UNIVERSITÄT GREIFSWALD
Wissen lockt. Seit 1456

Verschlüsselung gehackt? Der Client ist schuld!

Universitätsrechenzentrum
G.K. Grubert

gordon.grubert@uni-greifswald.de

27.05.2019

Paradigmenwechsel

Pre-Snowden

Eine schlechte Verschlüsselung ist besser als gar keine Verschlüsselung.

Post-Snowden

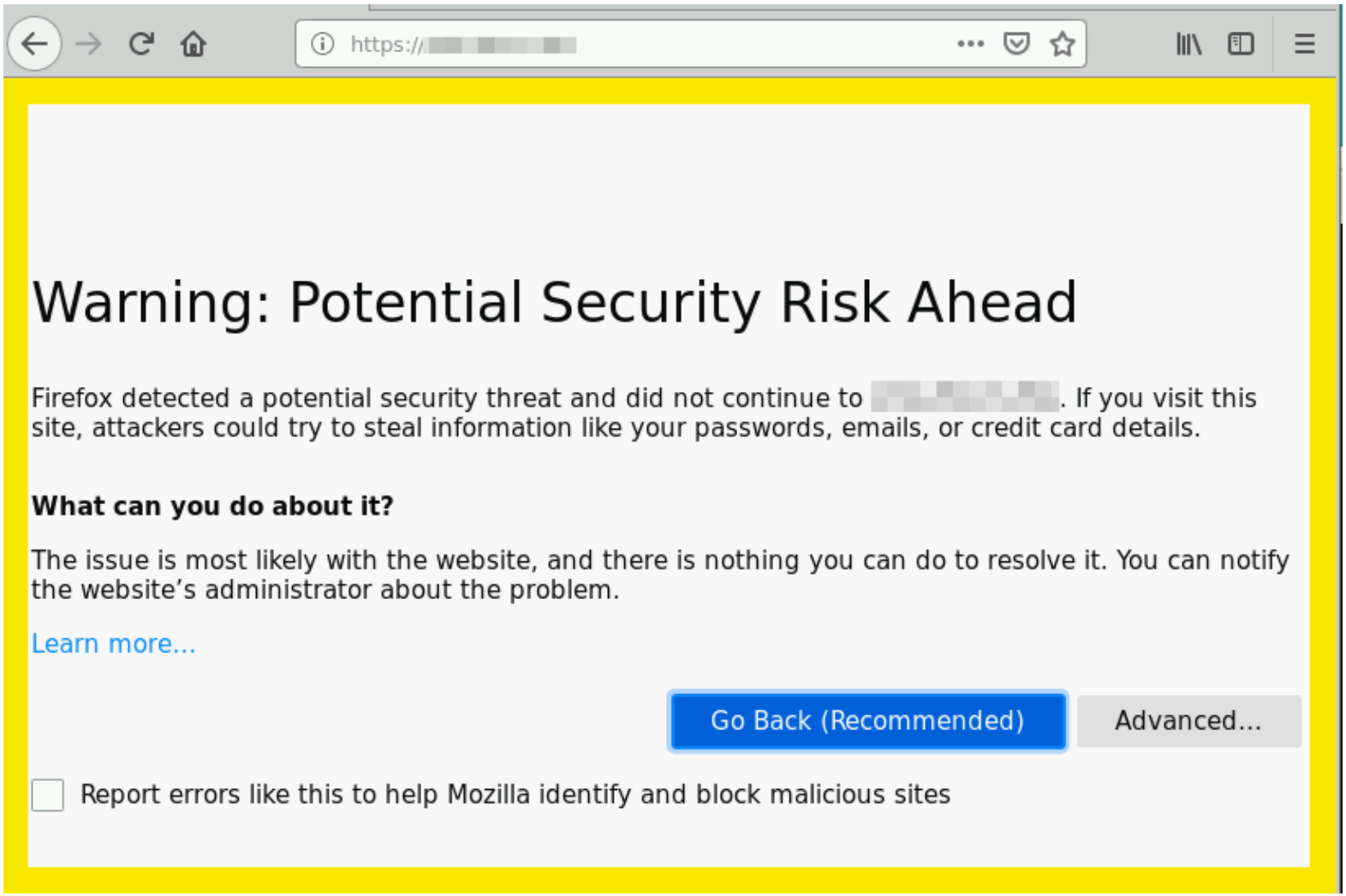
Eine schlechte Verschlüsselung ist keine Verschlüsselung!

Schutzbedarf

- Schutzbedarf muss individuell festgelegt werden
- wenn Verschlüsselung verlangt wird, muss diese auch dem Stand der Technik entsprechen

$\Rightarrow$ ansonsten kann auch konsequenterweise auf die Verschlüsselung verzichtet werden

Beispiele aus dem Alltag (Laie)



Beispiele aus dem Alltag (Laie)

- Browser stellt gravierendes Problem mit der Verschlüsselung fest
- **JEDER** Laie sollte wissen, dass er lieber nicht weitermachen sollte, v.a. wenn dies das Login für das Onlinebanking ist

Beispiele aus dem Alltag (fortgeschritten)

Etwas kniffeliger:

Beispiele aus dem Alltag (fortgeschritten)

Etwas kniffeliger:

Beispiele aus dem Alltag (fortgeschritten)

Etwas kniffeliger:

- man muss immer auf die **Domain** achten, mit der man letztlich verbunden ist
- Problem der **CAs**, der vertraut werden

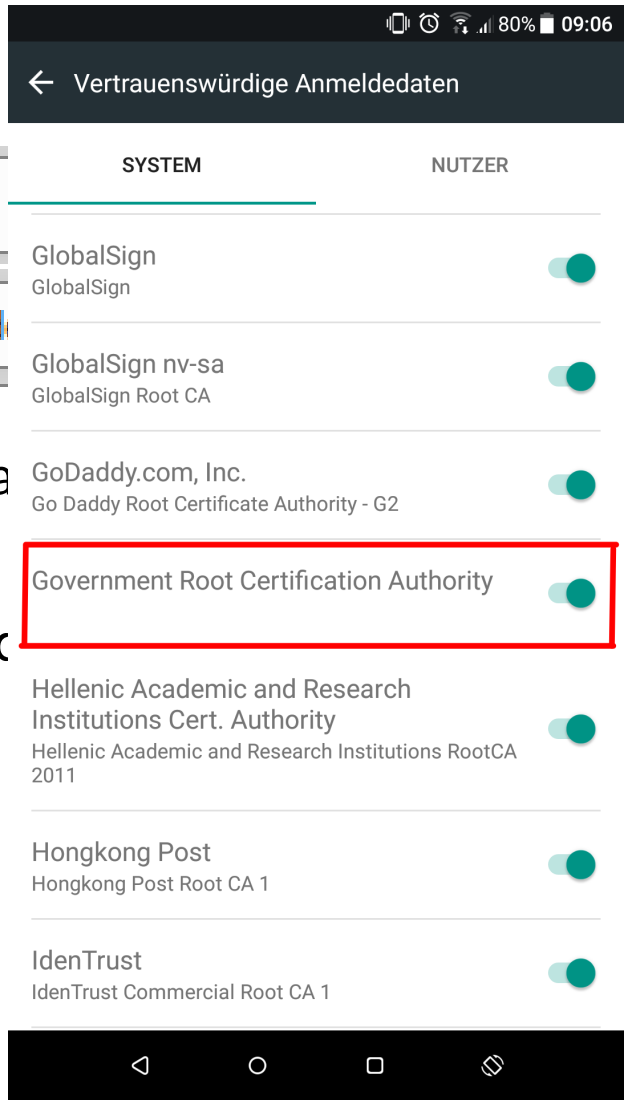
⇒ Wer vertraut denn noch der **Government Root CA**?

Beispiele aus dem Alltag (fortgeschritten)

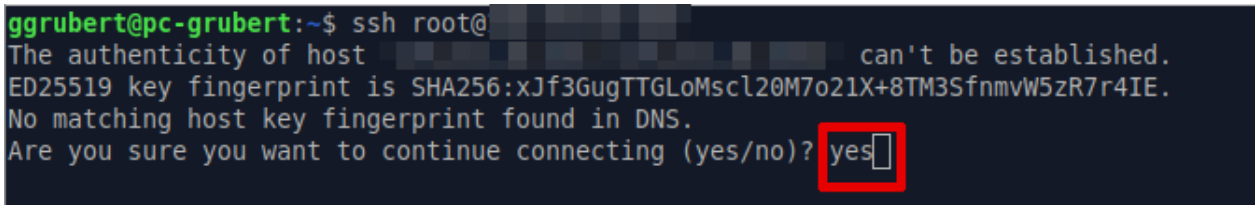
Etwas kniffeliger:

- man muss immer auf die **Domain** a
  letztlich verbunden ist
- Problem der **CAs**, der vertraut werd

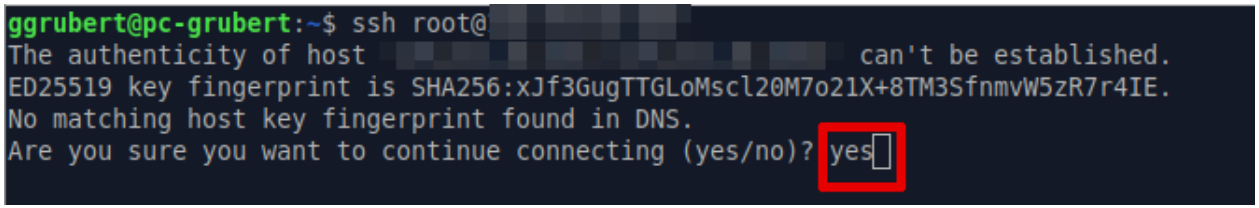
⇒ Wer vertraut denn noch der **Government Root CA**?



Beispiele aus dem Alltag (Nerd)

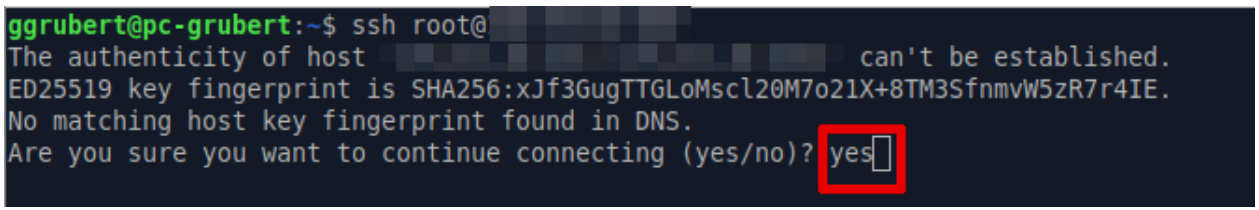


Beispiele aus dem Alltag (Nerd)



- dies entspricht der **Laien-Warnung** im Browser
- ein ungeprüftes Fortsetzen entspricht einer Klartextkommunikation

Beispiele aus dem Alltag (Nerd)



- dies entspricht der **Laien-Warnung** im Browser
- ein ungeprüftes Fortsetzen entspricht einer Klartextkommunikation

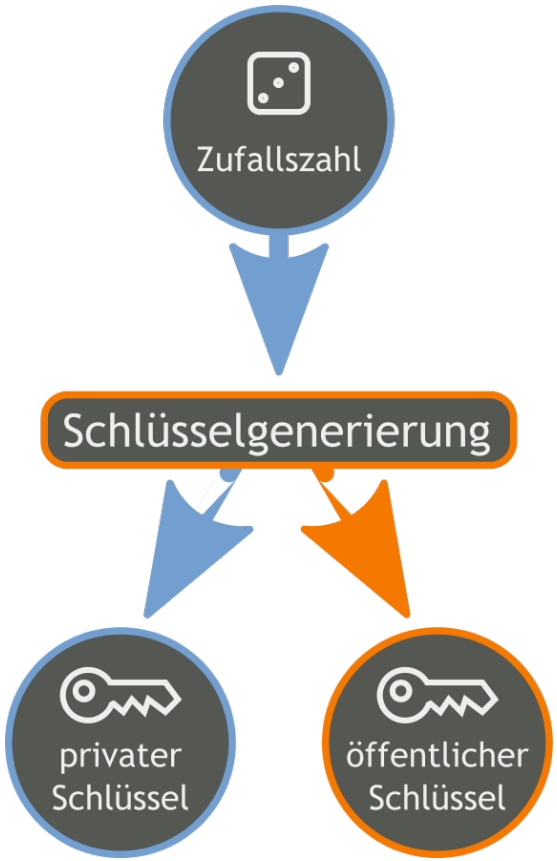
Für alle gezeigten Probleme gibt es Lösungen!

Asymmetrische Verschlüsselung

- anstelle eines Schlüssels wird ein **Schlüsselpaar** verwendet; RSA-Verfahren (**R**ivest, **S**hamir und **A**dleman)
- hier spielt der **Zufall** eine ganz entscheidende Rolle
- aktuelle Methoden:
  - RSA: basiert auf der Zerlegung in Primzahlen
  - ED25519: basiert auf elliptischen Kurven

Asymmetrische Verschlüsselung

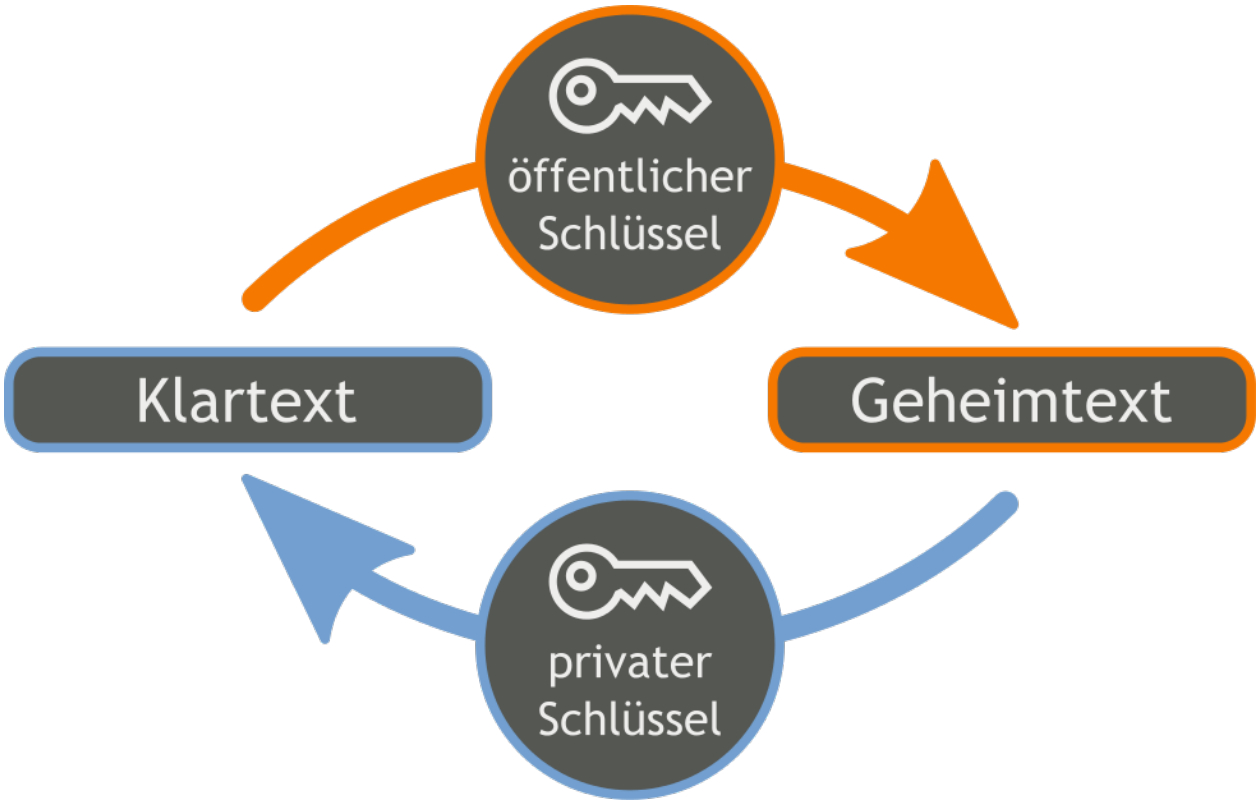
Schüsselerzeugung



https://de.wikipedia.org/w/index.php?title=Asymmetrisches_Kryptosystem&oldid=133963715

Asymmetrische Verschlüsselung

Ver- und Entschlüsselung



https://de.wikipedia.org/w/index.php?title=Asymmetrisches_Kryptosystem&oldid=133963715

Asymmetrische Verschlüsselung

- privater Schlüssel muss nicht mehr übertragen werden
- Problem gelöst, da nur noch ein öffentlicher Schlüssel übertragen werden muss?

Asymmetrische Verschlüsselung

- privater Schlüssel muss nicht mehr übertragen werden
- Problem gelöst, da nur noch ein öffentlicher Schlüssel übertragen werden muss?

Nein!

Solange öffentlicher Schlüssel ohne Verifikation übertragen wird, ist die Gegenstelle unbekannt $\Rightarrow$ **unsichere Kommunikation**

Cipher Suite

- standardisierte Sammlung kryptographischer Verfahren
- bei **T**ransport **L**ayer **S**ecurity wird der Aufbau einer gesicherten Verbindung durch Cipher Suite festgelegt, z.B. **TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256**:
  - Schlüsselaustausch (ECDHE)
  - Authentifizierung (RSA)
  - Verschlüsselung (AES_128_GCM)
  - Hashfunktion (SHA256)

https://de.wikipedia.org/w/index.php?title=Cipher_Suite&oldid=168101214

Hashfunktion

Definition

Eine Hashfunktion ist eine eindeutige Einwegfunktion. Im kryptographischen Kontext sollte diese **kollisionsresistent** sein. Zudem gibt es *salted* Hashfunktionen.

https://de.wikipedia.org/w/index.php?title=Kryptologische_Hashfunktion&oldid=169261236

Hashfunktion

Definition

Eine Hashfunktion ist eine eindeutige Einwegfunktion. Im kryptographischen Kontext sollte diese **kollisionsresistent** sein. Zudem gibt es *salted* Hashfunktionen.

Beispiele:

- MD5 (**dead**)
- SHA-1 (**obsolete**)
- SHA-2

⇒ Nach aktuellem Stand der Technik sollte derzeit mindestens SHA-2 mit einer Schlüssellänge von 256 Bit verwendet werden.

https://de.wikipedia.org/w/index.php?title=Kryptologische_Hashfunktion&oldid=169261236

Protokolle

- SSLv2 (**dead**)
- SSLv3 (**dead**)
- TLSv1.0 (**obsolete**)
- TLSv1.1 (**obsolete**)
- TLSv1.2 (**strongly recommended!**)
- TLSv1.3
  (**recommended, sobald im jeweiligen System verfügbar!**)

Mindeststandard des BSI zur Verwendung von Transport Layer Security (TLS) vom 05.04.2019

Angebot beschränken

- Aktivierung/Bereitstellung von Protokollen und Cipher Suiten, die man als Dienstanbieter vertreten kann
- Hinweis:
  schwächste Einstellung gilt i.d.R. für alle Clients, da der **Client** letztlich entscheidet, was er unterstützt

Angebot beschränken

- Aktivierung/Bereitstellung von Protokollen und Cipher Suiten, die man als Dienstanbieter vertreten kann
- Hinweis:
  schwächste Einstellung gilt i.d.R. für alle Clients, da der **Client** letztlich entscheidet, was er unterstützt
- Universität Greifswald:
  - TLSv1.2 only
  - nur starke Cipher Suiten

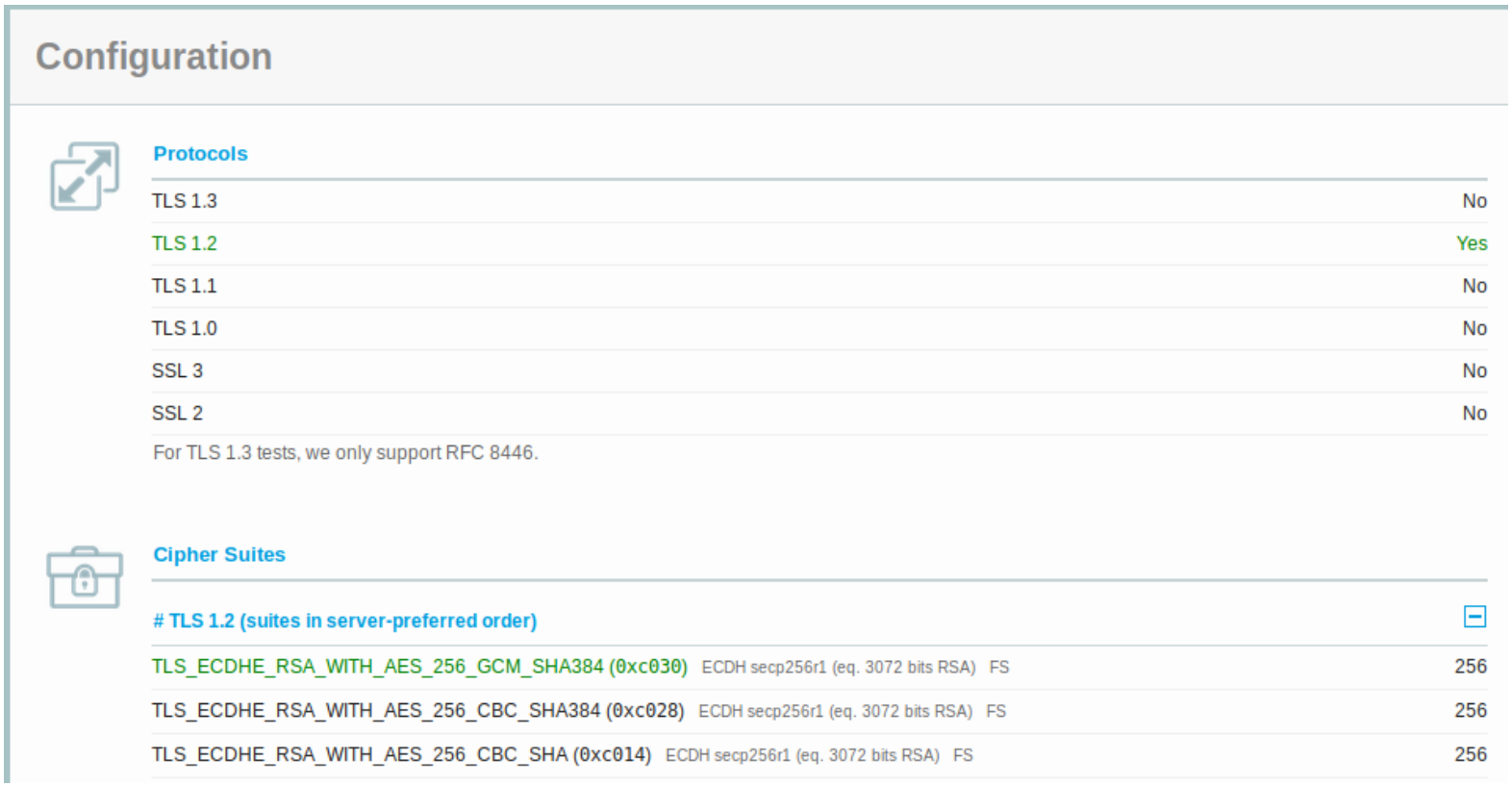
Beispiel IMAP-Server

```
Testing protocols via sockets

SSLv2      not offered (OK)
SSLv3      not offered (OK)
TLS 1      not offered
TLS 1.1    not offered
TLS 1.2    offered (OK)

Testing ~standard cipher categories

NULL ciphers (no encryption)                  not offered (OK)
Anonymous NULL Ciphers (no authentication)    not offered (OK)
Export ciphers (w/o ADH+NULL)                 not offered (OK)
LOW: 64 Bit + DES encryption (w/o export)     not offered (OK)
Weak 128 Bit ciphers (SEED, IDEA, RC[2,4])    not offered (OK)
Triple DES Ciphers (Medium)                   not offered (OK)
High encryption (AES+Camellia, no AEAD)       offered (OK)
Strong encryption (AEAD ciphers)              offered (OK)

Testing vulnerabilities

BEAST (CVE-2011-3389)                    no SSL3 or TLS1 (OK)
```

https://testssl.sh/

Eingeschränkte Client-Kompatibilität: von Outlook 2010 abgesehen - nein

Beispiel Web-Server



Configuration

Protocols

TLS 1.3	No
TLS 1.2	Yes
TLS 1.1	No
TLS 1.0	No
SSL 3	No
SSL 2	No

For TLS 1.3 tests, we only support RFC 8446.

Cipher Suites

 # TLS 1.2 (suites in server-preferred order)

TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)	ECDH secp256r1 (eq. 3072 bits RSA)	FS	256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)	ECDH secp256r1 (eq. 3072 bits RSA)	FS	256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)	ECDH secp256r1 (eq. 3072 bits RSA)	FS	256

https://www.ssllabs.com

Beispiel Web-Server

Eingeschränkte Client-Kompatibilität:

- Android ab 4.4.2
- Chrome
- Firefox
- IE ab 11 (auch unter Win 7)
- Edge
- Safari ab 6

Beispiel Web-Server

Eingeschränkte Client-Kompatibilität:

- Android ab 4.4.2
- Chrome
- Firefox
- IE ab 11 (auch unter Win 7)
- Edge
- Safari ab 6

$\Rightarrow$ **AKZEPTABEL**

Idee

- Bildung von Vertrauenshierarchien, um die Echtheit von Public Keys/Zertifikaten sicherzustellen
- Bedingt **weitere Anforderungen**:
  - Nutzung von **C**ertificate **A**uthoritys
  - Verwendung von **C**ertificate **R**evocation **L**ists
  - alternativ Verwendung des **O**nline **C**ertificate **S**tatus **P**rotocols

Idee

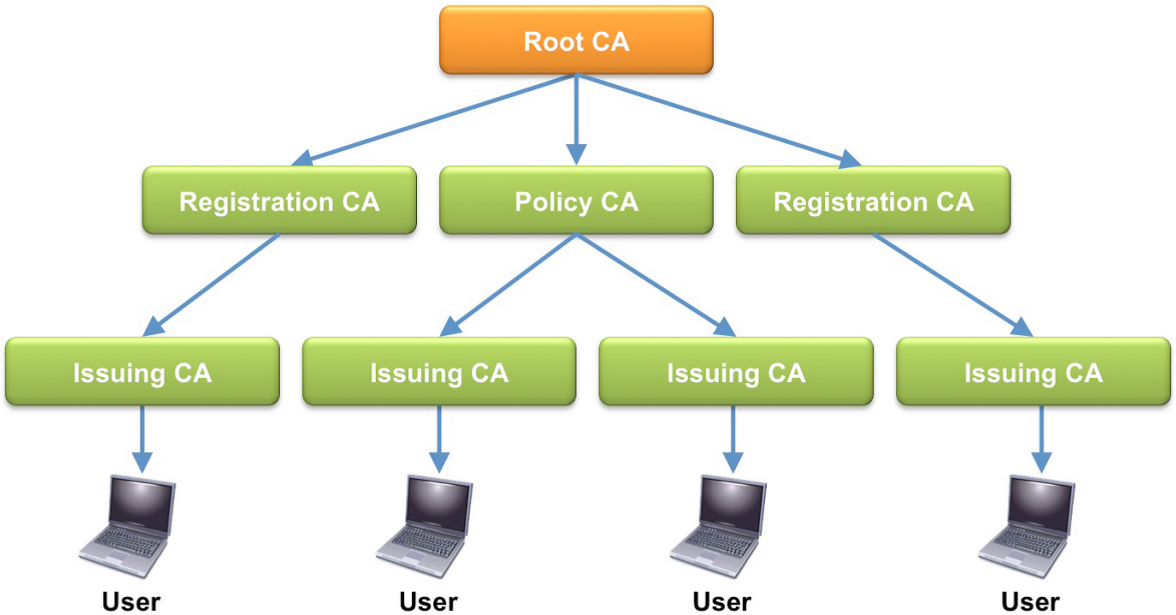
- Bildung von Vertrauenshierarchien, um die Echtheit von Public Keys/Zertifikaten sicherzustellen
- Bedingt **weitere Anforderungen**:
  - Nutzung von **C**ertificate **A**uthoritys
  - Verwendung von **C**ertificate **R**evocation **L**ists
  - alternativ Verwendung des **O**nline **C**ertificate **S**tatus **P**rotocols

$\Rightarrow$ Anforderungen sind komplett **clientseitig**

Realisierung

- weltweit existiert Struktur **paralleler Vertrauenshierarchien**
- in der Wurzel steht immer die **Root Certificate Authority**, die das Vertrauen an untergeordnete Zertifizierungsstellen (**C**ertificate **A**uthoritys) weitergibt
- Nutzung von Zertifikaten (signierter öffentlicher Schlüssel)

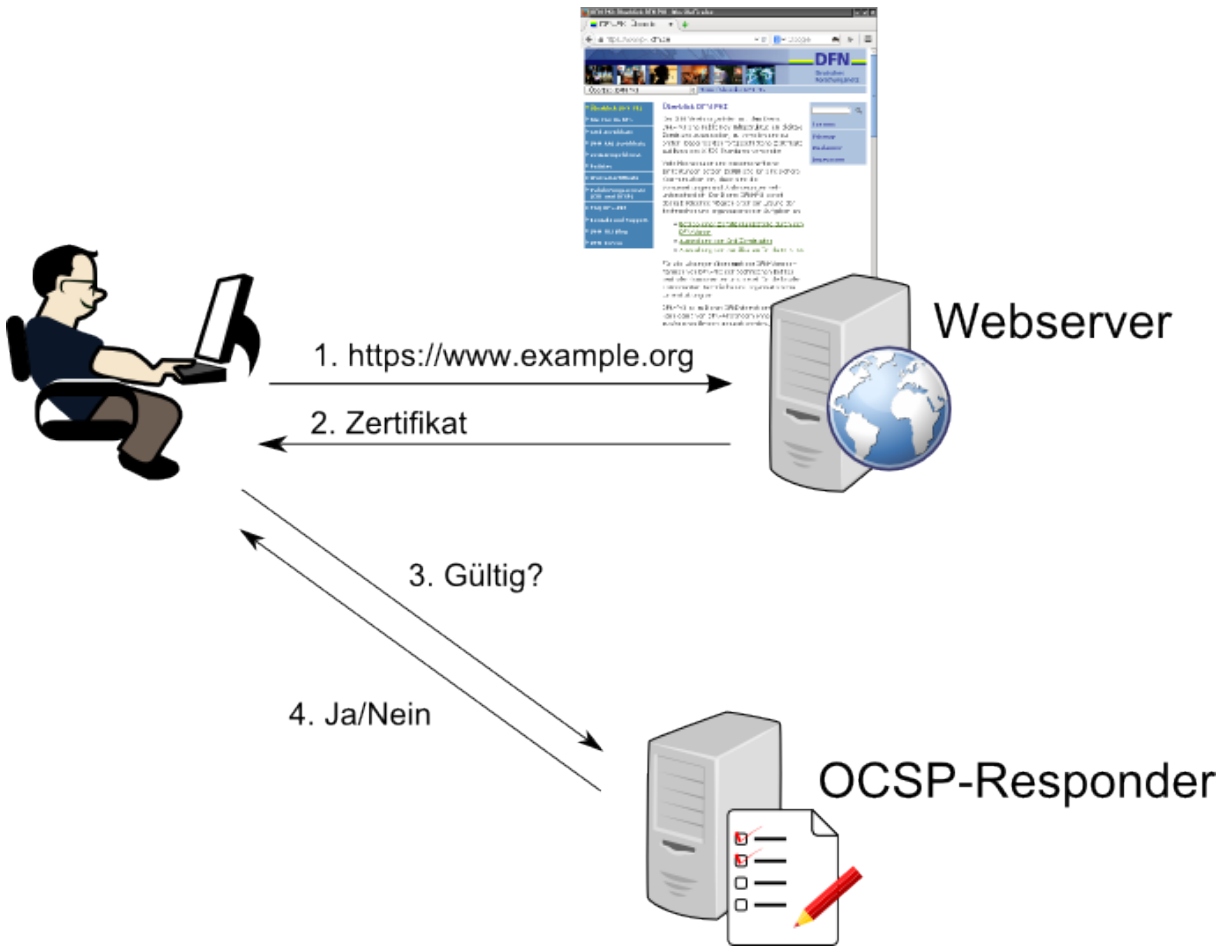
Realisierung



https://www.thales-esecurity.com/blogs/2013/april/security-considerations-of-a-pki

Online Certificate Status Protocol

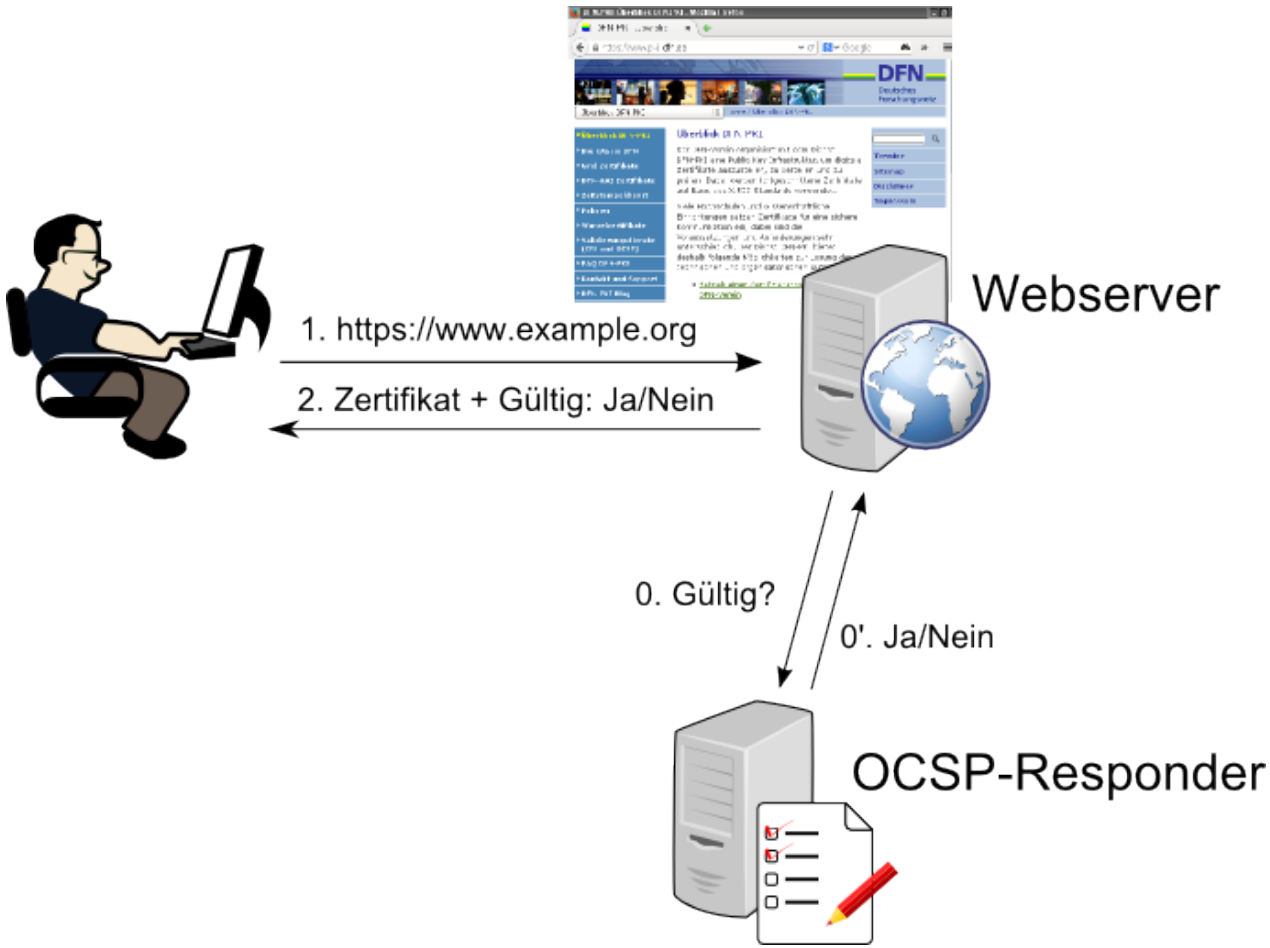
Klassisch



https://blog.pki.dfn.de/2015/03/mehr-privacy-fuer-den-nutzer-ocsp-stapling/

Online Certificate Status Protocol

OCSP Stapling



https://blog.pki.dfn.de/2015/03/mehr-privacy-fuer-den-nutzer-ocsp-stapling/

PKI korrekt verwenden (Server)

1. Erstellung Private Key und Certificate Request
2. CA stellt das Zertifikat aus

```
# [...]
Serial Number:
    20:02:33:a2:3a:0b:43:b5:a7:29:f7:4f

Issuer: C = DE, O = Verein zur Foerderung eines Deutschen Forschungsnetzes e. V.,\
        OU = DFN-PKI, CN = DFN-Verein Global Issuing CA

Validity
    Not Before: Nov  7 09:53:26 2018 GMT
    Not After : Feb  8 09:53:26 2021 GMT

X509v3 Subject Alternative Name:
    DNS:typo3.uni-greifswald.de
# [...]
```

PKI korrekt verwenden (Client)

Was muss der Client mit dem Serverzertifikat tun?

Der eigentliche Schutz und die Qualität bzw. deren Überprüfung obligt einzig dem **Client/Empfänger** - und nur ihm!

PKI korrekt verwenden (Client)

Was muss der Client mit dem Serverzertifikat tun?

Der eigentliche Schutz und die Qualität bzw. deren Überprüfung obligt einzig dem **Client/Empfänger** - und nur ihm!

- Überprüfung der Zertifikatsdaten selbst, z.B.
  - Ist das Zertifikat zeitlich gültig?
  - Kann das Zertifikat anhand einer Root CA überprüft werden?
  - Stimmt der angefragte FQDN mit dem FQDN des Zertifikates überein?

PKI korrekt verwenden (Client)

Was muss der Client mit dem Serverzertifikat tun?

Der eigentliche Schutz und die Qualität bzw. deren Überprüfung obligt einzig dem **Client/Empfänger** - und nur ihm!

- Überprüfung der Zertifikatsdaten selbst, z.B.
  - Ist das Zertifikat zeitlich gültig?
  - Kann das Zertifikat anhand einer Root CA überprüft werden?
  - Stimmt der angefragte FQDN mit dem FQDN des Zertifikates überein?
- Nutzung des **O**nline **C**ertificate **S**tatus **P**rotocol
  Wurde das Zertifikat bereits von der CA widerrufen?
  (optimiertes Verfahren: OCSP Stapling)

PKI korrekt verwenden (Client)

Was muss der Client mit dem Serverzertifikat tun?

Der eigentliche Schutz und die Qualität bzw. deren Überprüfung obligt einzig dem **Client/Empfänger** - und nur ihm!

- Überprüfung der Zertifikatsdaten selbst, z.B.
  - Ist das Zertifikat zeitlich gültig?
  - Kann das Zertifikat anhand einer Root CA überprüft werden?
  - Stimmt der angefragte FQDN mit dem FQDN des Zertifikates überein?
- Nutzung des **O**nline **C**ertificate **S**tatus **P**rotocol
  Wurde das Zertifikat bereits von der CA widerrufen?
  (optimiertes Verfahren: OCSP Stapling)
- Nutzung einer **C**ertificate **R**evocation **L**ist
  Wurde das Zertifikat bereits von der CA widerrufen?
  (alternativ zu OCSP)

PKI korrekt verwenden (Client)

Warum muss der Client dies alles tun?

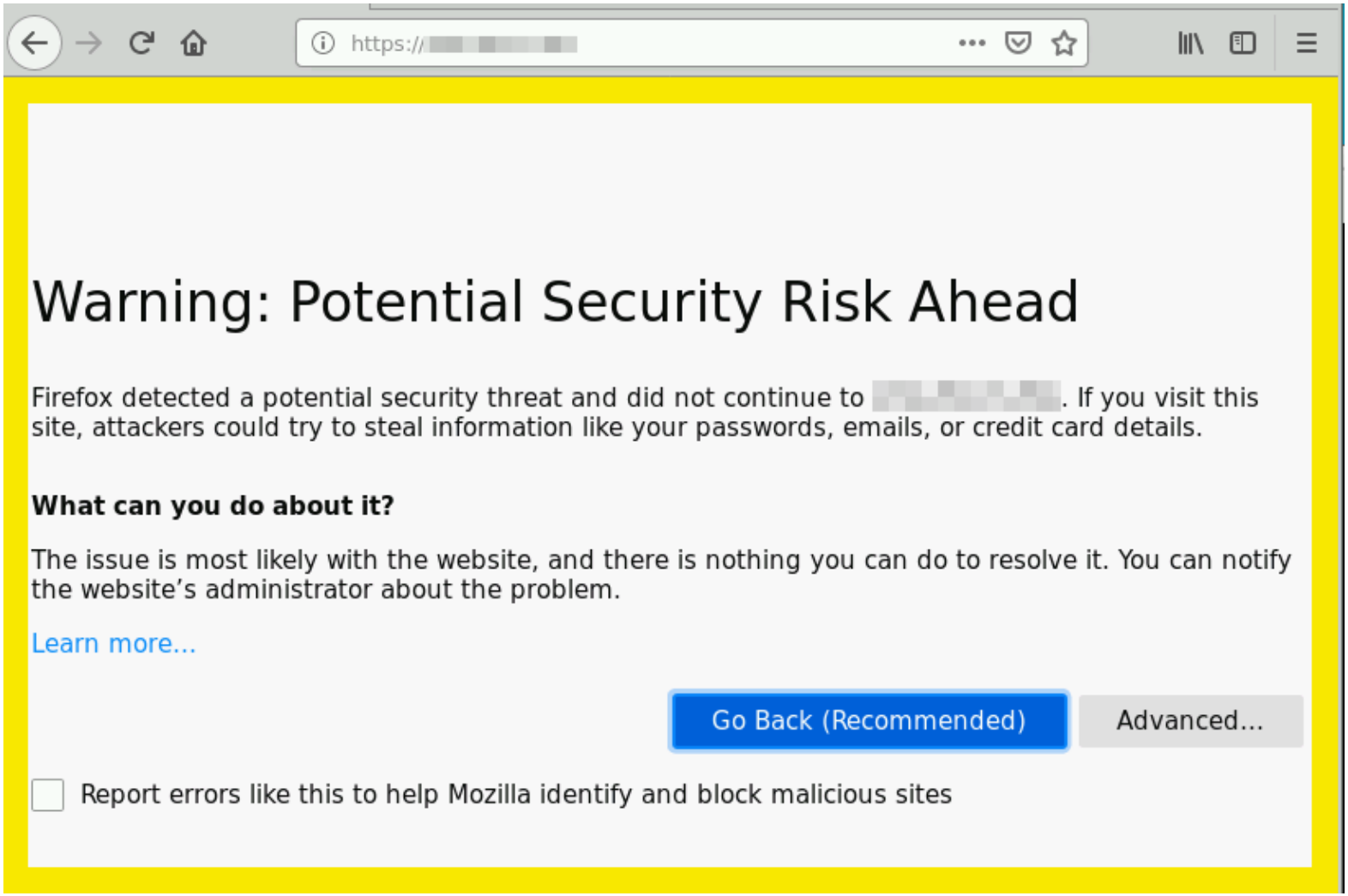
- ohne Abgleich der Zertifikatsdaten öffnet man MITM-Attacken Tür und Tor
- eine CA stellt **befristete** Zertifikate aus

  (statisches System)
- dynamische Änderungen erfolgen ausschließlich via OCSP oder CRL

⇒ Dynamik macht jedoch die Sicherheit aus!

PKI korrekt verwenden (Client)

Zur Erinnerung:



Die entscheidene Frage

Was ist ein Client im Sinne der Verschlüsselung?

Dies ist i.d.R. das System, welches verschlüsselte Verbindungen initiiert.

Die entscheidene Frage

Was ist ein Client im Sinne der Verschlüsselung?

Dies ist i.d.R. das System, welches verschlüsselte Verbindungen initiiert.

- Mensch, der im Browser https-Webseiten aufruft

Die entscheidene Frage

Was ist ein Client im Sinne der Verschlüsselung?

Dies ist i.d.R. das System, welches verschlüsselte Verbindungen initiiert.

- Mensch, der im Browser https-Webseiten aufruft
- Mailserver, der LDAP-Abfragen zur Nutzerauthentifikation macht

Die entscheidene Frage

Was ist ein Client im Sinne der Verschlüsselung?

Dies ist i.d.R. das System, welches verschlüsselte Verbindungen initiiert.

- Mensch, der im Browser https-Webseiten aufruft
- Mailserver, der LDAP-Abfragen zur Nutzerauthentifikation macht
- Radsecproxy, der Anfragen an einen weiteren Radsecproxy schickt

Die entscheidene Frage

Was ist ein Client im Sinne der Verschlüsselung?

Dies ist i.d.R. das System, welches verschlüsselte Verbindungen initiiert.

- Mensch, der im Browser https-Webseiten aufruft
- Mailserver, der LDAP-Abfragen zur Nutzerauthentifikation macht
- Radsecproxy, der Anfragen an einen weiteren Radsecproxy schickt
- Telefon, welches TLS-gesicherte Verbindungen startet

...

Die entscheidene Frage

Was ist ein Client im Sinne der Verschlüsselung?

Dies ist i.d.R. das System, welches verschlüsselte Verbindungen initiiert.

- Mensch, der im Browser https-Webseiten aufruft
- Mailserver, der LDAP-Abfragen zur Nutzerauthentifikation macht
- Radsecproxy, der Anfragen an einen weiteren Radsecproxy schickt
- Telefon, welches TLS-gesicherte Verbindungen startet

...

In allen Fällen müssen alle zuvor genannten Clientanforderungen erfüllt werden!

LDAP-Zugriff (falsch)

- **/etc/ldap/ldap.conf** (sehr häufig)

```
#
# LDAP Defaults
#

# See ldap.conf(5) for details
# This file should be world readable but not world writable.

#BASE   dc=example,dc=com
#URI    ldap://ldap.example.com ldap://ldap-master.example.com:666

#SIZELIMIT      12
#TIMELIMIT      15
#DEREF          never
```

LDAP-Zugriff (falsch)

- **/etc/ldap/ldap.conf** (sehr häufig)

```
#
# LDAP Defaults
#

# See ldap.conf(5) for details
# This file should be world readable but not world writable.

#BASE   dc=example,dc=com
#URI    ldap://ldap.example.com ldap://ldap-master.example.com:666

#SIZELIMIT      12
#TIMELIMIT      15
#DEREF          never
```

- LDAP-Request

```
ldapsearch -H ldaps://ldap.uni-greifswald.de -LLL -x -w PASSWORD -b BASEDN -D BINDDN FILTER
```

LDAP-Zugriff (falsch)

- **/etc/ldap/ldap.conf** (sehr häufig)

```
#
# LDAP Defaults
#

# See ldap.conf(5) for details
# This file should be world readable but not world writable.

#BASE   dc=example,dc=com
#URI    ldap://ldap.example.com ldap://ldap-master.example.com:666

#SIZELIMIT      12
#TIMELIMIT      15
#DEREF          never
```

- LDAP-Request

```
ldapsearch -H ldaps://ldap.uni-greifswald.de -LLL -x -w PASSWORD -b BASEDN -D BINDDN FILTER
```

⇒ der Client (in diesem Fall ein Server, der eine LDAP-Abfrage ausführt), validiert das LDAP-Serverzertifikat nicht

⇒ Laien-Fehler!

LDAP-Zugriff (korrekt)

- **/etc/ldap/ldap.conf**

```
TLS_CACERT      /etc/ssl/certs/ca-certificates.crt
                # systemabhängig

TLS_REQCERT     demand

TLS_CRLFILE     /etc/CertsAndKeys/CRL.pem
                # systemabhängig

TLS_CRLCHECK    all
```

Radsecproxy-Verbindungen

```
# Global TLS settings
# ###################
tls default {
    # Specification of valid CA certificates
    CACertificatePath  /etc/ssl/certs

    # [...]

    # Enable CRLs
    CRLCheck on
    cacheExpiry 3600
}

# [...]

# Set target server with FQDN
# ###########################
server targetserver {
    host fqdn_of_target_server
    type tls
    StatusServer on
}
```

IP-Telefonie

Setup:

- FW-Server wurde mit Zertifikat ausgestattet, welches **nicht** zum Aufruf-FQDN passt
- Telefon wurde Test-FQDN mitgeteilt und es wurde zum FW-Upgrade aufgefordert
- Nutzer erhält in diesem Setup die mehrfach gezeigte Browserwarnung

Im Log des Webservers sieht man den Download der Firmware:

```
<IP> - - [03/Apr/2019:10:34:57 +0200] "GET <FW-File> HTTP/1.1" 200 29773226 "-" \
        "Mozilla/4.0 (compatible; <MODEL> <CURR FW VERSION>)"
```

IP-Telefonie

Setup:

- FW-Server wurde mit Zertifikat ausgestattet, welches **nicht** zum Aufruf-FQDN passt
- Telefon wurde Test-FQDN mitgeteilt und es wurde zum FW-Upgrade aufgefordert
- Nutzer erhält in diesem Setup die mehrfach gezeigte Browserwarnung

Im Log des Webservers sieht man den Download der Firmware:

```
<IP> - - [03/Apr/2019:10:34:57 +0200] "GET <FW-File> HTTP/1.1" 200 29773226 "-" \
        "Mozilla/4.0 (compatible; <MODEL> <CURR FW VERSION>)"
```

Das Telefon müsste Zugriff auf den Webserver verweigern!

Eine der blödesten Ideen: Certificate Pinning

- Domain legt explizit Zertifikate fest, die für Domain gelten
- **Client** muss bei Initialvalidierung dennoch korrekt arbeiten

https://de.wikipedia.org/w/index.php?title=HTTP_Public_Key_Pinning&oldid=185911896

Eine der blödesten Ideen: Certificate Pinning

- Domain legt explizit Zertifikate fest, die für Domain gelten
- **Client** muss bei Initialvalidierung dennoch korrekt arbeiten

⇒ kein Sicherheitsgewinn, da Sicherheit wieder vom Client abhängt

https://de.wikipedia.org/w/index.php?title=HTTP_Public_Key_Pinning&oldid=185911896

Eine der blödesten Ideen: Certificate Pinning

- Domain legt explizit Zertifikate fest, die für Domain gelten
- **Client** muss bei Initialvalidierung dennoch korrekt arbeiten

⇒ kein Sicherheitsgewinn, da Sicherheit wieder vom Client abhängt

Nachteile:

- zu komplex in der Anwendung für Serverbetreiber
- viele Clients pinnen Zertifikate bis in alle Ewigkeit

⇒ Probleme beim Zertifikatstausch

https://de.wikipedia.org/w/index.php?title=HTTP_Public_Key_Pinning&oldid=185911896

Eine der blödesten Ideen: Certificate Pinning

- Domain legt explizit Zertifikate fest, die für Domain gelten
- **Client** muss bei Initialvalidierung dennoch korrekt arbeiten

⇒ kein Sicherheitsgewinn, da Sicherheit wieder vom Client abhängt

Nachteile:

- zu komplex in der Anwendung für Serverbetreiber
- viele Clients pinnen Zertifikate bis in alle Ewigkeit

⇒ Probleme beim Zertifikatstausch

Fazit

Certificate Pinning bringt Null Sicherheitsmehrwert - es gefährtet durch Fehlimplementierung die Sicherheit sogar noch extrem.

https://de.wikipedia.org/w/index.php?title=HTTP_Public_Key_Pinning&oldid=185911896

Sicheres Domain Name System

- Warum ist eine Alternative zur PKI sinnvoll?

Sicheres Domain Name System

- Warum ist eine Alternative zur PKI sinnvoll?
- PKI hat grundlegende Probleme bzw. zu hohe technische Anforderungen:
  - Berücksichtigung der CA-Prüfung
  - Verwendung von CRLs/OCSP
  - technisch kann **jede** CA für **jede** Domain Zertifikate ausstellen

    (CAA-Records im DNS verhindern dies nur organisatorisch!)

    https://de.wikipedia.org/w/index.php?title=DNS_Certification_Authority_Authorization&oldid=186737985

Sicheres Domain Name System

- Warum ist eine Alternative zur PKI sinnvoll?
- PKI hat grundlegende Probleme bzw. zu hohe technische Anforderungen:
  - Berücksichtigung der CA-Prüfung
  - Verwendung von CRLs/OCSP
  - technisch kann **jede** CA für **jede** Domain Zertifikate ausstellen

    (CAA-Records im DNS verhindern dies nur organisatorisch!)

    https://de.wikipedia.org/w/index.php?title=DNS_Certification_Authority_Authorization&oldid=186737985

⇒ **Vertrauensproblem**

Vorteile

- es gibt nur **einen** Rootkey („*DNS global CA*“), dem man vertrauen muss
- weitere Hierarchie ist **geolokalisiert**, d.h. nur die DENIC ist für de-Domains zuständig
- Zertitifkate benötigen kein Ablaufdatum

  (es gibt keinen technischen Grund, Zertifikate nach fester Zeit zu tauschen)
- CRL/OCSP unnötig

⇒ zur Sperrung eines Zertifikates entfernt man dieses einfach aus dem DNS

(es bleiben nur normale DNS-TTLs)

Warum nutzt man noch PKIs?

Reine Spekulation

- rein kommerzielle Interessen
- CAs sind ein gigantischer Industriezweig
- Einsatz von DNSSec an dieser Stelle würde diese gesamte Industrie ad absurdum führen

Mögliche technische Gründe:

- Komplexität bei Einführung von DNSSec?
- Weitere: **NEIN**

Grundlagen

- befragen Sie einfach die Suchmaschine Ihres geringsten Misstrauens
- weitere Vorträge auf dieser Konferenz

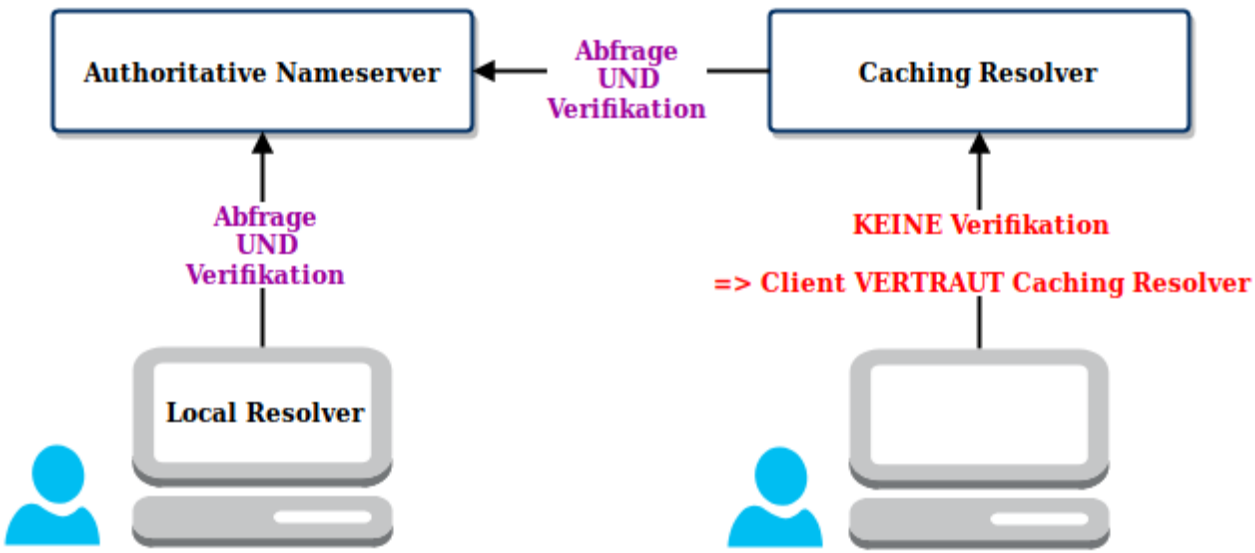
Designproblem Nr. 1?

- Verifikation i.d.R. nur durch *DNS-Server-Software*
- authoritative Nameserver verifiziert sich nicht selbst
  - $\Rightarrow$ ohne Caching-Resolver in der eigenen Einrichtung i.d.R. keine Verifikation möglich
  - $\Rightarrow$ **Caching-Resolver vs. ACL-View-Konzept**

Designproblem Nr. 1?

- Verifikation i.d.R. nur durch *DNS-Server-Software*
- authoritative Nameserver verifiziert sich nicht selbst
  - ⇒ ohne Caching-Resolver in der eigenen Einrichtung i.d.R. keine Verifikation möglich
  - ⇒ **Caching-Resolver vs. ACL-View-Konzept**

⇒ Lösung ist Einsatz eines **Local** Resolvers



https://rz.uni-greifswald.de/support/dokumentation/anleitungen/dnssec/

Designproblem Nr. 2?

- Nutzung von DNSSec für **interne** Infrastruktur

Designproblem Nr. 2?

- Nutzung von DNSSec für **interne** Infrastruktur
- Ausfall der Internetanbindung

Designproblem Nr. 2?

- Nutzung von DNSSec für **interne** Infrastruktur
- Ausfall der Internetanbindung
- ... tick tack ...

Designproblem Nr. 2?

- Nutzung von DNSSec für **interne** Infrastruktur
- Ausfall der Internetanbindung
- ... tick tack ...
- Ablauf der DNSSec-Verifikationsantworten bis zur Root-Zone

Designproblem Nr. 2?

- Nutzung von DNSSec für **interne** Infrastruktur
- Ausfall der Internetanbindung
- ... tick tack ...
- Ablauf der DNSSec-Verifikationsantworten bis zur Root-Zone
- BOOM!

Designproblem Nr. 2?

- Nutzung von DNSSec für **interne** Infrastruktur
- Ausfall der Internetanbindung
- ... tick tack ...
- Ablauf der DNSSec-Verifikationsantworten bis zur Root-Zone
- BOOM!

$\Rightarrow$ Mögliche Lösung:
Umrouten der DNS-Server auf alternativen Internetuplink

Wie nutzt man DNSSec praktisch?

DANE-gesicherte Kommunikation mit einem Mail Transfer Agent

```
_25._tcp.mailgate1.uni-greifswald.de. 3600 IN TLSA 3 1 2 83B527FE9A84
33BFE8FA892C3738D59FA7A2437D1581318C9B17A57B 2593E525E28C77528E219949
389B28480A6553A03233ECDDE54782E1 EB4EC1087A444771
```

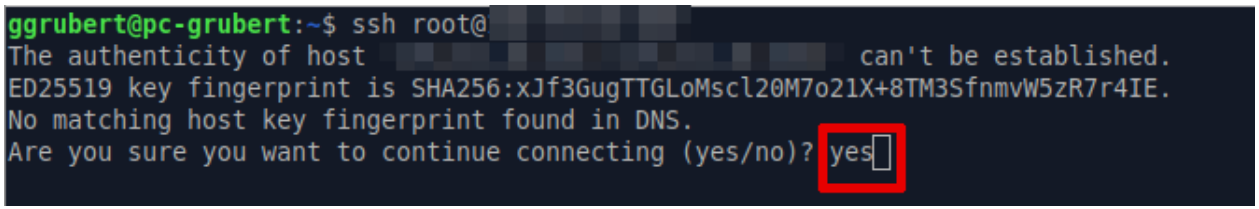
Via DNS werden alle relevanten Daten des Public Keys der Gegenstelle **sicher** als **TLSA Resource Record** mitgeteilt:

_25	Port
_tcp	Protokol
mailgate1...de.	FQDN
3	Zertifikat nur für angegebene Domain
1	nur „SubjectPublicKeyInfo“ wird gehashed
2	SHA-512 Hashwert
	Hashwert

https://thomas-leister.de/dane-tlsa-records-erklaert/

Wie nutzt man DNSSec praktisch?

SSH Fingerprints




```
mailgate1.uni-greifswald.de. 3600 IN        SSHFP        4 2 9D5769E3D495F1CEC
F6DCE3CBFC92F67918A2E1C2E613DFF8369C439 AEEA8E0F
```

Fingerprint des SSH Public Keys kann **sicher** als **SSHFP Resource Record** mitgeteilt werden:

4	Ed25519-Schlüssel
2	SHA-512 Hashwert
	Hashwert

https://de.wikipedia.org/w/index.php?title=SSHFP_Resource_Record&oldid=152571939

Hinweis: LibreSSL unterstützt dies nicht 😕

DNSSec verwenden (Client)

Auch hier könnte der Client Fehler machen!

Im Gegensatz zur PKI ist nur **ganz wenig** zu tun:

- DNSSec-fähigen Resolver einsetzen
- DNS-Anfrage bzgl. TLSA-Record durchführen

```
#dig -t TLSA _25._tcp.mailgate1.uni-greifswald.de +dnssec

# [...]
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

# [...]
;; ANSWER SECTION:
_25._tcp.mailgate1.uni-greifswald.de. 3591 IN TLSA 3 1 2 83B527FE9...
```

- püfen, ob das **ad-Flag** gesetzt ist

DNSSec verwenden (Client)

Was muss der Client hier **nicht** tun?

- das Zertifikat gegen eine CA prüfen
- OCSP-Anfragen stellen und prüfen
- CRL vorrätig haben und prüfen

DNSSec verwenden (Client)

Was muss der Client hier **nicht** tun?

- das Zertifikat gegen eine CA prüfen
- OCSP-Anfragen stellen und prüfen
- CRL vorrätig haben und prüfen

Fazit

Abgesehen von einer DNS-Abfrage kann der Client bei der Verwendung von DNSSec nichts mehr falsch machen.

Client-Beispiel aus der Praxis

Postfix-Konfiguration:

```
# [...]
smtp_dns_support_level  = dnssec
smtp_tls_security_level = dane
smtp_tls_policy_maps    = hash:/etc/postfix/tls_policy
# [...]
```

TLS-Policy:

```
# [...]
mailbox.org                     dane-only
# [...]
tuhh.de                         dane-only
tum.de                          dane-only
uni-erlangen.de                 dane-only
uni-kl.de                       dane-only
uni-muenchen.de                 dane-only
uni-passau.de                   dane-only
uni-regensburg.de               dane-only
uni-rostock.de                  dane-only
```

Client-Beispiel aus der Praxis

Postfix-Konfiguration:

```
# [...]
smtp_dns_support_level  = dnssec
smtp_tls_security_level = dane
smtp_tls_policy_maps    = hash:/etc/postfix/tls_policy
# [...]
```

TLS-Policy:

```
# [...]
mailbox.org                     dane-only
# [...]
tuhh.de                         dane-only
tum.de                          dane-only
uni-erlangen.de                 dane-only
uni-kl.de                       dane-only
uni-muenchen.de                 dane-only
uni-passau.de                   dane-only
uni-regensburg.de               dane-only
uni-rostock.de                  dane-only
```

Fazit

Keine PKI-Nebenbedingungen - einfach nur DNSSec & ad-Flag prüfen.

Allgemeine Betrachtungen

- man muss sich des Schutzniveaus seiner Systeme/Daten selbst bewusst sein
- wenn Verschlüsselung erforderlich ist, dann aber richtig

$\Rightarrow$ **schlechte Verlüsselung ist Klartext**

Serverseitige Lösungen

- Server muss Mindeststandards vorgeben, die **kein** Client unterbieten kann
- TLS Protokollversion muss mindestens **TLSv1.2** sein
- ausschließlich **starke Cipher Suiten** zulassen

$\Rightarrow$ direkte Auswirkung auf Liste kompatibler Clients

DNSSec als Lösung

- mit DNSSec nimmt man dem Client viel Arbeit/Verantwortung ab
- dafür handelt man sich andere Probleme ein:
  - Verifikation i.d.R. durch nächstgelegenen Caching Resolver
  - Ausfall der Internetanbindung
- die Welt wäre ein ganzes Stück sicherer, wenn TLSA Resource Records zur Standardimplementierung für TLS-Verbindungen gehören würden

Client-Verantwortung in der (PKI-) Praxis

- gesamte Verantwortung bzgl. der Verschlüsselungsqualität obligt dem Client
- problematisch sind Clients, auf die man keinen Einfluss hat

  (z.B. Supplikanten für 802.1x)
- Client muss im PKI-Umfeld sehr viel beachten

  (CA-Prüfung, CRL, OCSP)
- **Client kann auch ein Server sein**

  (in einer IT-Infrastruktur ist i.d.R. jeder Server auch ein TLS-Client)

Client-Verantwortung in der (PKI-) Praxis

- gesamte Verantwortung bzgl. der Verschlüsselungsqualität obligt dem Client
- problematisch sind Clients, auf die man keinen Einfluss hat

  (z.B. Supplikanten für 802.1x)
- Client muss im PKI-Umfeld sehr viel beachten

  (CA-Prüfung, CRL, OCSP)
- **Client kann auch ein Server sein**

  (in einer IT-Infrastruktur ist i.d.R. jeder Server auch ein TLS-Client)

Fazit

Verschlüsselung gehackt? Der Client ist schuld!

(betrachtet man Heartbleed, dann vielleicht nur „fast immer“ 🙂)