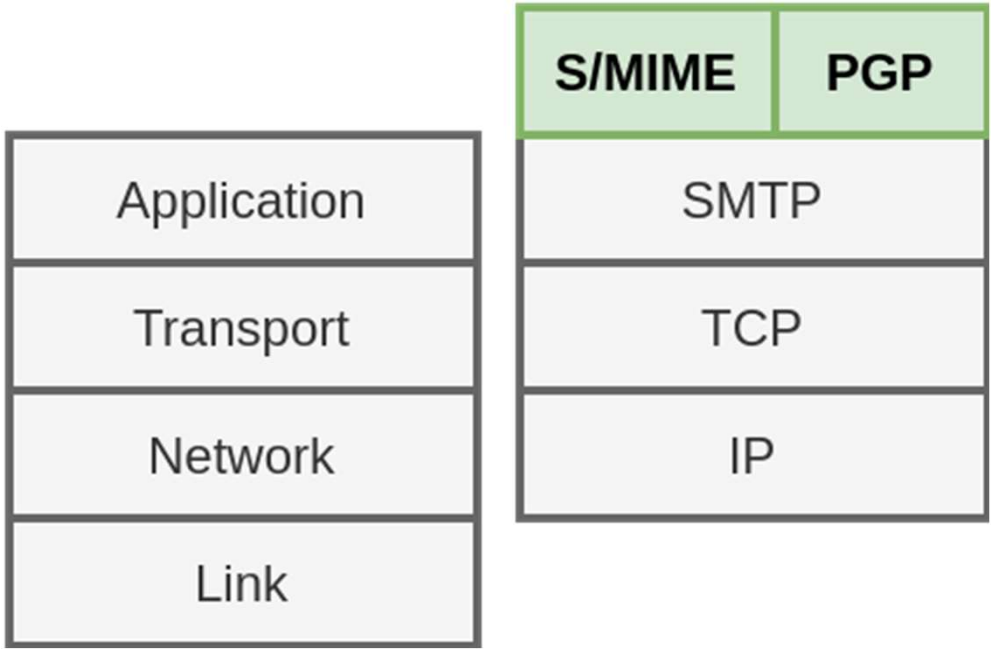


# S/MIME

- **Signierung:**

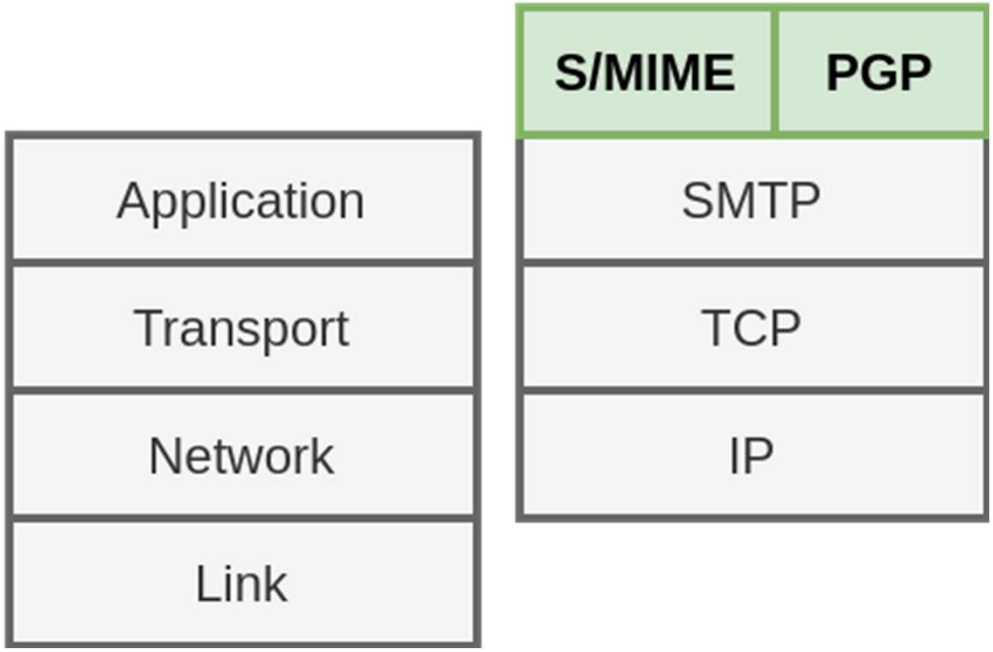
- Der **private Schlüssel** des Absenders wird verwendet, um die Nachricht zu signieren. Dies erzeugt eine digitale Signatur, die an die Nachricht angehängt wird.

- Der **öffentliche Schlüssel** des Absenders, der in seinem digitalen Zertifikat enthalten ist und von einer Zertifizierungsstelle (CA) ausgestellt und signiert wurde, wird vom Empfänger verwendet, um die Signatur zu überprüfen.



# S/MIME

- **Verschlüsselung:**
- Der **öffentliche Schlüssel** des Empfängers, der in seinem digitalen Zertifikat enthalten ist, wird vom Absender verwendet, um die Nachricht zu verschlüsseln.
- Nur der **private Schlüssel** des Empfängers kann die verschlüsselte Nachricht entschlüsseln. Da nur der Empfänger Zugang zu seinem privaten Schlüssel hat, kann auch nur er die Nachricht lesen.



# #02 Transport Layer Security

# Kommunikationskanal

- Transportverschlüsselung bietet einen verschlüsselten Kommunikationskanal über ein nicht vertrauenswürdiges Netz.

| OSI Schicht | Protokoll | Anwendung |
|---|---|---|
| Anwendung (Schicht 7) | SSH | Remote-Server Administration |
| Transport (Schicht 4) | TLS | Verschlüsselungsplattform für Anwendungsprotokolle |
| Internet (Schicht 3) | IPSec | VPN Verbindung |

# Transport Layer Security (TLS)

- Verschlüsselungsplattform für Protokolle der Anwendungsschicht
  - HTTPS, DoH, DoT, SMTPS, IMAPS, POP3S, XMPPS, IRCS, FTPS, EAP-TLS, OpenVPN

| OSI-Schicht | Protokoll |
|---|---|
| Anwendung | HTTP, DNS, SMTP, IMAP, POP3, XMPP, IRC, FTP, EAP, OpenVPN |
| Transport | TLS |
| | TCP |

# TLS Funktionalität

**Schlüsselaustausch:**

1. Etabliert einen sicheren Kanal für den Austausch von Schlüsseln, die für die Verschlüsselung und Entschlüsselung der übertragenen Daten verwendet werden.
2. Beinhaltet Algorithmen wie Diffie-Hellman (DH) oder Elliptic Curve Diffie-Hellman (ECDH), die es zwei Parteien ermöglichen, einen gemeinsamen Geheimschlüssel zu generieren, ohne diesen über das Netzwerk zu übertragen.

**Digitale Signatur und Zertifikatauthentifizierung:**

1. Verifiziert die Identität des Senders und Empfängers, meistens des Servers und optional des Clients, mittels digitaler Zertifikate, die von einer Zertifizierungsstelle ausgestellt sind.
2. Nutzt Signaturalgorithmen wie RSA, DSA (Digital Signature Algorithm) oder ECDSA (Elliptic Curve Digital Signature Algorithm).

**Verschlüsselung:**

1. Schützt die Daten während der Übertragung vor dem Zugriff Dritter durch Verschlüsselung.
2. Setzt symmetrische Verschlüsselungsalgorithmen wie AES (Advanced Encryption Standard) oder ChaCha20 ein, um die Daten zu verschlüsseln und zu entschlüsseln.

# TLS Funktionalität

**Integritätssicherung:**

1. Stellt sicher, dass die Daten während der Übertragung nicht verändert wurden.
2. Verwendet Hash-Funktionen wie SHA-256 in Verbindung mit einem MAC (Message Authentication Code), um die Integrität der Nachrichten zu gewährleisten.

**Forward Secrecy (FS):**

1. Gewährleistet, dass die Entschlüsselung vergangener Sitzungen nicht möglich ist, selbst wenn der private Schlüssel in der Zukunft kompromittiert wird.
2. Wird erreicht durch den Einsatz von ephemeren Schlüsseln, die für jede Sitzung neu generiert werden und nicht aus früheren Verbindungen abgeleitet werden können.

**Sitzungsmanagement:**

1. Beinhaltet das Aushandeln von Sitzungsschlüsseln, die Wiederverwendung von Sitzungen, um die Verbindungsgeschwindigkeit zu verbessern, und das ordnungsgemäße Schließen von Sitzungen.

**Versionierung und Kompatibilität:**

1. Unterstützt verschiedene Versionen des TLS-Protokolls und gewährleistet so eine Kompatibilität mit einer Vielzahl von Geräten und Software.

# Ziele von TLS

- Primäre Ziel ist die Bereitstellung eines sicheren Kanals zwischen zwei kommunizierenden Peers.
- Kryptographische Stärke
  - Starke verschlüsselte Verbindung zwischen zwei Kommunikationspunkten
- Interoperabilität
  - Unabhängige Entwickler entwickeln Programme und Bibliotheken die miteinander verschlüsselt kommunizieren können
- Erweiterbarkeit
  - Unabhängig von spezifischen kryptographischen Primitiven (z.B. Cipher oder Hashing-Funktion). Parameter können verändert werden ohne neues Protokoll erstellen zu müssen
- Effizienz
  - Kostenintensive kryptographische Operationen werden minimiert, u.a. dank Sessions

# TLS Versionen

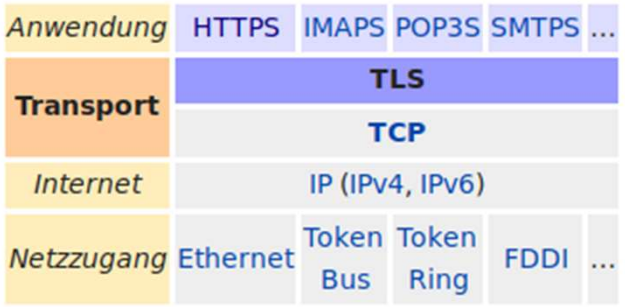
| Version | Erscheinungsjahr |
|---|---|
| SSL 1.0 | 1994 |
| SSL 2.0 | 1995 |
| SSL 3.0 | 1996 |
| TLS 1.0 | 1999 |
| TLS 1.1 | 2006 |
| TLS 1.2 | 2008 |
| TLS 1.3 | 2018 |

# Hauptunterschiede von TLS 1.2 zu TLS 1.3

- Alte symmetrische Verschlüsselungsalgorithmen entfernt
- Key Exchange und Authentication nicht mehr Teil der Cipher Suite
- Zero Round-Trip Time (0-RTT)
- Keine statische RSA und DH Cipher Suites → PK basierter Schlüsselaustausch mit Forward Secrecy
- Handshake Protokoll ab ServerHello verschlüsselt
- Einige weitere, siehe RFC 8446

# TLS Übersicht

- Grundlage moderner Verschlüsselung bspw. HTTPS, IMAPS, SMTPS etc.
- TLS Versionen 1.2 und 1.3




- SSL sowie TLS 1.0 und TLS 1.1 sind nicht zu verwenden
- Besteht aus den vier Protokollen

- Record Protocol, Es stellt die grundlegende Transportebene dar, die für die Fragmentierung, Kompression, Verschlüsselung und den Schutz der Integrität der übertragenen Daten verantwortlich ist.

- Handshake Protocol, Das Handshake Protocol wird verwendet, um eine sichere Verbindung zwischen dem Client und dem Server zu initiieren. Es handelt sich um eine Reihe von Nachrichten, die zwischen den beiden Parteien ausgetauscht werden, um die Protokollversion, die Auswahl der Cipher Suites, den Austausch der Schlüsselinformationen und die gegenseitige Authentifizierung zu verhandeln.

- Application Data Protocol, Seine Rolle besteht darin, die Anwendungsdaten für die Übertragung über das Record Protocol vorzubereiten, indem es sie verschlüsselt und sicherstellt, dass sie korrekt übertragen werden.

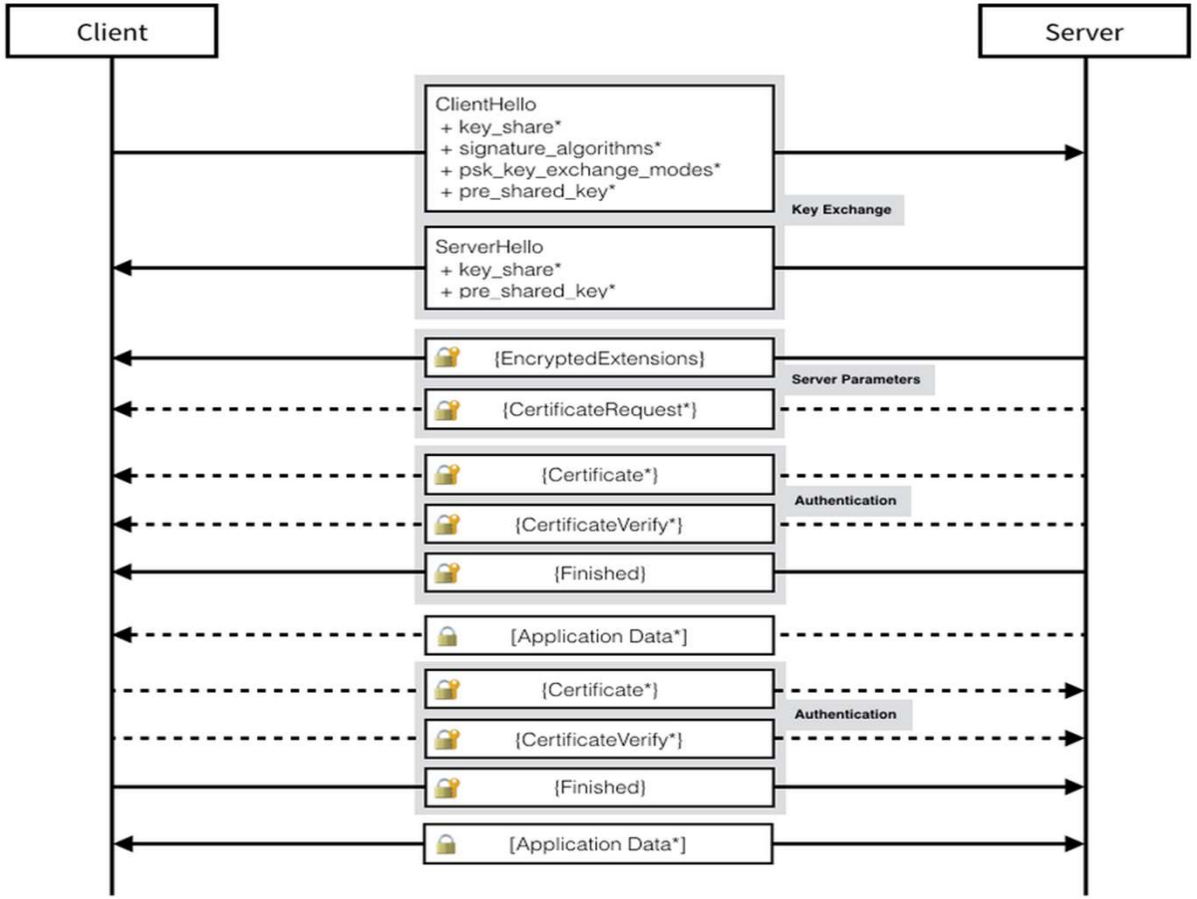
- Alert Protocol, Das Alert Protocol wird verwendet, um Alarmmeldungen zu senden, die Fehler anzeigen oder andere wichtige Zustände.

- Interessante Historie: https://www.feistyduck.com/ssl-tls-and-pki-history/

# TLS Handshake Protokoll

- Aufbau einer verschlüsselten Verbindung vom Client zum Server
- Handelt Parameter zwischen Client und Server aus:
  - TLS Version, kryptographische Parameter, Erweiterungen und Features etc.
- Ist der interessanteste Teil!

# TLS 1.3 Handshake

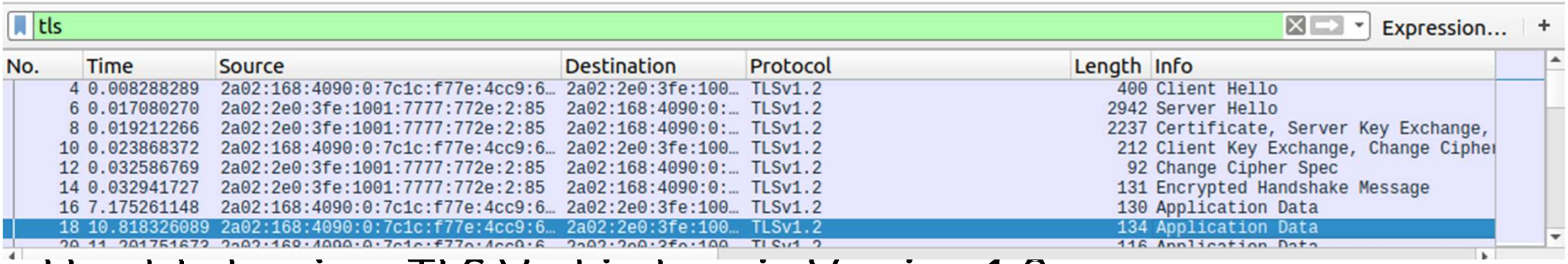




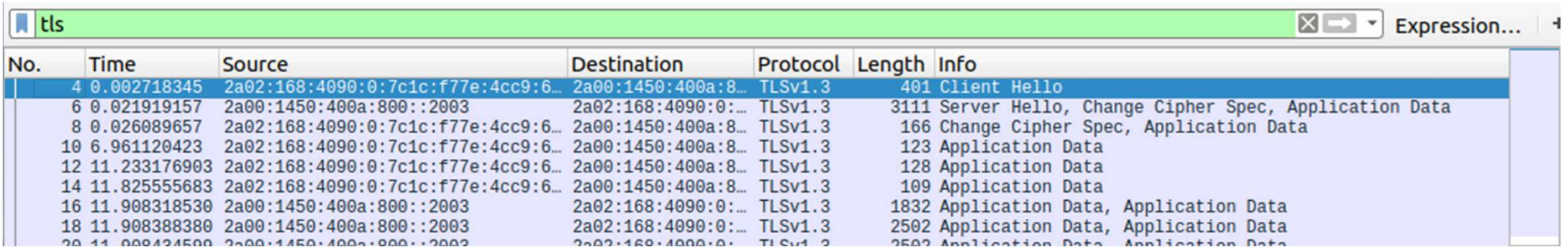

https://www.davidwong.fr/tls13/

# TLS Handshake Ablauf in Praxis

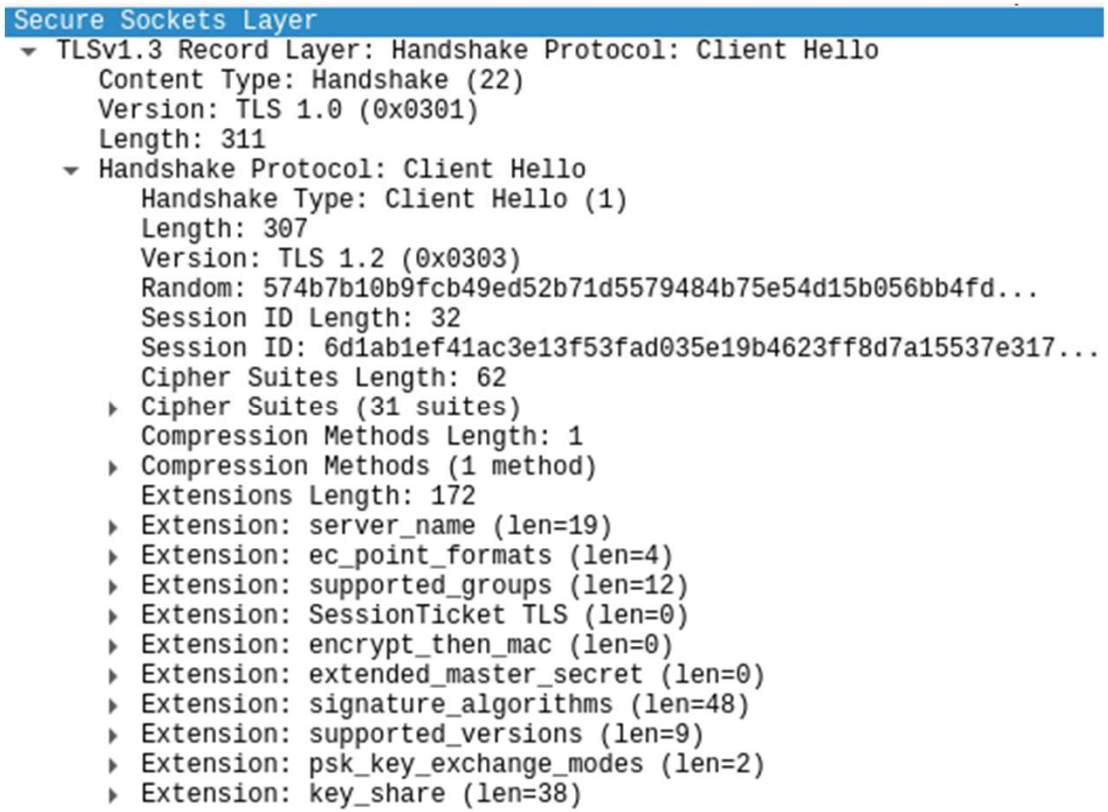
- Handshake einer TLS Verbindung in Version 1.2




- Handshake einer TLS Verbindung in Version 1.3



# TLS 1.3 Client Hello

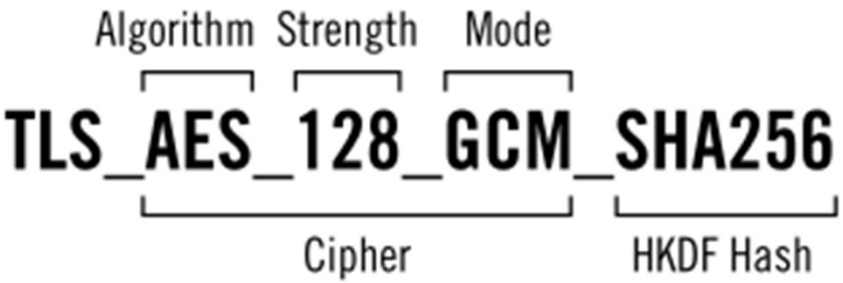


# TLS 1.3 Server Hello

```
Secure Sockets Layer
▾ TLSv1.3 Record Layer: Handshake Protocol: Server Hello
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 122
  ▾ Handshake Protocol: Server Hello
      Handshake Type: Server Hello (2)
      Length: 118
      Version: TLS 1.2 (0x0303)
      Random: 9ffc61d49e4ebcd0b400fcfd15cf5698420b4d948b188d0a...
      Session ID Length: 32
      Session ID: 6d1ab1ef41ac3e13f53fad035e19b4623ff8d7a15537e317...
      Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
      Compression Method: null (0)
      Extensions Length: 46
    ▸ Extension: key_share (len=36)
    ▸ Extension: supported_versions (len=2)
▾ TLSv1.3 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
    Content Type: Change Cipher Spec (20)
    Version: TLS 1.2 (0x0303)
    Length: 1
    Change Cipher Spec Message
```

# Cipher Suite in TLS 1.2 und frühere

# Cipher Suite ab TLS 1.3

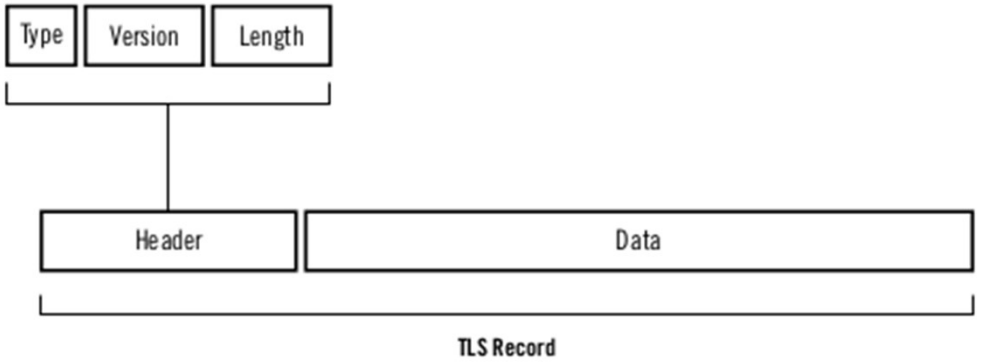


# Cipher Suite in TLS 1.3

- Nur noch Cipher und Hash-Algorithmus
  - TLS_AES_128_GCM_SHA256
  - TLS_AES_256_GCM_SHA384
  - TLS_CHACHA20_POLY1305_SHA256
  - TLS_AES_128_CCM_SHA256
  - TLS_AES_128_CCM_8_SHA256
- Key Exchange und Authentisierung ist in Extensions ausgelagert

# Record-, Application Data- & Alert Protocol

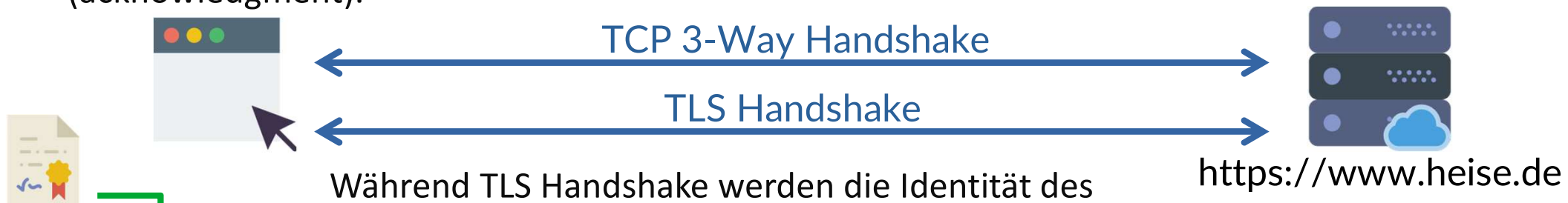
- Record Protocol ist der Rahmen der anderen drei Protokolle
  - «Containerschiff»




- Application Data Protocol beinhaltet verschlüsselte Daten
- Alert Protocol meldet Fehler
  - **fatal** → führt zum sofortigen Verbindungsabbruch bspw. **Decryption failed**
  - **warning** → kann Gegenstelle über Ereignis informieren bspw. **Unsupported extension**

# TLS und X.509 Zertifikate

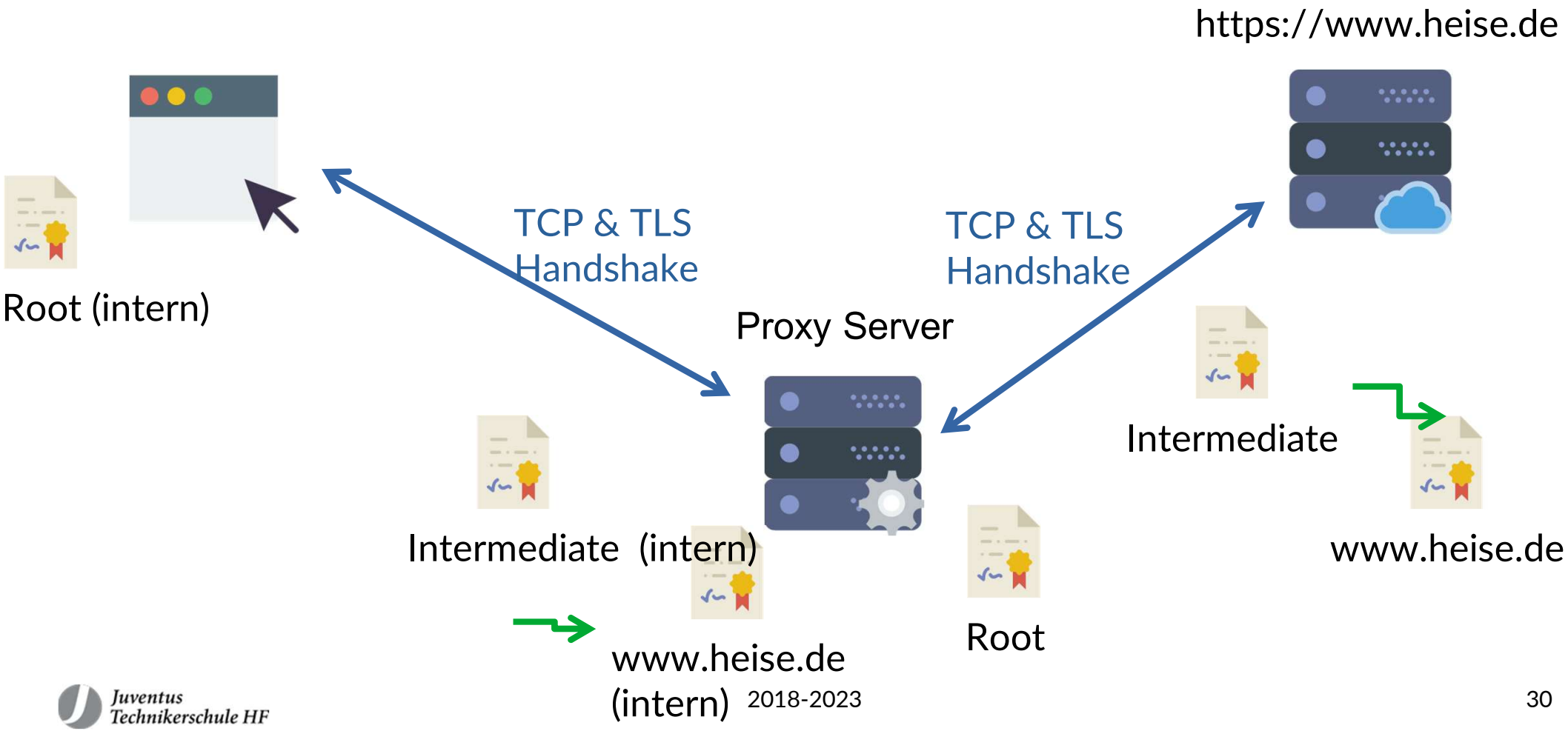
Der 3-Way Handshake ist Teil des TCP (Transmission Control Protocol) und besteht aus drei Schritten: SYN (synchronize), SYN-ACK (synchronize acknowledgment), und ACK (acknowledgment).




**Root**: Das Root-Zertifikat, das von einer Root-Zertifizierungsstelle (CA) ausgestellt wurde und im Zertifikatsspeicher des Clients als vertrauenswürdig hinterlegt ist.

**Intermediate**: Ein Zwischenzertifikat von einer Zwischen-CA, das die Verbindungskette vom Root-Zertifikat zum Serverzertifikat herstellt.

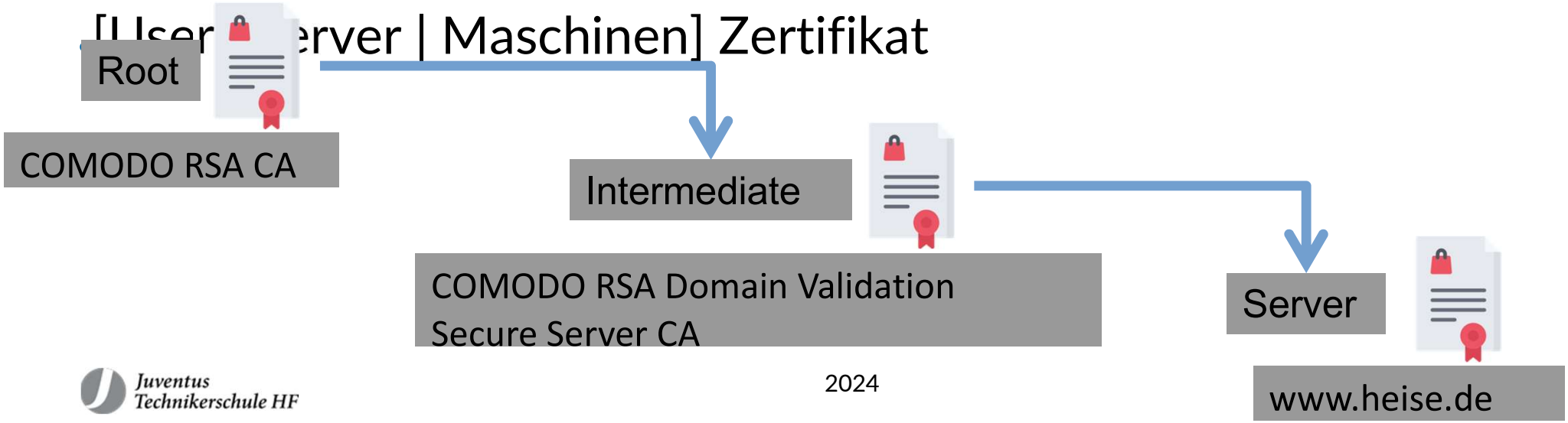
# TLS-Scanning



# #03 Internet PKI

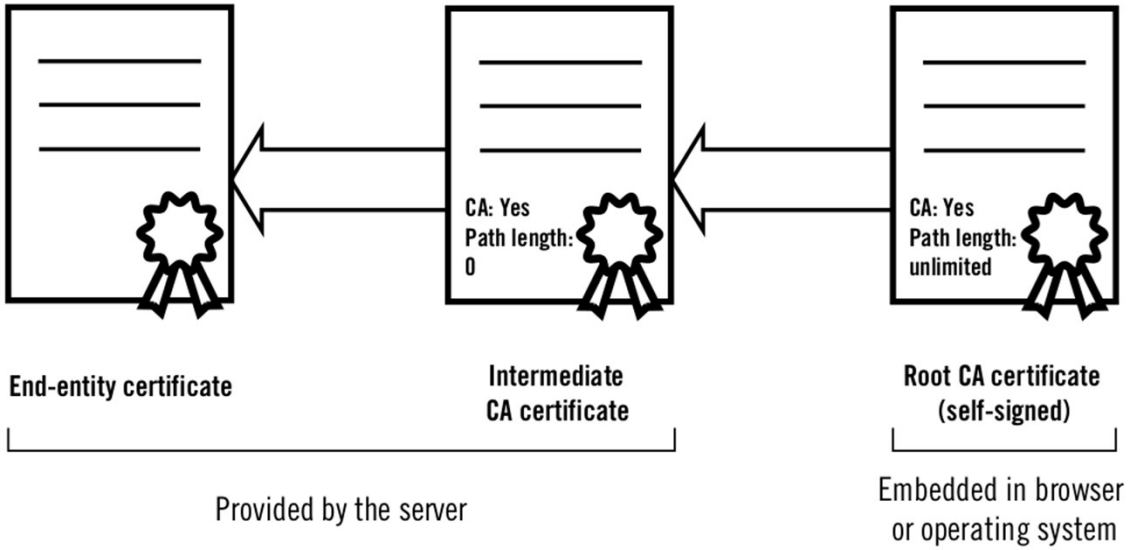
# Chain of Trust

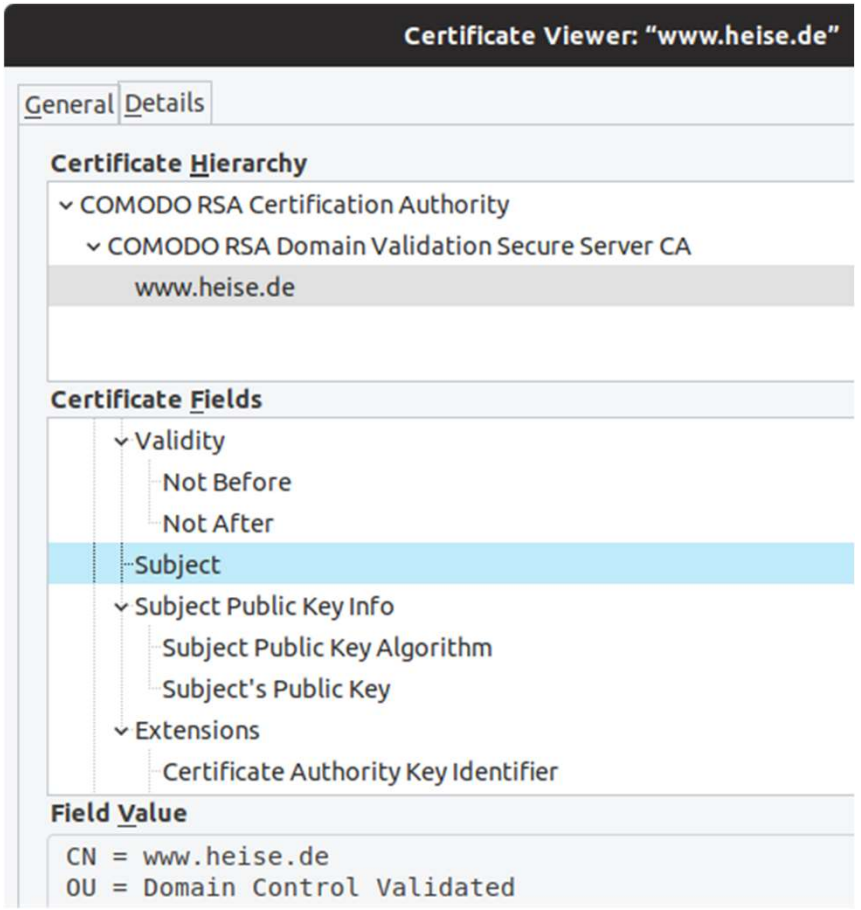
- Root Zertifikat ausgestellt von einer Certification Authority (CA)
- Intermediate Zertifikat 1 ausgestellt ebenfalls von der CA
- Intermediate Zertifikat n ausgestellt ebenfalls von der CA
- [User | Server | Maschinen] Zertifikat



# Chain of Trust im Browser

- Am Beispiel von www.heise.de



# Certification Authorities (CA)

- Müssen strenge Policy befolgen
- Haben Root Zertifikate in Trusted Store von Browsern und OS
- Können Zertifikate für alle(!) Domains erstellen

| Rank | Issuer | Usage | Market share |
|---|---|---|---|
| 1 | IdenTrust | 20.4% | 39.7% |
| 2 | Comodo | 17.9% | 34.9% |
| 3 | DigiCert | 6.3% | 12.3% |
| 4 | GoDaddy | 3.7% | 7.2% |
| 5 | GlobalSign | 1.8% | 3.5% |
| 7 | Certum | 0.4% | 0.7% |
| 8 | Actalis | 0.2% | 0.3% |
| 9 | Entrust | 0.2% | 0.3% |
| 9 | Secom | 0.1% | 0.3% |
| 10 | Let's Encrypt | 0.1% | 0.2% |
| 11 | Trustwave | 0.1% | 0.1% |
| 12 | WISeKey Group | < 0.1% | 0.1% |
| 13 | StartCom | < 0.1% | 0.1% |
| 14 | Network Solutions | < 0.1% | 0.1% |

https://en.wikipedia.org/wiki/Certificate_authority Mai 2018

# Let’s Encrypt

- Eine Initiative von Mozilla und der Electronic Frontier Foundation (EFF)
- Start 2014, bis jetzt bereits über 200 Millionen aktive Zertifikate ausgestellt
- Sind gratis und jeweils nur 90 Tage gültig
- Verlängerung einfach möglich

| Date | Certificates issued |
|---|---|
| March 8, 2016 | 1 million[49] |
| April 21, 2016 | 2 million[50] |
| June 3, 2016 | 4 million[51] |
| June 22, 2016 | 5 million[52] |
| September 9, 2016 | 10 million[53] |
| November 27, 2016 | 20 million[54] |
| December 12, 2016 | 24 million[55] |
| June 28, 2017 | 100 million[56] |
| August 6, 2018 | 115 million[57] |
| September 14, 2018 | 380 million[58] |
| October 24, 2019 | 837 million[59] |
| February 27, 2020 | 1 billion[60] |

Mai 2021 158 Millionen aktive Zertifikate

https://en.wikipedia.org/wiki/Let%27s_Encrypt

# Certificate Trust Store

- Root Zertifikaten muss vertraut werden

  Kommen in OS, Browsern und weiterer SW «vorinstalliert» mit

- Apple

  IOS und OS X Plattform

- Chrome

  Vertraut dem OS Root Store und führt zusätzliche Sicherheitsmechanismen ein

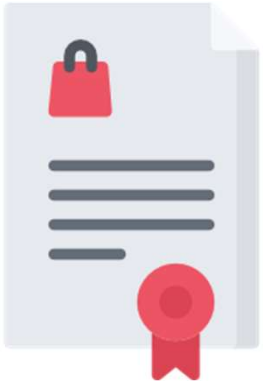
- Microsoft

  MS Plattform und Produkte

- Mozilla

# Der Standard X.509

- Wichtige Felder eines v3 X.509 Zertifikates: www.heise.de



Version: 3

Serial Number: 16:96:80:B7:7D:03:78:36:...

Signature Algorithm: SHA256

Issuer: COMODO RSA Domain Validation Secure Server CA

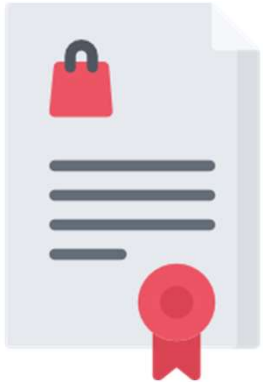
Validity: 8.1.2018 bis 8.4.2020

Subject: www.heise.de

Public Key: RSA 2048 Bit

# X.509 Zertifikat v3 Extensions

- Wichtige Extension Felder eines v3 Zertifikates



Subject Alternative Name: www.heise.de und heise.de

Key Usage: Signing & Key Encipherment

Extended Key Usage: wie Key Usage

Signature Algorithm: SHA256

CRL Distribution Points: http://crl.comodoca.com/COMODORSADomainValic

# Zertifikatspaar erstellen

- Ein X.509 Zertifikat besteht immer aus einem Zertifikat und einem Private Key

  - Zertifikat = Meta Informationen + Public Key
- OpenSSL erlaubt die einfache Erzeugung von Zertifikaten:

- $ openssl req -x509 -newkey rsa:2048 -keyout key.pem -out cert.pem -days 30 -nodes

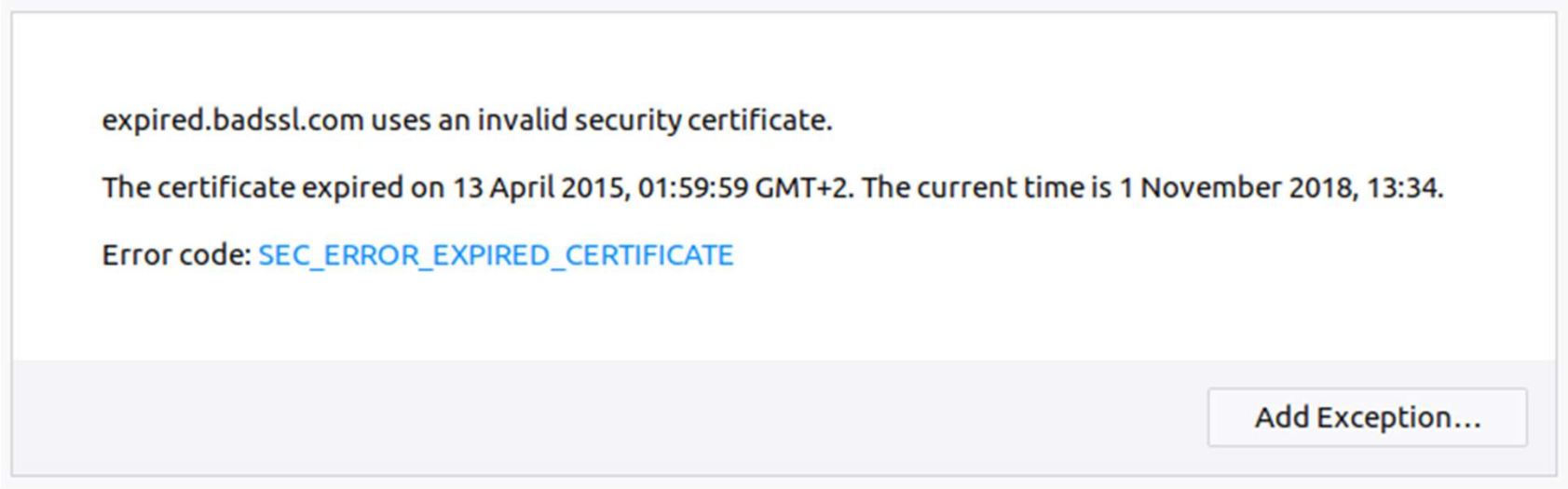
# Zertifikat Lebenszyklus

- Wenn Sie ein offizielles Webserver Zertifikat benötigen:

  - Erstellen eines Certificate Signing Request (CSR)
  - Private Key bleibt immer bei Ihnen
  - openssl req -nodes -new -newkey rsa:2048 -sha256 -out csr.pem
  - CSR an CA senden → zwecks Unterschrift
  - CA validiert die Anfrage
  - Erfolgreich: Sendet Zertifikat zurück
  - Zertifikat mit Private Key einsatzbereit
  - Bis Ablauf oder Zurückziehung

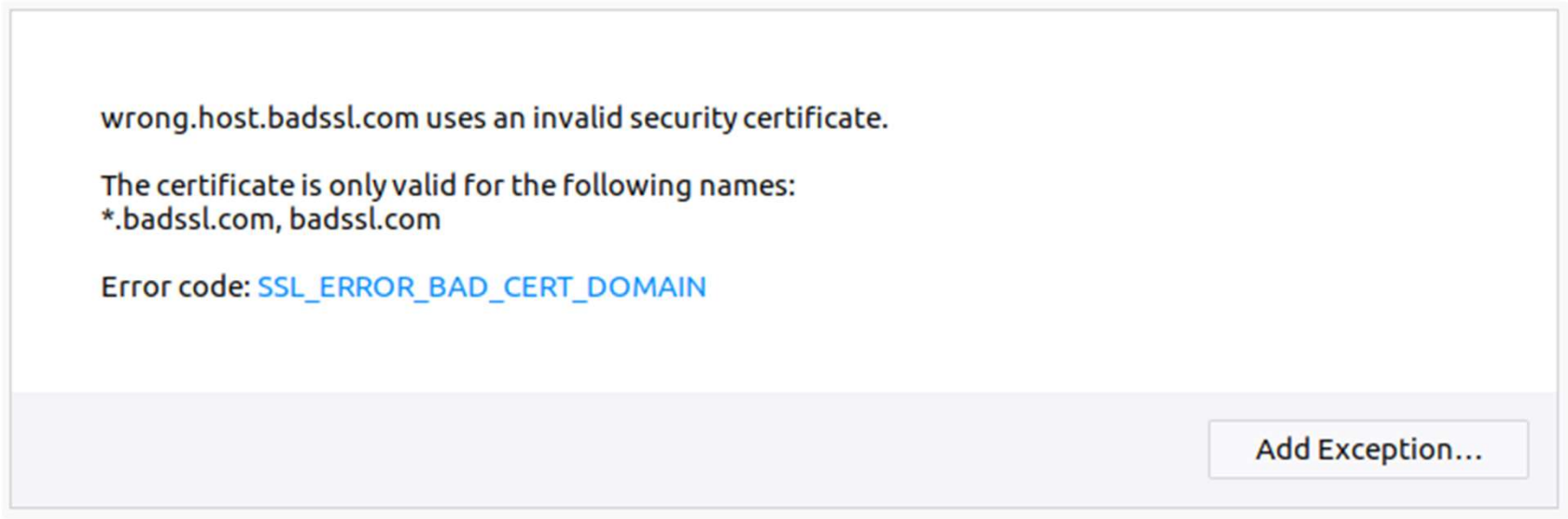
# Zertifikatsfehler

- Was passiert, wenn ein Zertifikat abgelaufen ist und nicht erneuert wurde?



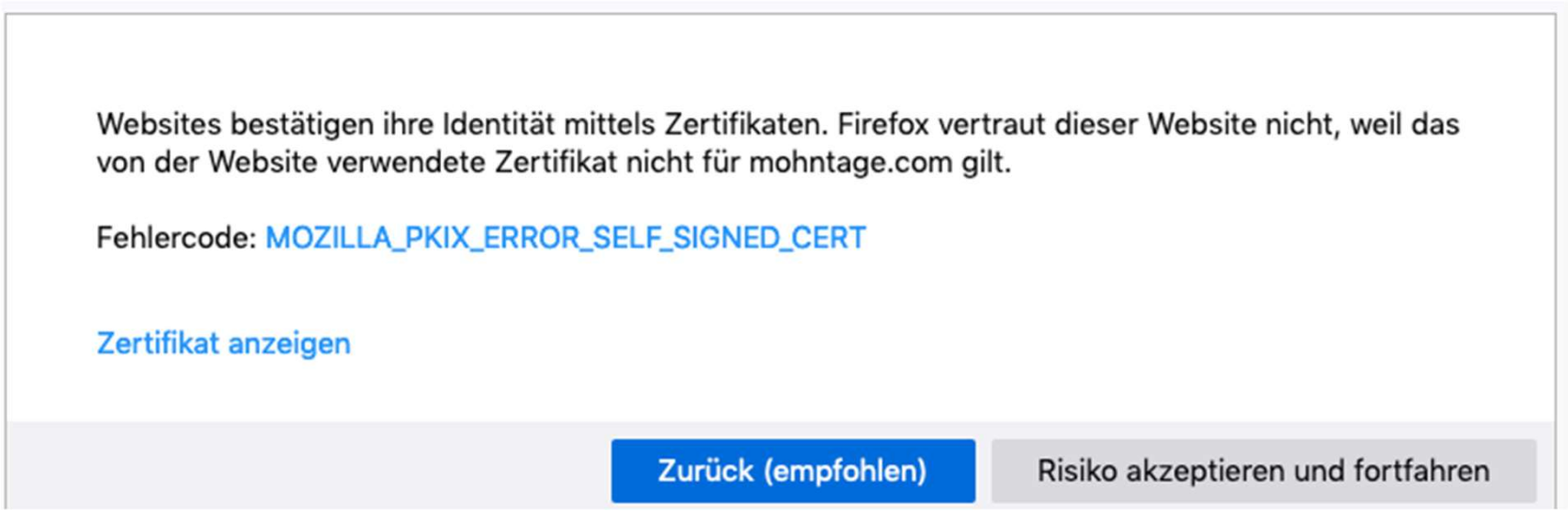
# Zertifikatsfehler

- Was passiert, wenn ein Zertifikats Common Name nicht mit dem Servernamen übereinstimmt?



# Zertifikatsfehler

- Was passiert, wenn ein Browser das Vertrauen eines Zertifikates nicht herleiten kann?




Diverse Zertifikatsprobleme unter [badssl.com](badssl.com)

# Validierung

- Domain Validation (DV)

  – Erfolgt bei Feststellung des Besitzes der Domain via DNS, HTTP, ACME oder Email
- Extended Validation (EV)

  – Erfolgt durch zusätzliche Prüfung der anfragenden Entität: Hauptsächlich vertraglich / nicht technisch

  – Banken, Versicherungen, Unternehmen welche besonderes Augenmerk auf Security legen

# Zurückziehung / Revocation

- Zertifikate können und sollen für ungültig erklärt werden wenn:

  - Private Key gestohlen/veröffentlicht wurde

  - Das Zertifikat nicht mehr benötigt wird
- Voraussetzung ist ein Revocation Certificate

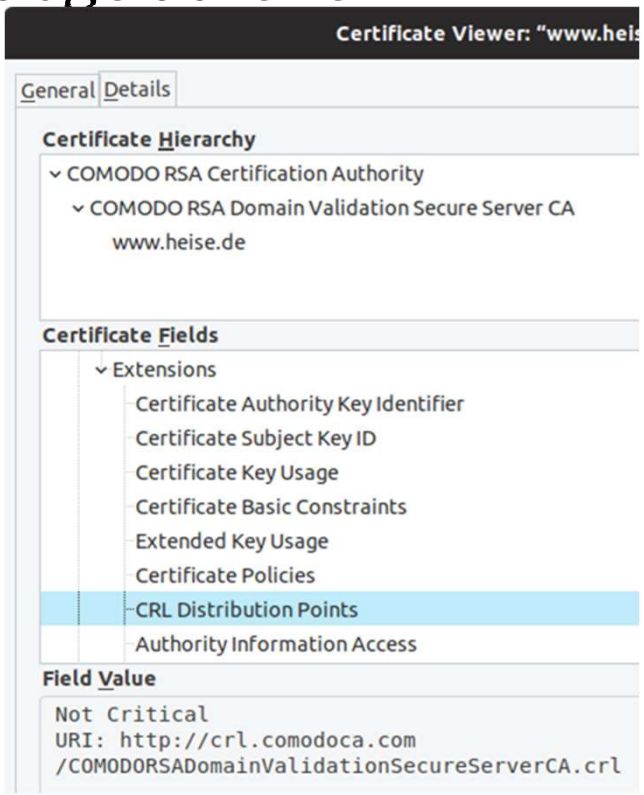
  - Wird auf Basis des Private Keys erstellt
- Empfehlung: Erstellen Sie immer gleich ein Revocation Certificate, wenn Sie ein neues Zertifikat erzeugen
- Zwei Möglichkeiten

  - Certificate Revocation List (CRL)

  - Online Certificate Status Protocol (OCSP)

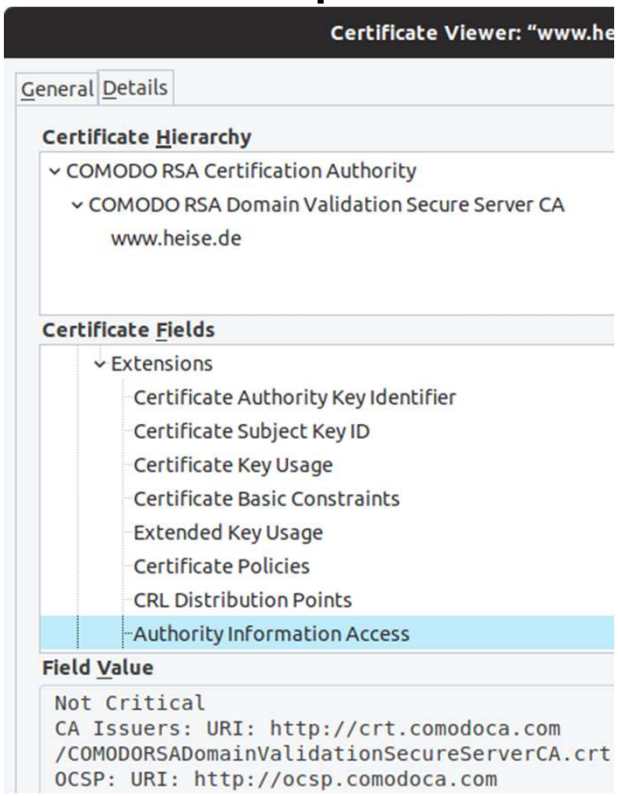
# Certificate Revocation List (CRL)

- Von der CA gepflegte Liste von Serial Numbers abgelaufener Zertifikate
- URL im Zertifikat verankert
- Gross und langsam
  - Beispiel COMODO (heise.de) 3.4 MB

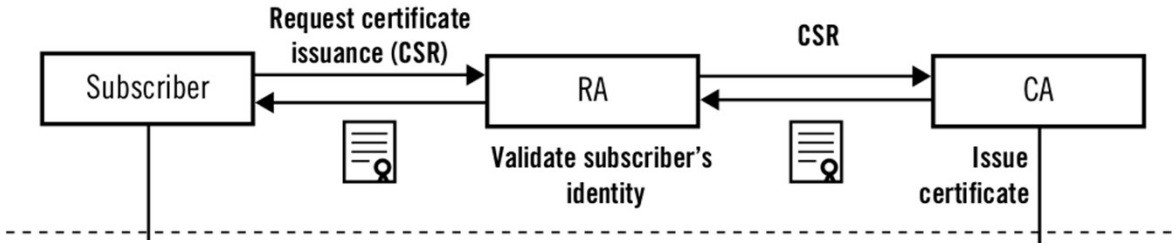


# Online Certificate Status Protocol (OCSP)

- Von CA betriebene OCSP Responder Service erlaubt Lookup via API
- URL im Zertifikat verankert
- Löst das Problem mit CRL und bringt neue
- Probleme mit sich:
  - Performance und Privatsphäre
- Lösung: OCSP Stapling



# Internet PKI Zusammenspiel



# Angriffe auf Internet PKI

- CAs sind einem grossen Risiko ausgesetzt
- Bereits angegriffen in der Vergangenheit:

– VeriSign, Thawte, StartCom (2008, 2011), CertStar, RapidSSL, Comodo, DigiNotar, DigiCert, TURKTRUST, …

# Angriff auf PKI am Beispiel DigiNotar

# Übungen & Labor

Labor: https://github.com/hexposed/Lab

# Videoempfehlungen für's Selbststudium

- DNS mit DoT und DoH (34 Min)