IT-Sicherheit

Anwendungen Kryptographischer Protokolle (Teil 2)

Prof. Dr. Dominik Merli, Prof. Dr. Lothar Braun

Sommersemester 2020

Hochschule Augsburg - Fakultät für Informatik

Public Key Infrastructure (PKI)

X.509 Struktur

- Certificate
  - Version Number
  - Serial Number
  - Signature Algorithm ID
  - Issuer Name
  - Validity period
    - Not Before
    - Not After
  - Subject name
  - Subject Public Key Info
    - Public Key Algorithm
    - Subject Public Key
  - Issuer Unique Identifier (optional)
  - Subject Unique Identifier (optional)
  - Extensions (optional)
- Certificate Signature Algorithm
- Certificate Signature

QUIZ: Zertifikate (twbk.de)

Was ist eine wichtige Aufgabe von Zertifikaten?

A) Zertifikate enthalten Informationen über private Schlüssel und erlauben den sicheren Austausch der privaten Schlüssel

B) Zertifikate enthalten Informationen über die Identität und öffentlichen Schlüssel eines Teilnehmers und erlauben es die Integrität und Authentizität zu verifizieren

C) Zertifikate bilden eine Public Key Infrastruktur

Unter welchen Umständen wird ein Zertifikat als *vertrauenswürdig* angesehen?

A) Wenn es von einer vertrauenswürdigen CA unterschrieben wurde

B) Wenn es zu dem Zertifikat eine Zertifikats-Kette gibt, deren oberstes Zertifikat ein Wurzel-Zertifikat ist

C) Wenn die Anwendung dem Zertifikat vertraut

Verwendung von X.509 PKIs in der Praxis

- **Web PKI**
  - Absicherung von Webseiten mittels HTTPS
  - CA/Browser Forum → Public Keys in Browsern
- **Code Signing**
  - Ausführen von Software aus vertrauenswürdigen Quellen
- **Secure / Multipurpose Internet Mail Extensions (S/MIME)**
  - Verschlüsselung und Signaturen für Emails
- **Absicherung beliebiger Protokolle**
  - Beispiele: IMAPS, IPSec, OpenVPN, ...
- **Frage: Wie Verwendungszweck und Vertrauen von Zertifikaten bestimmen?**

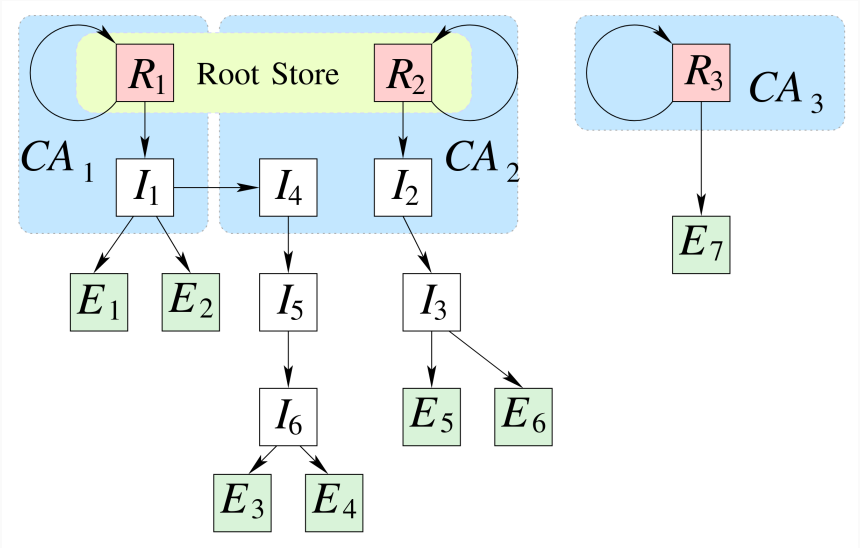
Verwendungszweck von Zertifikaten

- **Zertifikat enthält Informationen über Verwendungszweck**
  - *Key Usage* definiert Verwendungszweck, z.B. signieren anderer Zertifikate
  - *Extended Key Usage* definieren Rechte (z.B. Code Signing, Email Protection, ...)

- **Zertifikat enthält Informationen über Gültigkeit für Identität**
  - *Common Name* enthält z.B. Informationen über Domain
  - *Subject Alternative Name* kann weitere Domain-Namen enthalten

- **Anwendung muss Gültigkeit von Zertifikaten prüfen**
  - Prüfung der Signaturen des Zertifikats (meistens TLS-Bibliothek)
  - Prüfung der Gültigkeitsdauer des Zertifikats (oft TLS-Bibliothek)
  - Prüfung von Zertifikats-Kette und *Key Usage* (TLS-Bibliothek und Anwendung)
  - Prüfung der Plausibilität der Identität (immer Anwendung)

Prüfung von Zertifikats-Ketten



Relevante Faktoren für Prüfung von Zertifikats-Ketten

- Root Store
  - Definiert die Menge an CAs die vertrauenswürdig sind
  - Vertrauen: Eine Kette zwischen End-Zertifikat und Root-Zertifikat existiert
  - Unterschiedliche Anwendungen können unterschiedliche *Root Stores* verwenden
- Intermediate CAs
  - CAs die nicht im *Root Store* enthalten sind, die aber eine Zertifikats-Kette zu einem Zertifikat im *Root Store* haben
- Certificate Chain
  - Anwendung kennt nur Zertifikate im *Root Store*
  - Alle Intermediate-Zertifikate *müssen* im TLS-Handshake mitgeliefert werden

Alternative Wege um Zertifikate im Client zu Prüfen

- *Certificate Pinning*
  - Client verifiziert Zertifikat nur hinsichtlich korrekter Identität
  - Zertifikat wird nur als valide anerkannt, wenn Hash von Zertifikat bekannt

- Trust-on-First Use (TOFU)
  - Beim ersten Kontakt: Nutzer wird zur manuellen Prüfung aufgefordert
  - Hash von Zertifikat wird nach erfolgreicher Prüfung gespeichert
  - Findet auch in anderen Protokollen Anwendung: z.B. SSH

- Listen vertrauenswürdiger Zertifikate im Programm-Code
  - Programmierer codiert Liste vertrauenswürdiger Zertifikate
  - Änderungen an Zertifikaten benötigen Update des Programms
  - Beispiele: Mobile Apps, Chrome Browser, ...

Prozesse: Erstellen von Zertifikaten

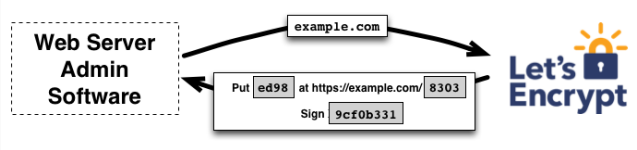
- CAs können Zertifikate für beliebige Identitäten ausstellen
  - Regeln und Prozesse müssen zur Prüfung der Identitäten definiert werden
  - Prüfung der Einhaltung der Regeln durch Mitglieder der PKI
- Domain Validation (DV)
  - Nachweis über Besitz einer Domain durch Antragsteller des Zertifikats
  - Möglichkeiten:
    - CA schickt Email an *Administrator-Email*, z.B. admin@domain.com
    - Antragsteller veröffentlicht *TXT Records* im DNS Protokoll für Domain
- Extended Validation (EV)
  - CA verlangt detaillierte Nachweise über Identität des Antragsstellers
- Organization Validation (OV)
  - Prüfungsaufwand zwischen DV und EV. Weniger Dokumentation notwendig

ACME-Protokoll

- **ACME: Automated Certificate Management Environment**
  - Automatisierung der Domain-Validierung für DV-Zertifikate
  - Wird z.B. von *Let's Encrypt* eingesetzt
  - JSON-basiertes Protokoll als IETF-Standard veröffentlicht: RFC 8555
    https://tools.ietf.org/html/rfc8555

- **Ablauf der Ausstellung von Zertifikaten**
  - ACME-Client generiert *Account Key* (asymmetrisches Schlüsselpaar)
    - Schickt öffentlichen Schlüssel, Terms-of-Use Agreement und Kontakt-Informationen an Registrar
  - Validierung der Kontrolle einer Domain / Hostnamen durch *Account Key*
  - Anfrage von Server-Zertifikaten mittels *Account Key*

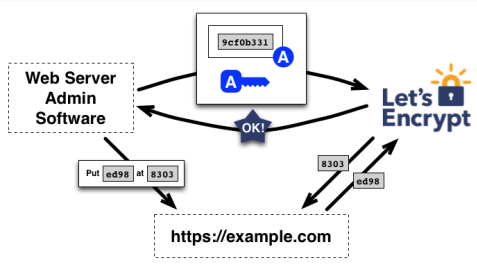
ACME: Validierung von Kontrolle über Domain mittels HTTP



Quelle: https://letsencrypt.org/de/how-it-works/

- Inhaber des *Account Key* fordert Account-Verifizierung für Domain an
- Zertifizierungsstelle fordert Beweis über Kontrolle der Domain und Schlüssel
  - Antragsteller soll *Challenge* auf Web-Server online stellen
  - Antragsteller soll *Nonce* mit Schlüssel signieren

ACME: Validierung von Kontrolle über Domain mittels HTTP



Quelle: https://letsencrypt.org/de/how-it-works/

- Antragsteller sendet signierte *Nonce* zurück
- Zertifizierungsstelle prüft ob *Challenge* online ist
  - Typische URL:
    http://example.com/.well-known/acme-challenge/AlotOfRandomChars

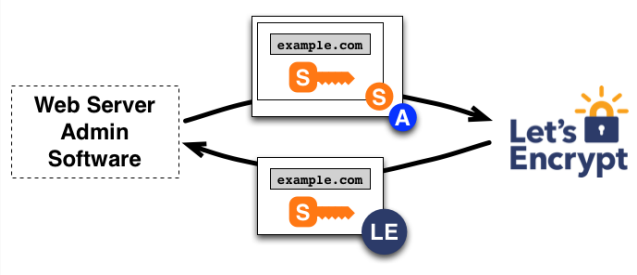
ACME: Validierung mittels DNS

- CA bittet *Account Key* Besitzer um veröffentlichen eines DNS-Eintrages mittels Token
  - Eintrag abhängig von Unique Key, Domain und *Account Key*
  - ACME-Client veröffentlicht TXT Record, CA fragt Token an

```
_acme-challenge.example.org. 300 IN TXT "gfj9Xq...Rg85nM"
```

- Nachdem CA HTTP oder DNS-Validierung durchgeführt hat, kann Account neue Zertifikate beantragen
  - Bei HTTP-Validierung: Nur Zertifikate für diesen Host
  - Bei DNS-Validierung: Wild-Card Zertifikate (z.B. *.example.com)

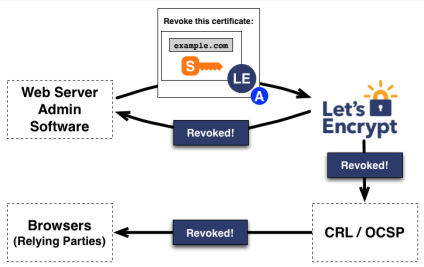
ACME: Zertifikate ausstellen



Quelle: https://letsencrypt.org/de/how-it-works/

- ACME-Client generiert CSR für Domain mit gewünschtem Public-Key
  - CSR wird mit private Key für Zertifikat signiert
  - CSR with mit *Account Key* für Domain signiert
- CA stellt Zertifikat aus wenn beide Signaturen korrekt sind

ACME: Zertifikate zurückrufen



Quelle: https://letsencrypt.org/de/how-it-works/

- ACME-Client stellt Antrag auf Rücknahme von Zertifikat
  - Anfrage muss mit *Account Key* signiert sein
- CA fügt Zertifikat Liste der zurückgezogenen Zertifikate ein
  - Abfrage durch Clients mittels Abfrage der CRL-Liste oder *OCSP* möglich

Certificate Transparency

- **Problem: CAs können Zertifikate für beliebige Identitäten ausstellen**
  - Ob Zertifikat ausgestellt wurde wissen nur CA und Antragssteller
  - In der Vergangenheit häufiger Probleme mit falsch ausgestellten Zertifikaten

- **Lösungsansatz: Öffentliches *Log-Buch* aller jemals ausgestellten Zertifikate**
  - CAs registrieren jedes Zertifikat in einem *append-only* Speicher
  - Log-Daten sind für jedermann öffentlich einsehbar und überprüfbar
  - Webseite zur Überprüfung von Log-Daten: https://crt.sh/

- **Zertifikate bekommen *SCT-Extension* die Eintrag in Log nachweist**
  - Browser wie Chrome verlangen SCT-Einträge in Zertifikaten ab April 2018
  - Warnung falls neue Zertifikate keine SCT-Einträge

Certificate Transparency

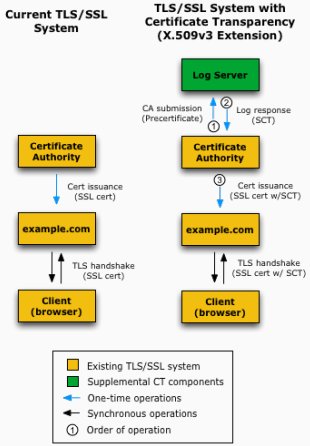


Figure 1

Erstellung von Zertifikaten (Quelle)

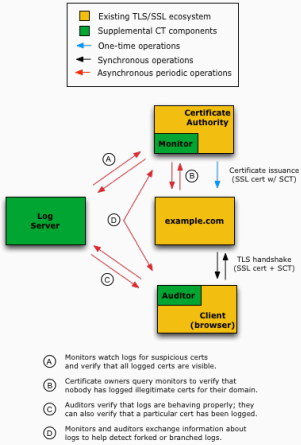


Figure 3

Prüfung von Zertifikaten (Quelle)

Fazit: Public Key Infrastruktur

- **Ausstellen und Prüfen von Zertifikaten in einer PKI benötigt Prozesse**
  - Prozesse nicht in technischer Spezifikation von X.509 enthalten
  - Einhaltung muss unabhängig von der Verwendung von X.509 kontrolliert werden

- **Prüfung der richtigen Benutzung von Zertifikaten muss durch Anwendung geschehen**
  - Technische Validierung von Signaturen Bibliotheken notwendig
  - Wichtig: Anwendung muss prüfen ob Zertifikat Anwendungszweck entspricht und ob Identität auch gewünschte Identität ist

Email-Sicherheit

Kommunikation über Email

- Email als unverschlüsseltes und nicht-authentifiziertes Klartext-Protokoll
  - Spezifizierung als Internet-Standard: RFC 822
  - Einsatz seit Jahrzehnten als universelles Medium zur Kommunikation

```
Date: Mon, 25 Mai 2020 08:00:00 (GMT)
From: lothar.braun@hs-augsburg.de
Subject: Email-Nachricht
To: lothar.braun@hs-augsburg.de

Dies ist eine Email-Nachricht.
```

Sicherung der Kommunikation

- **Absicherung des Transports der Nachrichten zwischen Email-Clients**
  - Emails werden über mehrere Protokolle übertragen
  - POP / IMAP: Abrufen von Emails von Email-Server durch Email-Client
  - SMTP: Versenden von Emails durch Clients oder zwischen Email-Servern
  - Verschlüsselung des Transports durch TLS-Verbindungen

- **Verschlüsselung der Emails mit S/MIME oder PGP**
  - Text der Email wird verschlüsselt und/oder signiert
  - S/MIME (Secure / Multipurpose Internet Mail Extensions): Basierend auf X.509-Zertifikaten und PKIs
  - PGP (Pretty Good Privacy): Public-Key Verfahren mit *Web-of-Trust*

S/MIME: Secure Multipurpose Internet Mail Extensions

- MIME: Datenformat zur Übertragung beliebiger Binär-Daten via ASCII
  - Text-Daten werden häufig mittels Quoted-Printable-Kodierung
  - Binär-Daten werden im Base64-System codiert

- S/MIME: Inhalte werden verschlüsselt und/oder signiert als MIME-Objekte versendet
  - Mehrere MIME-Typen existieren: z.B. für Verschlüsselung und Signatur
  - S/MIME-Daten werden im Base64-Format codiert

Beispiel für MIME-Codierung

```
...
To: lothar.braun@hs-augsburg.de
MIME-Version: 1.0
Content-Type: multipart/mixed;boundary="-----=01BDCC1E.A4D02412"

-------=00001BDCC1E.A4D02412
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable

Ein deutschsprachiger Text wurde hier als quoted-printable codiert,
um m=F6glichst viel Text f=FCr nicht-MIME-Clients lesbar zu halten.

-------=00001BDCC1E.A4D02412
Content-Type: application/pdf; name="Folien.pdf"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
filename="Folien.pdf"

0M8R4KGxGuEAAA ... lcnNpb24gWA1NYWNpbnRvc2gNU2VydmVyc3lzdGVtZQ1O
-------=00001BDCC1E.A4D02412--
```

Beispiel für S/MIME Verschlüsselte Nachricht

```
From: Lothar Braun <lothar.braun@hs-augsburg.de>
To: "lothar.braun@hs-augsburg.de" <lothar.braun@hs-augsburg.de>
Date: Sun, 24 May 2020 16:47:05 +0200
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/pkcs7-mime; name="smime.p7m"; smime-type="enveloped-data"
Content-Description: S/MIME Encrypted Message
Content-Transfer-Encoding: base64
MIME-Version: 1.0

MIAGCSqGSIb3DQEHA6CAMIACAQAxggG7MIIBtwIBADCBnjCBjTELMAkGA1UEBhMCREUxRTBDBgNV
BAoMPFZlcmVpbiB6dXIgRm9lcmRlcnVuZyBlaW5lcyBEZXV0c2NoZW4gRm9yc2NodW5nc25ldHpl
cyBlLiBWLjEQMA4GA1UECwwHREZOLVBLSTElMCMGA1UEAwwcREZOLVZlcmVpbiBHbG9iYWwgSXNz
...
ELWCEG7ogaptCs7Q6JSxOcUAAAAAAAAAAAAA
```

Informationen zur Verschlüsselung / Signatur

- Relevante Informationen über eingesetzte Verfahren sind in S/MIME Objekt hinterlegt

- Informationen über Verschlüsselung
  - Informationen über Empfänger (Identifikation des Zertifikats)
  - Verfahren zur symmetrischen Verschlüsselung
  - Symmetrischer Schlüssel (mit öffentlichen Schlüssel der Empfänger verschlüsselt)
  - Verfahren zur Entschlüsselung des symmetrischen Schlüssels

- Informationen über Signatur
  - Verwendeter Hash-Algorithmus
  - Zertifikats-Kette zur Identifikation des Senders
  - End-Zertifikat welches öffentlichen Schlüssel zur Validierung der Signatur enthält

Verwendete Kryptographische Verfahren

- Mehrere Versionen von S/MIME standardisiert
  - Version 2.0 - RFC 2311 (1998)
  - Version 3.0 - RFC 2633 (1999)
  - Version 3.1 - RFC 3851 (2004)
  - Version 3.2 - RFC 5751 (2010)
  - Aktuell: Version 4.0 - RFC 8551 (2019)
  - Wesentliche Änderungen: Einführung neuer kryptographischer Verfahren

- Verfahren die für S/MIME 4.0 unterstützt werden *müssen*
  - *Hash-Funktionen:* SHA-256 und SHA-512
  - *Symmetrische Verschlüsselung:* AES-128 GCM, AES-256 GCM, AES-128-CBC
  - *Verschlüsselung des symmetrischen Schlüssels:* ECDH mit zwei Kurven, RSA
  - *Algorithmen für Signatur:* ECDSA mit curve P-256 und SHA-256, EdDSA mit curve25519, RSA mit SHA-256

Prüfung vor Ausstellung eines Zertifikats

- **Identität des Inhabers einer Email-Adresse muss geprüft werden**
  - Verfahren zur Prüfung abhängig von Prozessen der CA
  - Minimale Prüfung: Antragsteller kann Emails auf Adresse empfangen
  - Maximale Prüfung: Persönliches Erscheinen mit Vorlage von Ausweiskopien

- **Die Hochschule Augsburg betreibt Zertifizierungsstelle**
  - Mitglieder der Hochschule können Zertifikate für die Verwendung mit S/MIME erhalten
  - Generierung der Schlüssel über Lösung im Browser
  - Prüfung der Identität der Personen durch Identifikation mit Ausweis

Aufgabe

Informieren Sie sich über die Möglichkeit ein S/MIME-Zertifikat für Ihre Hochschul-Adresse zu bekommen

https://www.hs-augsburg.de/Rechenzentrum/ZertifizierungCA.html

Gibt es noch Fragen?