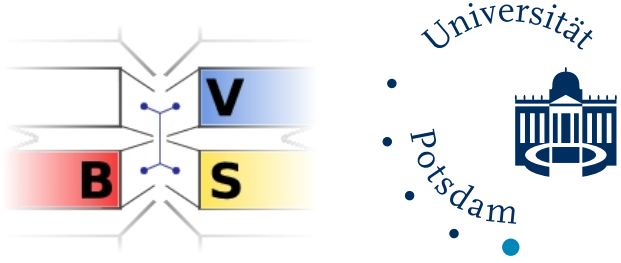
Securing SUIT-conform Firmware Update Management in the IoT with DNSSec/Dane

Max Schrötter, Wolf-Jörgen Stange, **Bettina Schnor**



FGSN‘22, 15.9.2022

BSI-Empfehlung

Bundesamt für Sicherheit in der Informationstechnik (BSI). BSI Grundschutz, SYS.4.4: Allgemeines IoT-Gerät, 2020:

A1 The devices SHALL have update functions. The manufacturer SHALL offer an update process.

A3 If security vulnerabilities are identified, they SHALL be fixed as soon as possible. ... In general, care MUST be taken to obtain patches and updates from **trusted** sources only.

A. Langiu, C. A. Boano, M. Schuß, K. Römer: *UpKit: An Open-Source, Portable, and Lightweight Update Framework for Constrained IoT Devices*, 2019

Grundidee: das IoT-Gerät sollte sich mit dem Update nur beschäftigen, wenn

- es aus einer vertrauensvollen Quelle stammt (Authenticated firmware)
- aktueller als die laufende Firmware ist (Freshness of the firmware)

$\Longrightarrow$ **Two-step approach: Manifest**

1. zuerst das Manifest mit signierten Metadaten validieren
2. dann erst das Update runterladen und installieren

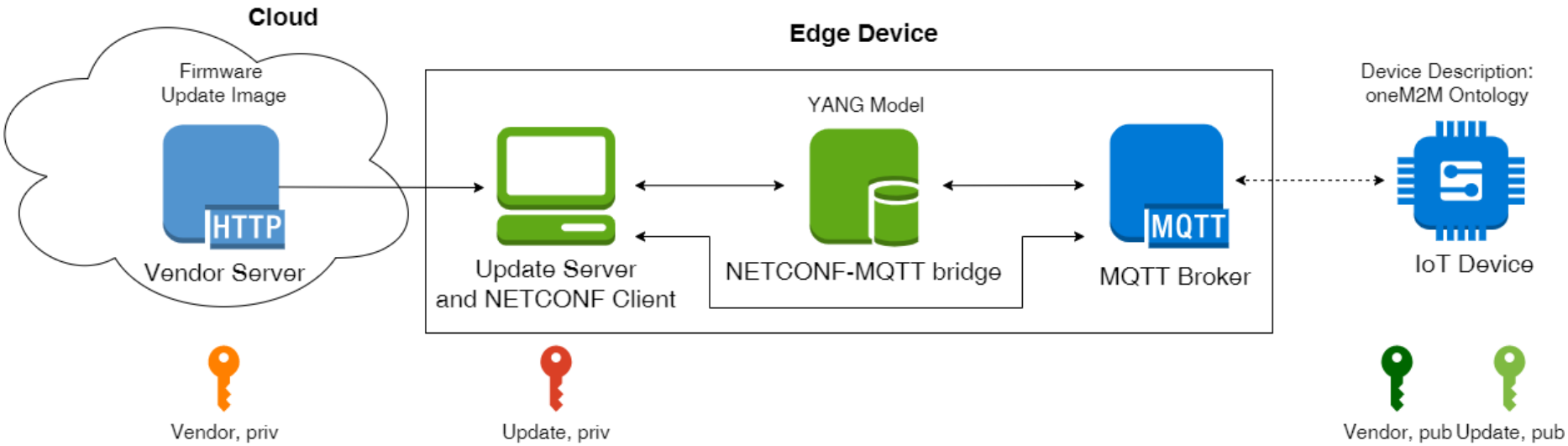
vorgeschlagen schon auf der CCS'10: TUF (The Update Framework)
für MYNO adaptiert: MYNO Update Protocol (MUP) (Sahlmann et al., Journal Sensors 2020)

Manifest

Vendor Manifest	
Field	**Description**
App ID	unique id for application
Link offset	memory address
Digest	hash value of the firmware
Size	size of the firmware in bytes
New Version	new firmware version
Old version	old firmware version
Inner signature	vendor signature
Manifest Extension	
Device UUID	unique device ID
Nonce	nonce generated by device
Outer signature	Update Server signature

Sahlmann et al.: MUP: Simplifying Secure Over-The-Air Update with MQTT for Constrained IoT Devices, Journal Sensors 2020)[1]

MUP Architektur



Wann vertraue ich dem öffentlichen Vendorschlüssel?

Bisher: Vorinstalliertes Zertifikat

MUP-Paper: *"An improvement of the MUP update protocol will be the extension of the key roll-over of the public vendor key using DNSSec/DANE."*

Kompromittierte CAs

Nvidias geleakte Code-Signing-Zertifikate missbraucht

Die Einbrecher haben bei Nvidia auch Code-Signing-Zertifikate entwendet und veröffentlicht. Mit denen werden nun Angriffs-Tools signiert.

Quelle: heise, März 2022

Wer aktualisiert den Vertrauensanker auf Consumer IoT-Geräten?

Studie auf der IMC'21 belegt:
"devices rarely remove deprecated and distrusted CA certificates from their root stores."[2]

Überblick

- DNSSEC/DANE
- Mittels DNSSEC/DANE abgesicherte TLS-Verbindung zum Vendor Server
- Validieren des Vendor Manifests mittels Code Signing Keys
- Vendor Key Rollover Protocol
- Implementierung

Absicherung mittels DNSSec/DANE

MUP benutzt:

1. **Zertifikat** für den Aufbau verschlüsselter Verbindungen zwischen Vendor und Update Server
2. **Code Signing Key**, mit dem das Manifest und die Firmware signiert wurde.

Absicherung mittels:

1. **DNSSec**: signierte DNS Records
2. **DANE: DNS-based Authentication of Named Entities (RFC 6698)**

DANE: DNS-based Authentication of Named Entities

Motivation (RFC 6698 (2012)): *TLS*[1] *uses certificates to bind keys and names.* **...** *The public CA model upon which TLS has depended is fundamentally vulnerable because it allows* ***any*** *of these CAs to issue a certificate for* ***any*** *domain name. A single trusted CA that betrays its trust, either voluntarily or by providing less-than-vigorous protection for its secrets and capabilities, can undermine the security offered by any certificates employed with TLS. This problem arises because a compromised CA can issue a replacement certificate that contains a fake key.*

Idee: *Keys are tied to names in the Domain Name System.*
$\Longrightarrow$ **TLSA Record:** Enthält den öffentlichen Schlüssel oder den Hashwert des öffentlichen Schlüssels

[1] Gilt analog für DTLS

Anwendungsmöglichkeiten:

1. Ergänzung zur PKI:
   Bei der Namensauflösung wird ein signierter Hashwert des zugehörigen Public Keys mitgeschickt.

   Most significantly, the keys associated with a domain name can only be signed by a key associated with the parent of that domain name; for example, the keys for "example.com" can only be signed by the keys for "com", and the keys for "com" can only be signed by the DNS root.

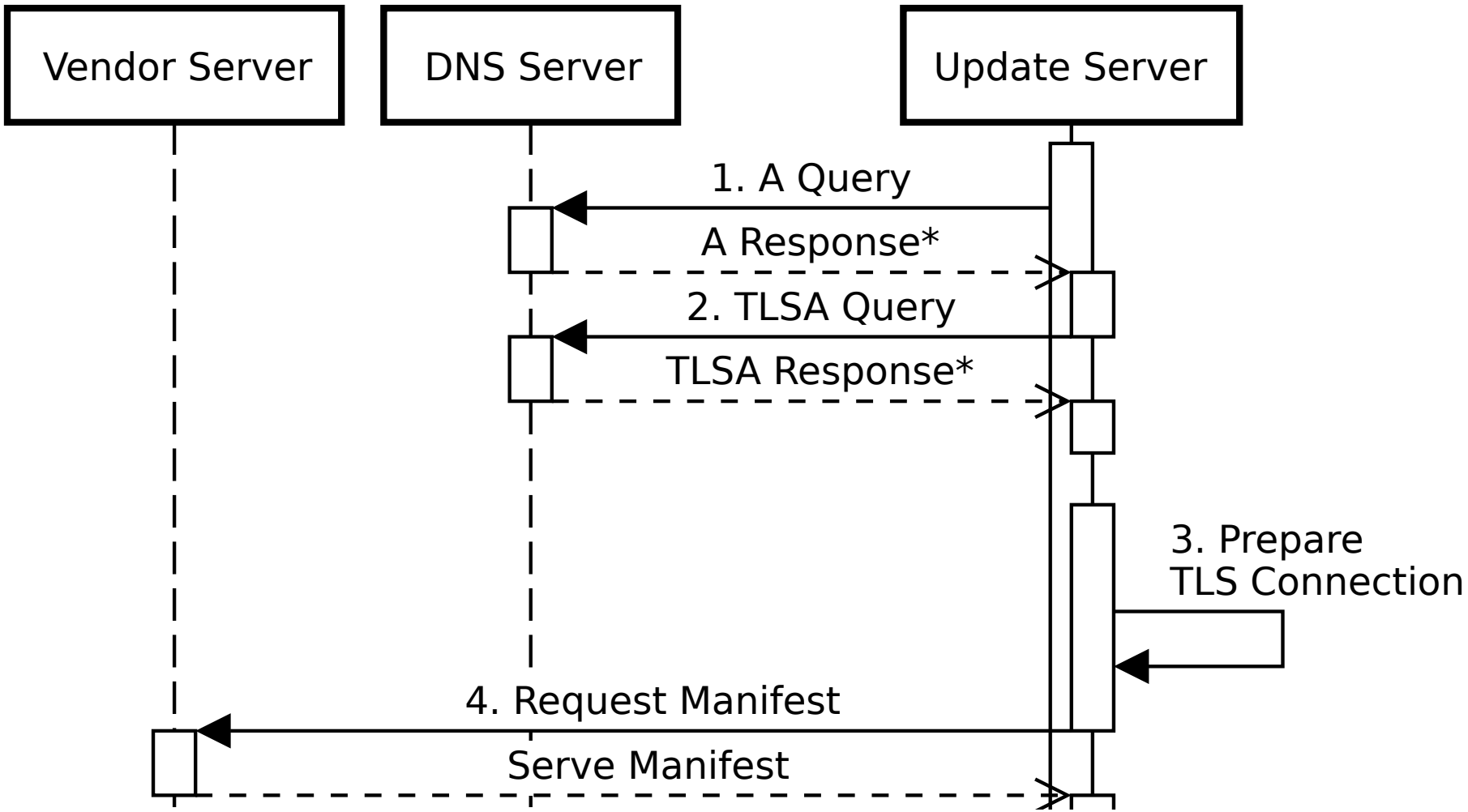
2. Radikale Variante:
   Die Zuordnung Server $\leftrightarrow$ Schlüssel passiert automatisch bei der Namensauflösung:
   Mit DNSSEC: Wir erhalten einen signierten Schlüssel aus einer **vertrauenswürdigen** Quelle!
   $\Longrightarrow$ Keine vorinstallierten Schlüssel oder Zertifikate mehr notwendig!
   Alternative zur PKIX

$\Longrightarrow$ benötigt DNSSEC

Wir benutzen beides:

1. Beim **Aufbau einer TLS- bzw. DTLS-Verbindung**: Überprüfen des Vendor Keys im Vendor-Zertifikat über DANE/DNSSEC mittels Hashwert
2. **Key Roll-Over:** Verteilen des Vendor Code Signing Keys über DNSSEC

TLS-Verbindung zum Vendor Server



DANEs TLSA Record enthält den **Hashwert** vom Public Key des Vendors.

Manifest Validation

bisher: Das IoT-Gerät validiert das Vendor Manifest.

neu: Der Update Server prüft ebenfalls das Vendor Manifest, bevor er es weiterreicht. Dafür braucht er den Code Signing Key des Vendors:

Proposal:

1. Alle Public Keys des Vendors werden im DNS in der Subdomain `keystore.vendordomain.tld` gespeichert.
2. Der aktuelle Code-Signing-Key wird im DNS mittels eines CNAME Records referenziert: `main.keystore.vendordomain.tld`.

Für den Beispiel-Vendor **mup.dnssec-uni-potsdam.de** ergeben sich daraus folgende Records:

```
-- main.keystore.mup.dnssec-uni-potsdam.de IN CNAME --
       cafecafe.keystore.mup.dnssec-uni-potsdam.de

-- cafecafe.keystore.mup.dnssec-uni-potsdam.de IN IPSECKEY --
       10 0 1 . bh9Ol1y7iaQJHbW4jn6BvwiZd2hOiLOT2gc3lZq/SDJ2
                bttKNd9TiTrR8r7QHQ4qxpLBMFoiuiSZ5EECHBJS8A==
```

Manifest Validation

Proposal:

1. Alle Public Keys des Vendors werden im DNS in der Subdomain `keystore.vendordomain.tld` gespeichert.
2. Der aktuelle Code-Signing-Key wird im DNS mittels eines CNAME Records referenziert: `main.keystore.vendordomain.tld`.
3. TLSA Record oder IPSECKEY?

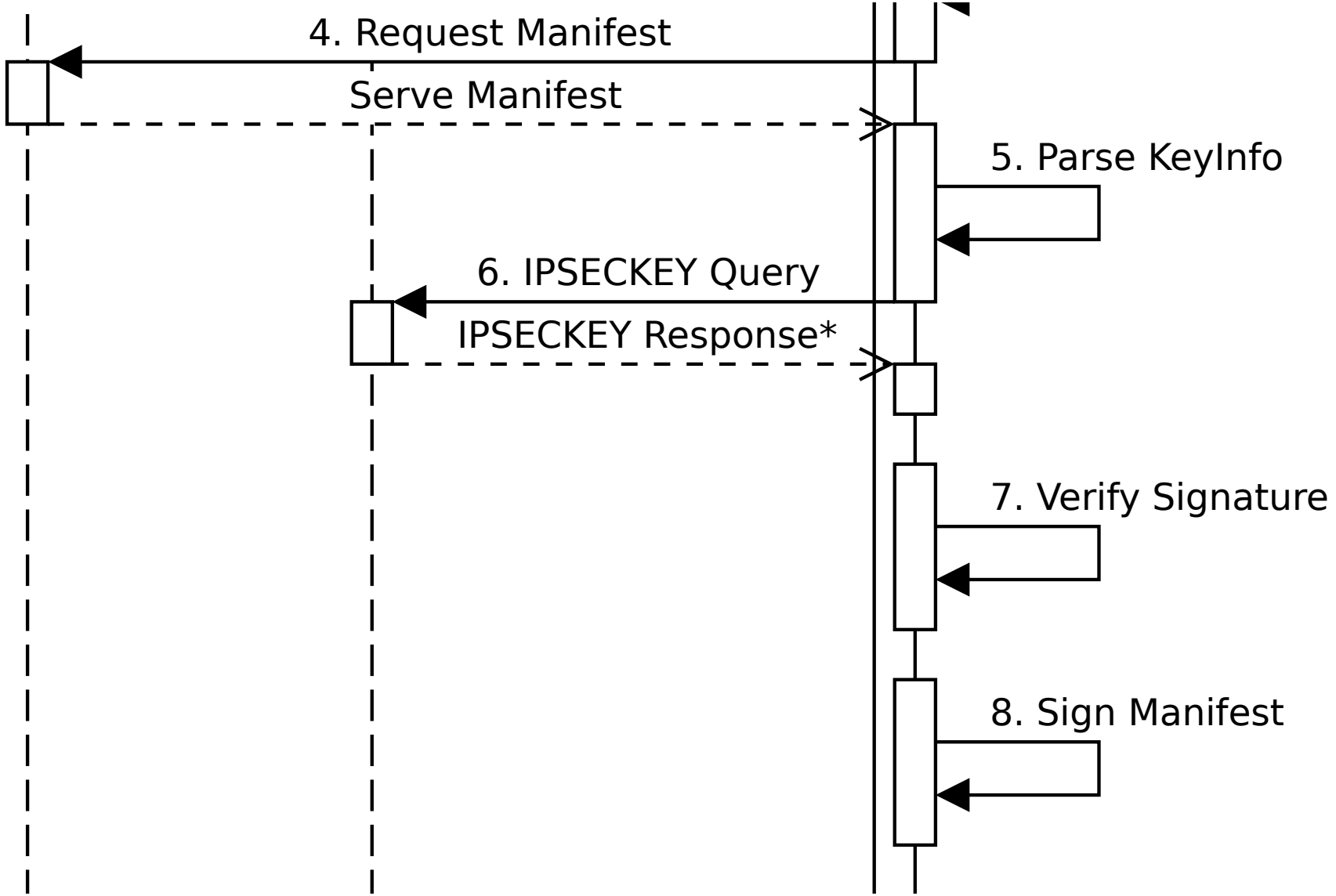
   RFC 4025: A Method for Storing IPsec Keying Material in DNS, 2005

   Hexadezimaldarstellung (TLSA) versus BASE64-Kodierung (IPSEC)
   $\Longrightarrow$ **IPSECKEY (spart ca. 66 %)**
4. Wie ermittelt der Update Server den Schlüssel und das benutzte Verschlüsselungsverfahren?
   Manifesterweiterung: Datenstruktur `KeyInfo`

`KeyInfo`-Feld im Manifest:

- `KeyID`: to identify the public key and to derive the FQDN for retrieving the public key
- `Algorithm`: the algorithm used to create signature
- `KeyType`: the type of public key
- `KeyCurve`: the elliptic curve used

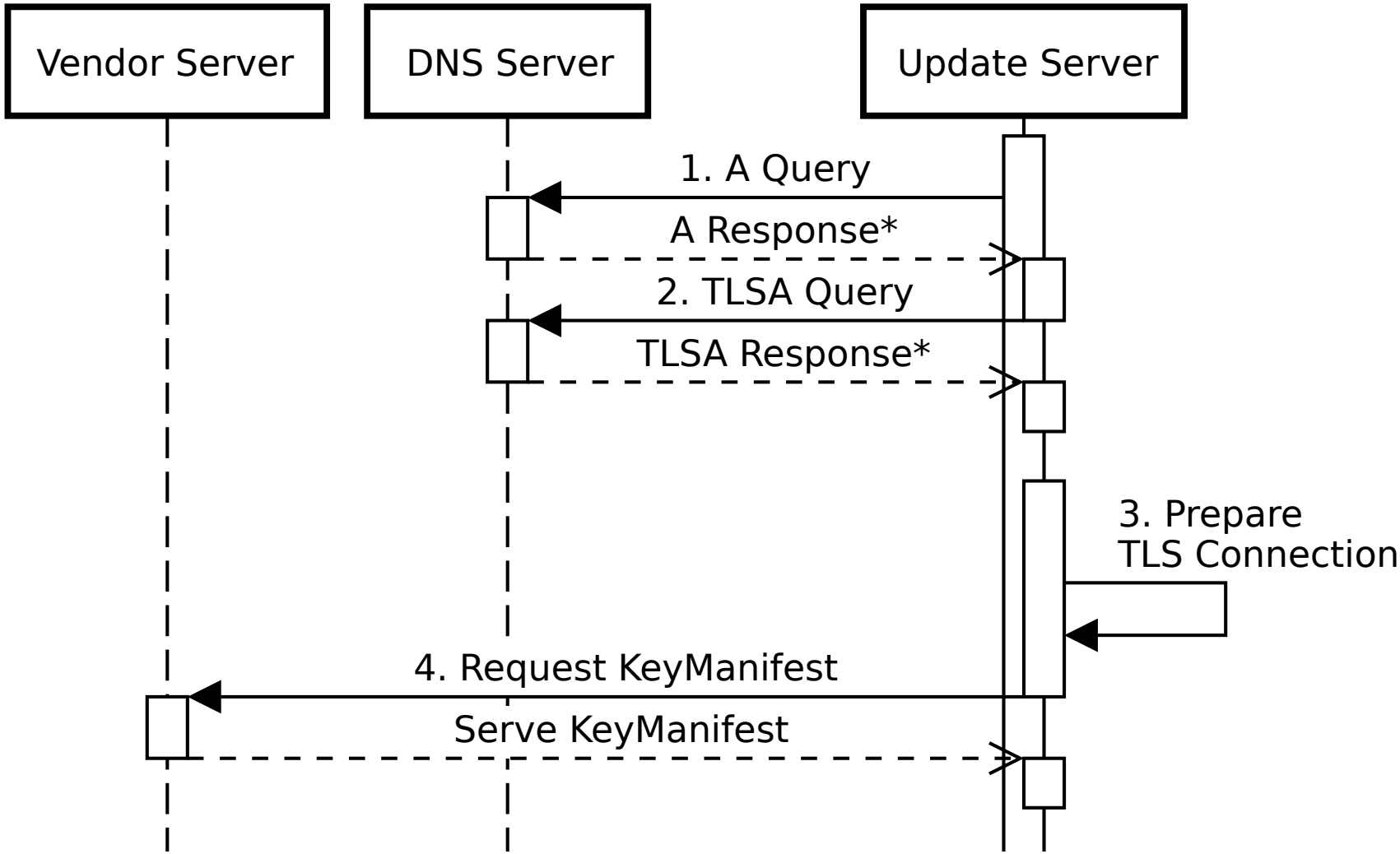
Die Felder werden gemäß COSE kodiert:
CBOR Object Signing and Encryption (COSE), RFC 8152: *"COSE is a data format designed for small code size and small message size."*


4. Request Manifest
Serve Manifest
5. Parse KeyInfo
6. IPSECKEY Query
IPSECKEY Response*
7. Verify Signature
8. Sign Manifest

Key Rollover

Das IoT device besitzt eine neue Key Rollover Funktion.
$\Longrightarrow$ Erweiterung der MYNO Device Capability

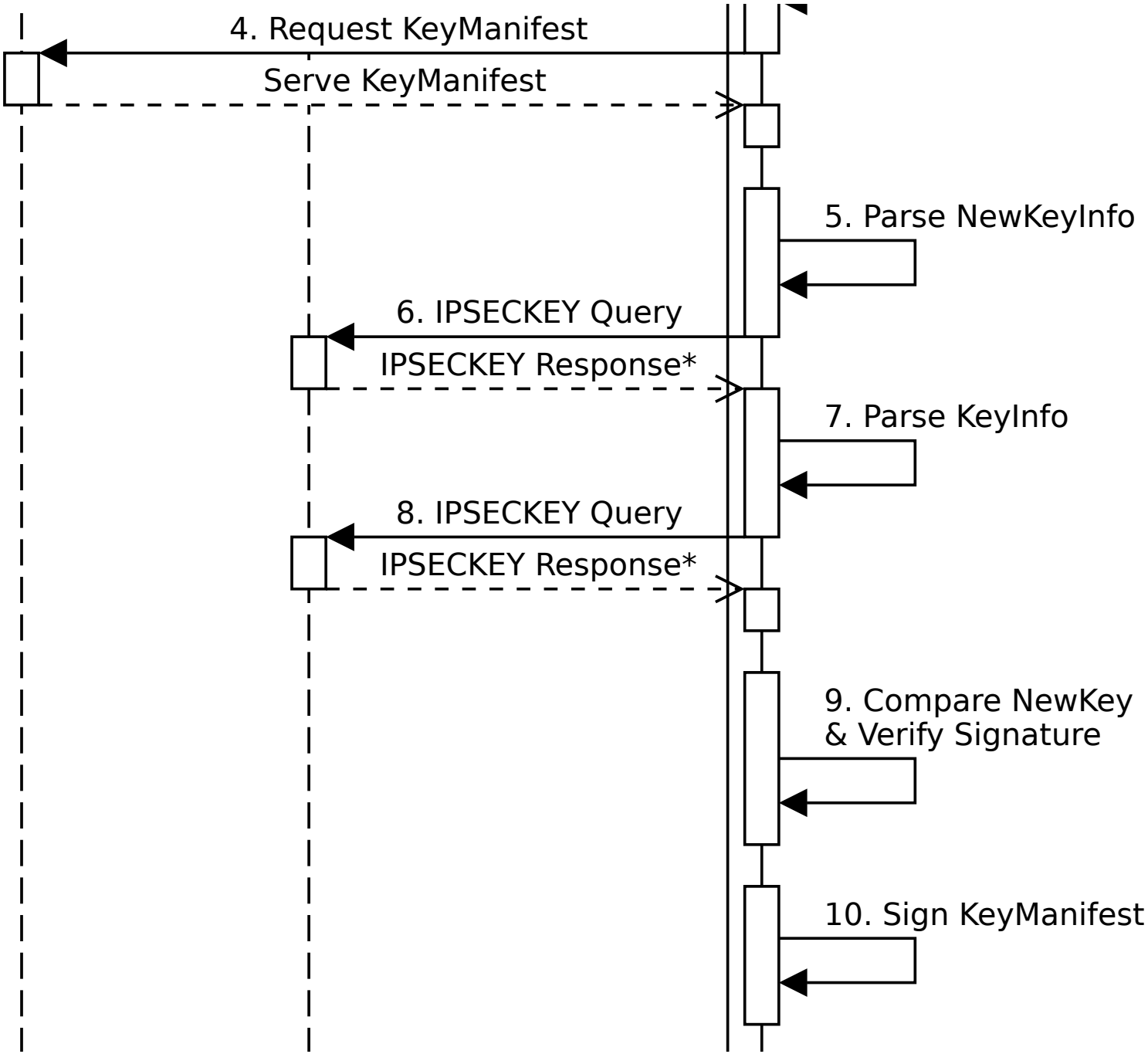
1. Vendor veröffentlicht ein Vendor Key Rollover Manifest (VKRM), um den neuen Schlüssel zu verbreiten.
2. Update Server lädt das VKRM über eine via DANE-abgesicherte TLS-Verbindung runter.


Vendor Server
DNS Server
Update Server
1. A Query
A Response*
2. TLSA Query
TLSA Response*
3. Prepare
TLS Connection
4. Request KeyManifest
Serve KeyManifest

Vendor Key Rollover Manifest	
field	**description**
App ID	unique id for app
version	the version of the app the key rollover is applied
new `KeyInfo`	`KeyInfo` structure for the new public key
public key	the public key to be added to the trust anchor
old `KeyInfo`	`KeyInfo` structure for the old public key
Inner signature	Vendor signature with the **old key** over the fields above

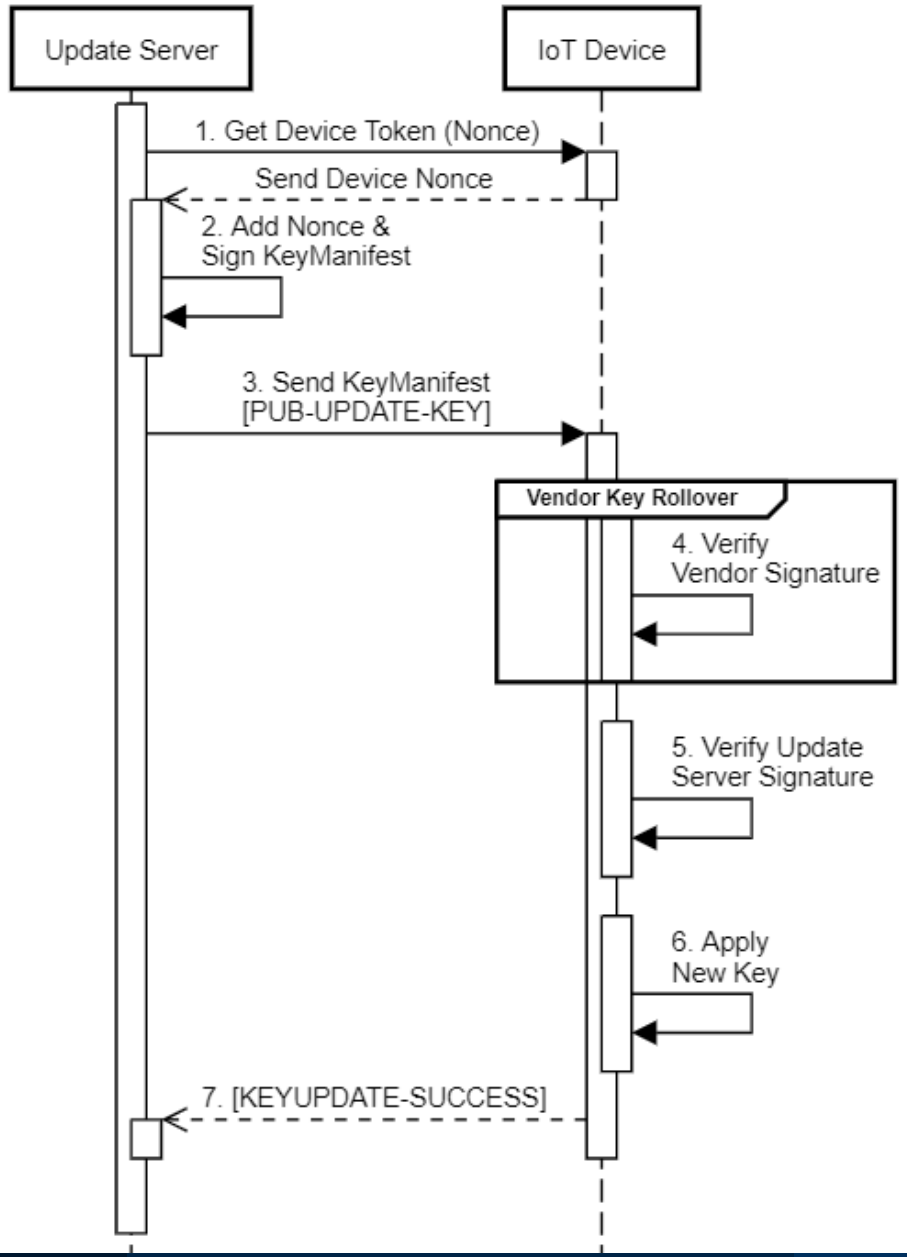
Der Update Server validiert das VKRM. Dazu gehört u.a.

- Prüfen, ob der neue Key derjenige ist, auf den der `main` CNAME Eintrag zeigt.
- Validieren der Inner Signature des Vendors im VKRM mittels des **alten** Keys.
- **...**


4. Request KeyManifest
Serve KeyManifest
5. Parse NewKeyInfo
6. IPSECKEY Query
IPSECKEY Response*
7. Parse KeyInfo
8. IPSECKEY Query
IPSECKEY Response*
9. Compare NewKey
& Verify Signature
10. Sign KeyManifest

Vendor Key Rollover Manifest	
field	**description**
App ID	unique id for app
version	the version of the app the key rollover is applied
new `KeyInfo`	`KeyInfo` structure for the new public key
public key	the public key to be added to the trust anchor
old `KeyInfo`	`KeyInfo` structure for the old public key
Inner signature	Vendor signature with the **old key** over the fields above
Manifest Extension	
Nonce	nonce generated by IoT device
Outer signature	Update Server signature

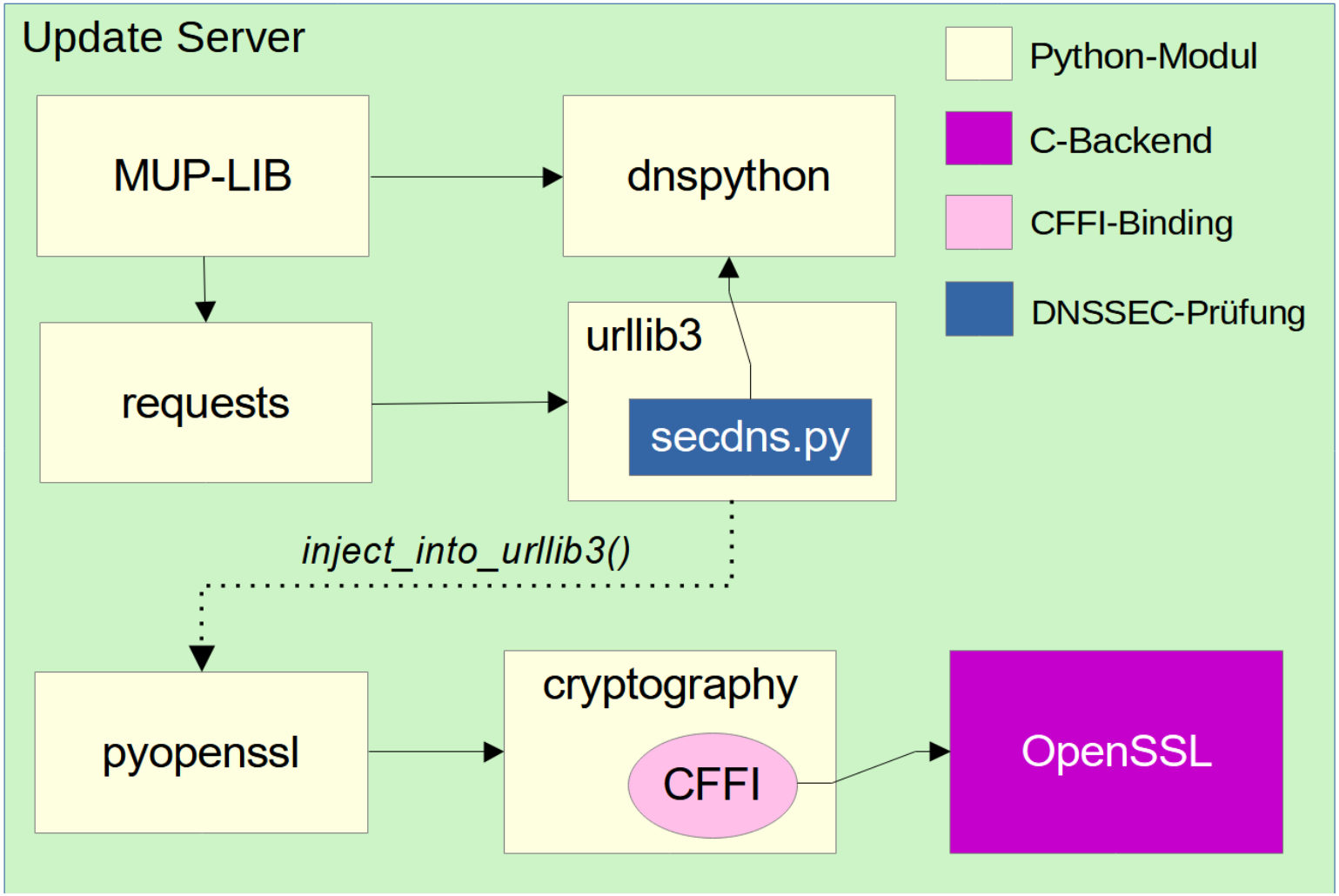
Key Rollover auf dem IoT Gerät



Implementierung auf nRF52840

- Key Rollover Funktion auf den nRF52840 von Nordic:
  256 kB RAM, 1 MB Flash, IEEE 802.15.4 (Bluetooth), wird von Contiki und RIOT unterstützt
- DNS-Server für die DNSSEC-Domain `dnssec-uni-potsdam.de`
- Nachnutzen der DANE-Funktionalität in OpenSSL $\Longrightarrow$ Erfordert das Anpassen von verschiedenen Python-Modulen: `requests`, `urllib3`, `pyopenssl` und `cryptography`
- Python-Skript `secdns` für das Verifizieren der DNSSEC-Kette

Beispiel: Ergänzen des Parameters `check_dane` im `requests`-Modul, um die DANE-Funktionalität von OpenSSL bekannt zu machen.



- Nachnutzen der DANE-Funktionalität in OpenSSL $\Longrightarrow$ Erweitern von verschiedenen Python-Modulen
- Python-Skript `secdns` für das Verifizieren der DNSSEC-Kette

Implementierungsaufwand in LoC

Library	Added LoC
requests	7
urllib3 incl. secdns	328
pyopenssl	9
cryptography	8
dnspython	0
MUP-DANE	193

MUP-DANE: übernimmt die IPSECKEY Anfragen und das Verifizieren der Manifeste

Lessons Learned and Take Away

- MYNOs Edge-based Approach *schont* ressourcenarme IoT-Geräte - auch beim Update
- Im DNS kann man nicht nur IP-Adessen speichern!
- DANE für die Absicherung von TLS-Verbindungen
  $\Longrightarrow$ DANE macht Certificate Revocation überflüssig
- Ermöglicht schnelles Zurückziehen eines nicht mehr vertrauenswürdigen Schlüssels über DNS-Eintrag, da der Update Server das Vendor Manifest zusätzlich prüft.
- Proposal für ein Vendor KEY Rollover Protocol (VKRP) mittels IPSECKEY Records und Vendor Key Rollover Manifest (VKRM)
  $\Longrightarrow$ Alternative zur PKIX
- sehr überschaubarer Implementierungsaufwand

DNSSec/DANE sollte so selbstverständlich wie Zähneputzen sein!

Quellen

K. Sahlmann, V. Clemens, M. Nowak, and B. Schnor, "MUP: Simplifying Secure Over-The-Air Update with MQTT for Constrained IoT Devices," *Journal Sensors (Open Access)*, Dec 2020. [Online]. Available: https://www.mdpi.com/1424-8220/21/1/10

M. T. Paracha, D. J. Dubois, N. Vallina-Rodriguez, and D. Choffnes, "IoTLS: Understanding TLS Usage in Consumer IoT Devices," in *Proceedings of the 21st ACM Internet Measurement Conference*, ser. IMC '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 165–178. [Online]. Available: https://doi.org/10.1145/3487552.3487830