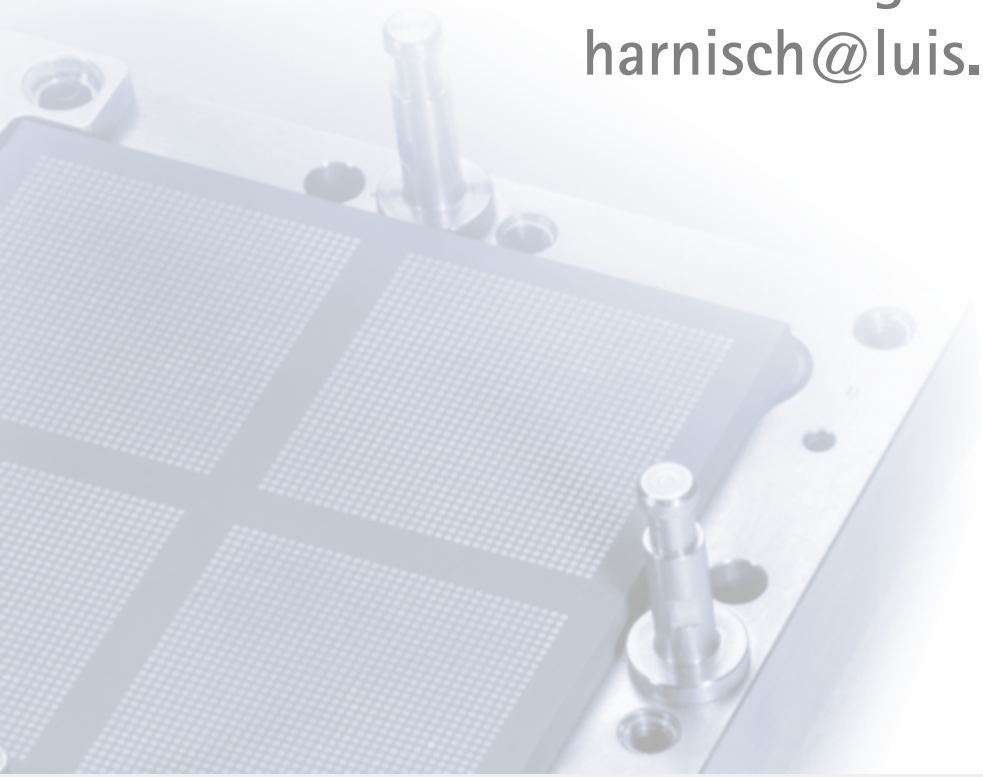
Leibniz Universität Hannover

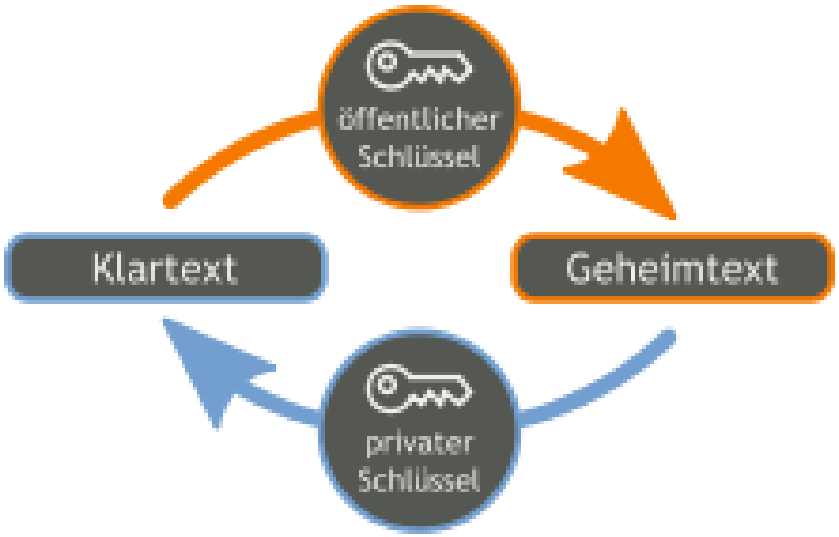
TLS-Verschlüsselung und Public-Key-Infrastructure

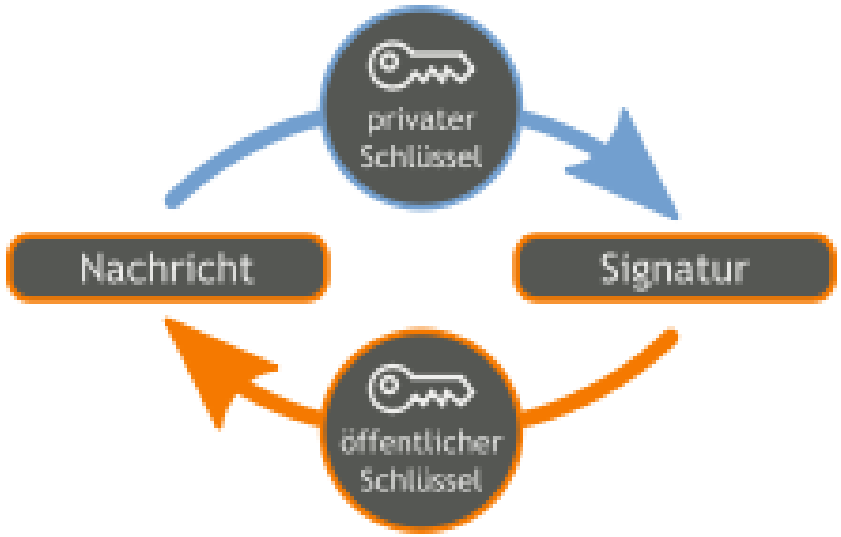
Hergen Harnisch
harnisch@luis.uni-hannover.de



asymmetrische Verschlüsselung

- Schlüsselpaar aus privatem und öffentlichem Schlüssel (asymmetrische Verschlüsselung)
  - mit öffentlichem Schlüssel Verschlüsseltes kann nur privater lesen
  - mit privatem Schlüssel Verschlüsseltes kann jeder mit öffentlichem Schlüssel lesen, erkennt dabei aber Besitzer des privaten als Urheber





PKI: Zertifikate & CA

- Zertifikat als Bestätigung der Zuordnung eines öffentlichen Schlüssels zu Nutzer-/Server-Daten
- Certificate-Authority (CA) als Notar (Zertifikatsaussteller): signiert und bestätigt damit die Zuordnung Schlüssel$\rightarrow$Nutzer
- CA-Kette: CA, die über Zertifikat Unter-CA bestätigt
- TLS bei `https`:
  Sessionkey wird über asymmetrische Verschlüsselung ausgehandelt, Server weist sich durch Zertifikat aus

Wissen über die Algorithmen, RSA o.ä. ist für den Vortrag nicht notwendig.

DFN-PKI:

- Umstellung Root-CA
  - Auswirkung WLAN
- CA für die LUH

DFN-PKI vs. Let's Encrypt

TLS:

- Handwerkszeug
- OCSP
- Webserver
- Nutzer-Zertifikate

DFN-PKI: Umstellung Root-CA

alte CA

- Root-CA "Deutsche Telekom Root CA 2"-Zertifikat läuft am 9.7.2019 aus
- Kette der CAs: Telekom $\rightarrow$ DFN $\rightarrow$ UH-CA ($\rightsquigarrow$ Nutzer/Server)
- jeder DFN-Teilnehmer hat eigene CA (dadurch DFN-PKI „riesig“)
- 965 gültige Zertifikate der UH-CA (Stand 28.1.19):
  - davon 210 Nutzer
  - und 622 Webserver

Alle diese Zertifikate laufen am 9.7. automatisch mit aus (Endedatum $\leq$ Root-Endedatum)

DFN-PKI: Umstellung Root-CA

neue CA

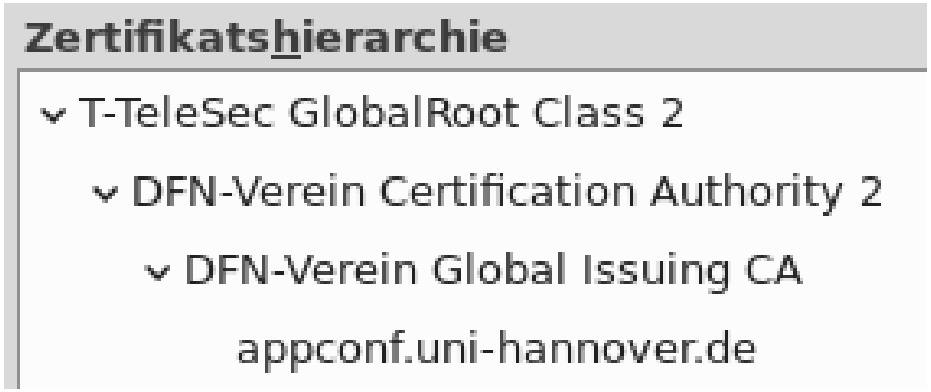
- Root-CA "T-TeleSec GlobalRoot Class 2" als Ablösung
- Kette der CAs: T-Telesec $\rightarrow$ DFN-Verein Certification Authority 2 $\rightarrow$ DFN-Verein Global Issuing CA ($\rightsquigarrow$ Nutzer-/Server-Zertifikat)
- DFN-Teilnehmer erhalten i.d.R. keine eigene CA (dadurch DFN-PKI „normal“)
- jeder DFN-Teilnehmer hat RA-Rolle (Teilnehmerservice) und wie bisher eigenen DN-Teilbaum
- auch neue Root-CA in gängigen Systemen verankert, aber
  - Android $\leq$ 4.4 kennt es nicht (obsolet?!)
  - Windows lädt dynamisch nach (Probleme SAP-Karte)

DFN-PKI: Umstellung Root-CA

CA-Hierarchie in der DFN-PKI

alt, bis max. 9.7.2019:

neu / aktuell:



- LUH/LUIS verliert CA, bleibt Teilnehmerservice (RA)
- Kette bleibt gleich lang (2x DFN)
- alle (benötigten) Zertifikate umzustellen

DFN-PKI: Umstellung Root-CA

Umstellung

- 2019: Schonfrist für Android 4.x ist vorbei
- neue Zertifikate rechtzeitig beantragen
- paralleles Halten zweier Zertifikate (alt+neu) ist erlaubt,
  z.B. altes nutzen, aber neues beantragen/testen/bereit legen
- Problem: Engpass beim Teilnehmer-Service zum Stichtag

Fragen Sie nicht erst am 10.7.!

- Sie erhalten automatisch Mails vor Ablauf durch DFN-PKI
- wir werden evtl. auch nochmal hinweisen
- Problem: wenn Mail-Adresse vom Antrag nicht mehr gilt
  (Server: geben Sie unbedingt Funktions-Adressen an)

DFN-PKI: Umstellung Root-CA

Umstellung praktisch

Server-Zertifikat

1. neuen Antrag mit neuen Schlüsseln (Anleitung vgl. LUIS-Seite), ggf. Generierung und Antrag direkt im LUIS durch RA
2. Antrag, ggf. Akkreditierungsschreiben zu LUIS $\rightsquigarrow$ neues Zertifikat
3. Root-CA ist im Betriebssystem bekannt,
   aber CA-Kette + Server-Zertifikat + privaten Schlüssel einspielen,
   Angaben dazu in Config ändern (Apache s.u.)
4. ggf. Dienst neu starten
5. Testen, auch auf Kette (s.u.)

Nutzer-Zertifikat? mehrere parallel völlig unspektakulär

DFN-PKI: Umstellung Root-CA

Problem WLAN etc.

Probleme dort zu erwarten, wo

- eine Root-CA fest ausgewählt ist,
- Server-Zertifikat „gepinnt“ ist.

Root-CA ist bei vernünftigem Eduroam-/LUHWPA-Setup fest eingetragen, derzeit noch alte Root-CA. Parallelbetrieb nur mit Kunstgriff möglich.

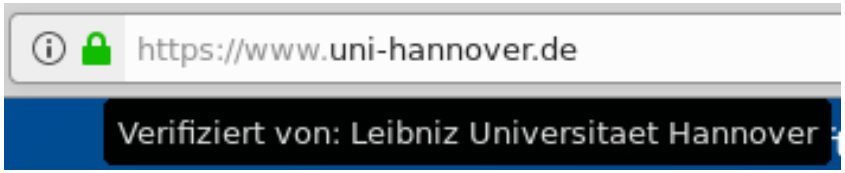
$\Longrightarrow$ demnächst Tool & Anleitungen für neue CA

$\Longrightarrow$ Zertifikatsangabe sehr wichtig,
sonst MitM-Angriffe und Credential-Abgriff

$\Longrightarrow$ bei Umstellung Nutzerzugänge auf IdM umstellen,
alte BIAS-Accounts bitte dann löschen

DFN-PKI vs. Let's Encrypt

DFN-PKI



- längere Laufzeit
- manuelle Ausstellung
- Prüfung Org.-Zugehörigkeit
- ggf. Ausweisprüfung
- Fortgeschrittene Signatur

Let's Encrypt

- kurze Laufzeiten
- automatisierte Ausstellung
- Prüfung nur auf Domain-Besitz
  - via DNS-
  - oder HTTP-Challenge

DFN-PKI hat stärkere Aussagekraft.

Aber: Nutzer / Empfänger sieht das nicht.

DFN-PKI vs. Let's Encrypt

DFN-PKI für die LUH

Probleme mit anderen CAs:

- schwache Prüfung der Berechtigung
- IP-Zugang in Universität leicht erreichbar
- Sicherheitsproblem durch „gefälschte" Zertifikate
- besonders durch *-Zertifikate

Maßnahme in Planung

Festlegung auf die DFN-PKI, technisch durch CAA:

```
uni-h.de. 86400 CAA 0 issue "pki.dfn.de"
uni-h.de. 86400 CAA 0 issuewild ";"
uni-h.de. 86400 CAA 0 iodef "mailto:security@uni-hannover.de"
```

grundsätzlich für alle Domains der LUH

TLS: Handwerkszeug

Dateien

mehrere Dateien gehören dazu:

- Private-Key (meist PEM, ggf. PKCS#8)
- Zertifikatsantrag (CSR=PKCS#10), enthält Public-Key
- Zertifikat, enthält Public-Key (und CA-Signatur)
- Paket aus Private-Key und Zertifikat: PKCS#12
- CA-Kette: Sammlung von CA-Zertifikaten
- CRL: Liste von zurückgerufenen Zertifikaten (je CA)
- (PKCS#7: Signatur an E-Mails)

Alles, was den Private-Key enthält, ist

- mit Passphrase zu sichern oder
- durch Dateiberechtigungen im Zugriff einzuschränken

TLS: Handwerkszeug

Formate

Einiges in 2 Formaten möglich:

PEM Base-64 kodiert, also Text
unter Linux bevorzugt, CA-Kette nur als PEM

DER binäres Format
eher bei Windows, nur ein Schlüssel je Datei

Problem:

- keine festen Endungen etabliert
- z.T. gleiche Endung für DER und PEM üblich

Alle CAs, Ketten und CRLs (aktuell) von uns zusammen getragen:
`https://appconf.uni-hannover.de/conf/pki/`

TLS: Handwerkszeug

Ablageorte auf Servern

Ort für CA-Zertifikate auf Debian-Servern

- `/etc/ssl/certs`: enthält nur Links, *nicht* Dateien
- `/usr/share/ca-certificates`: vom System eingespielte CAs
- eigne CAs $\longrightarrow$ `/usr/local/share/ca-certificates`, z.B. `/usr/local/share/ca-certificates/dfn/dfnglobal2-dfnca.crt`
- Endung muss `.crt` sein, Dateien im PEM-Format
- `update-ca-certificates` erzeugt Links in `/etc/ssl/certs` (mit Hash-Namen für schnelles Finden bei Prüfung)

TLS: Handwerkszeug

Ablageorte auf Servern

Rest auf Debian-Servern

- privater Schlüssel als PEM mit `ug=r,o=` o.ä. als `/etc/ssl/private/KEYNAME.key`
- CA-Kette und Serverzertifikat als PEM, Ablageort nicht ganz klar (`/etc/ssl` ist naheliegend)
- CRL werden höchstens bei Prüfung Client-Zertifikat benötigt, sind dann täglich durch cron-Job zu aktualisieren

Windows:

- unbedingt beachten: Certificate-Store des Computers, nicht des Nutzers (z.B. über MMC Konsole definieren)
- DER-Formate einspielen, Zertifikat & Schlüssel als PKCS#12

TLS: Handwerkszeug

Wandeln mit OpenSSL

OpenSSL hat als 2. Parameter Kommando, dazu Manpage (`man x509`)

- Zertifikatsrequest ansehen

  ```
  openssl req -in CSR.pem -text
  ```

- Zertifikat von PEM nach DER

  ```
  openssl x509 -in CERT.pem -out CERT.der -outform der
  ```

- PKCS#12 bauen (für Windows):
  - PKCS#12 mit Serverzertif. & privatem Schlüssel

    ```
    openssl pkcs12 -export -in CERT.pem -inkey KEY.pem -out SERVER.p12
    ```

  - PKCS#12 dabei mit CAs aus Kette anreichern

    ```
    ... -certfile dfnglobal2-chain.pem
    ```

TLS: OCSP

Leibniz Universität Hannover

Rückruf eines Zertifikats wird durch CA signiert.

- regelmäßige Publikation in CRLs
- Alternativ Betrieb von OCSP:
  - erhält Anfrage nach einzelnen Zertifikaten via http
  - prüft Sperrstatus und gibt das zurück
- Informationsquelle steht in Zertifikaten
- Client prüft, eigentlich Kette durchzuprüfen

Vergleich:

+ OCSP ist meist aktueller
- extra Anfrage (auch Privacy-Leak)
- häufiger wird fehlende OCSP-Antwort als okay gewertet
+ CRL-Aktualisierung meist nicht implementiert

TLS: OCSP

OCSP-Stapling

Normalerweise muss Client Server-Zertifikat bei OCSP-Dienst prüfen.
Stapling: Server holt regelmäßig OCSP-Bestätigung, gibt die an Client

$\longrightarrow$ zentral 1x geholt für viele Anfragen

$\longrightarrow$ kein Privacy-Problem für den Client

Apache

```
# OCSP-Stapling
SSLUseStapling on
SSLStaplingCache
shmcb:${APACHE_RUN_DIR}/stapling_cache(128000)
# PROBLEM: Stapling geht wohl nicht ueber Proxy ...
```

SSLOCSP* bezieht sich auf Prüfung von Client-Zertifikaten

TLS: Webserver

Konfiguration (Apache)

für das Verständnis der Dokumentation wichtig zu unterscheiden:

- Serverzertifikat für https-Angebot
- Prüfung von Client-Zertifikaten als Authentifizierung

Apache (ohne Client-Zertifikate)

```
# SSL-Einstellungen:
SSLEngine on
SSLCertificateFile /etc/ssl/certs/SERVER.crt
SSLCertificateKeyFile /etc/ssl/private/SERVER.key
SSLCertificateChainFile
    /usr/local/share/ca-certificates/dfnglobal2-chain.pem
```

TLS: Webserver

Heutzutage sind Einstellungen für Algorithmen nötig

- veraltete Protokolle wie SSLv3 nicht mehr anbieten
- veraltete Algorithmen nicht mehr anbieten
- Perfect-Forward-Secrecy wird erwartet
- machmal Kompromisse für alte Clients nötig, dann aber als letztes Mittel (Apache: `SSLHonorCipherOrder On`)
- kann sich immer mal ändern …
- Anleitungen bei BetterCrypto, Mozilla, SSLLabs

Bei Apache z.B. für alle Sites in `/etc/apache2/mods-enabled/ssl.conf`, `bettercrypto.org` enthält Konfigurationsbeispiele.

TLS: Webserver

Test der Einstellungen

Prüfung der Einstellung unbedingt vornehmen

- Kette muss unbedingt dabei sein,
  fällt aber bei Uni-Clients meist nicht auf
- Verschlüsselung sollte nicht ganz mies sein

Einfachste Prüfung mit Web-Dienst
`https://www.ssllabs.com/ssltest/`

- man muss nicht A+ erreichen
- Kompatibilität kann wichtiger sein
- meist ist TLS nicht die größte Angriffsfläche …

TLS: Webserver

Test interner/anderer Server

Mit lokalen Tools:

- Mailserver auf SSL-Port

```
openssl s_client -connect mailgate.uni-hannover.de:465
```

- Mailserver mit StartTLS

```
openssl s_client -connect mailgate.uni-hannover.de:25
                                                -starttls smtp
```

- Webserver (mit SNI)

```
openssl s_client -connect www.uni-hannover.de:443
                                    -servername www.uni-hannover.de
```

Verbose oder Debug von socat oder telnet-ssl liefern Ähnliches,
Test der Verschlüsselung etc. benötigt extra Tool.

TLS: Nutzer-Zertifikate

Mail

Für Mail können nur Nutzerzertifikate verwendet werden.

- zu einer Person für persönliche Mail-Adressen
- als Gruppen-Zertifikate für Funktionsadressen
- für Server-Systeme als Gruppen-Zertifikat

\+ als Signatur gut für
  - Nutzer-Mail und
  - Server-/System-Mail (technisch selten möglich)

o Verschlüsselung längst nicht immer
  + super für Passwort-Übermittlung
  - für dauerhaften Zugriff / Archivierung

TLS: Nutzer-Zertifikate

Web-Client

Ein Client-Zertifikat kann bei TLS-Verbindung zu einem Webserver genutzt werden (Server muss das anfordern, Browser wählt).

- dient der Authentifizierung eines Nutzers oder Rechners
- keine Aussage zur Authorization o.ä.

In der LUH im Einsatz für Spezialitäten

- Klientenverwaltung ptb
- Keyuser-Login für SAP-Kartenantragsserver
- E-Books der LUIS/RRZN-Handbücher

und (empfehlenswert!) zwischen Servern (mit 2 Server-Zertifikaten)