AC-SIF: ACE Access Control for Standardized Secure IoT Firmware Updates

Joel Höglund, Anum Khurshid and Shahid Raza, RISE
joel.hoglund@ri.se, anum.khurshid@ri.se, shahid.raza@ri.se

presented at SecurWare2022, October 16—20, 2022

Presenter resume

Joel Höglund is pursuing a phd degree within cyber security, working as an industrial phd student at RISE, Research Institutes of Sweden.

The focus is on PKI for IoT. This includes lightweight security solutions for certificate enrollment and sensor data storage.

Earlier work includes work as an research engineer within several EU projects, with low power wireless sensor network solutions, including solutions for routing, Smart Grids and structural monitoring.

When not working he is likely to be found on a dancefloor, or on one of his bikes, possibly carrying a camera.

Outline

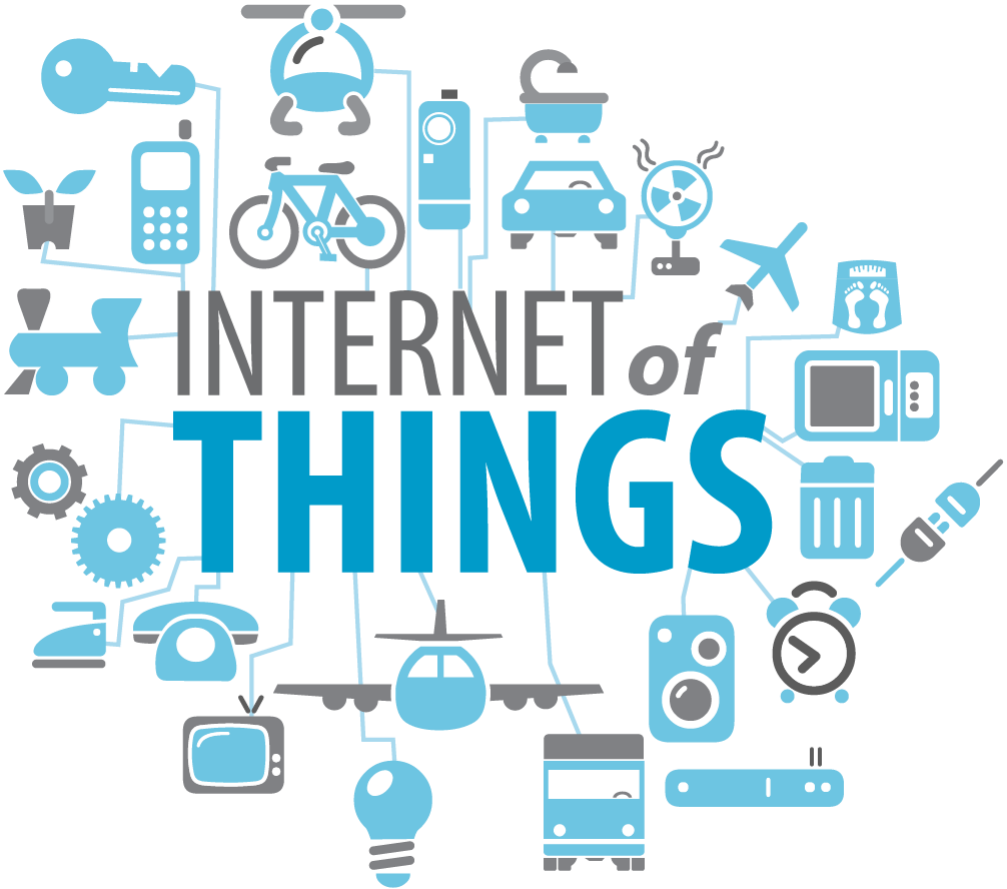
Challenges, current status & related work

Design and encodings

Evaluation

Security challenges of deployed IoT systems

- Deployed IoT systems are not sufficiently updated
  - leaving them open to security vulnerabilities
- *If* the systems can be updated, it is through proprietary nonstandard mechanisms
  - Vendor lock-in and no interoperability


INTERNET of
THINGS

Challenges and requirements for secure updates

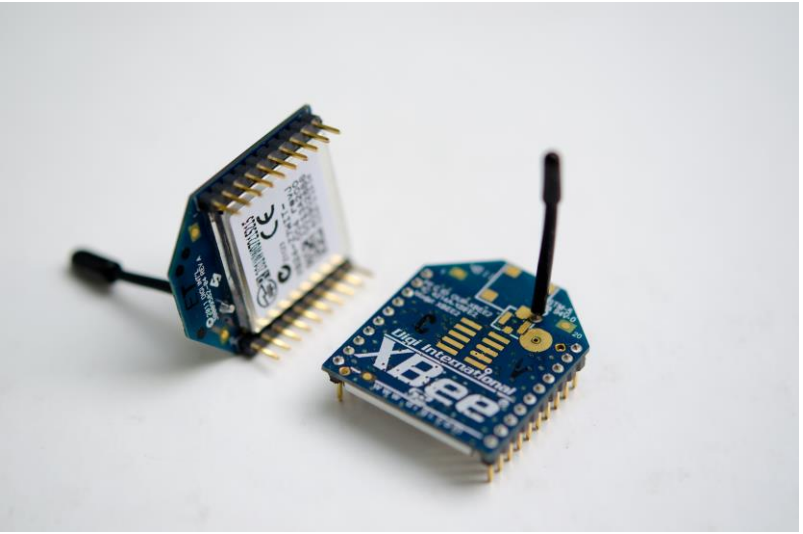
An unsecure update mechanism:

- Becomes an attack vector in itself
- Can be used to install malware or brick devices

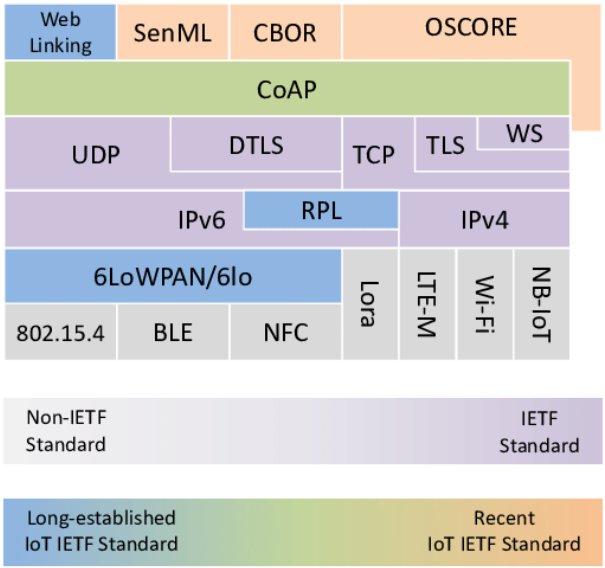
IoT devices must be able to:

- Authenticate and check permissions of the update author
- Verify origin & integrity of firmwares

Challenge: resource constraints



Challenge: lack of established standards suitable for IoT



Related work

Firmware updates

- Image-based
- Differential

Update distribution architectures

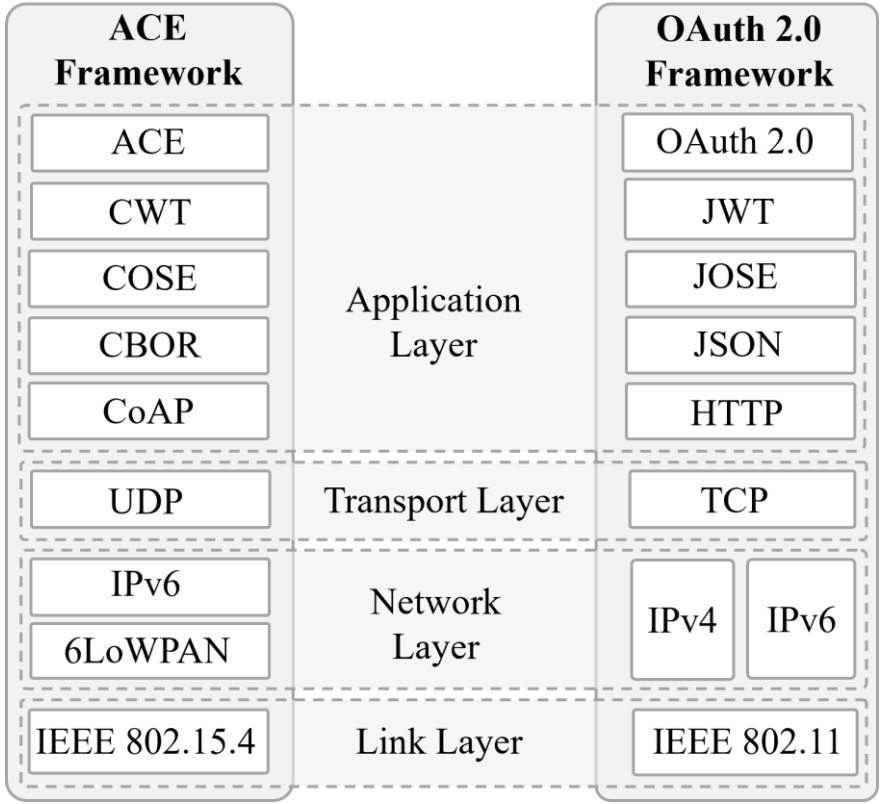
- Package managers
- How to establish and strengthen trust?
  - Require more signatures

The ACE Framework

Authentication & authorization for IoT

+SUIT

Software updates for the Internet of Things



Our work

Builds upon emerging IoT technologies and recommendations from IETF SUIT working group to design a firmware update architecture which

- Provides end-to-end security between update authors and IoT devices
- Does not require trust anchor provisioning by the manufacturer
- Is agnostic to the underlying transport protocols
- Uses standard solutions for crypto and message encodings.

System overview

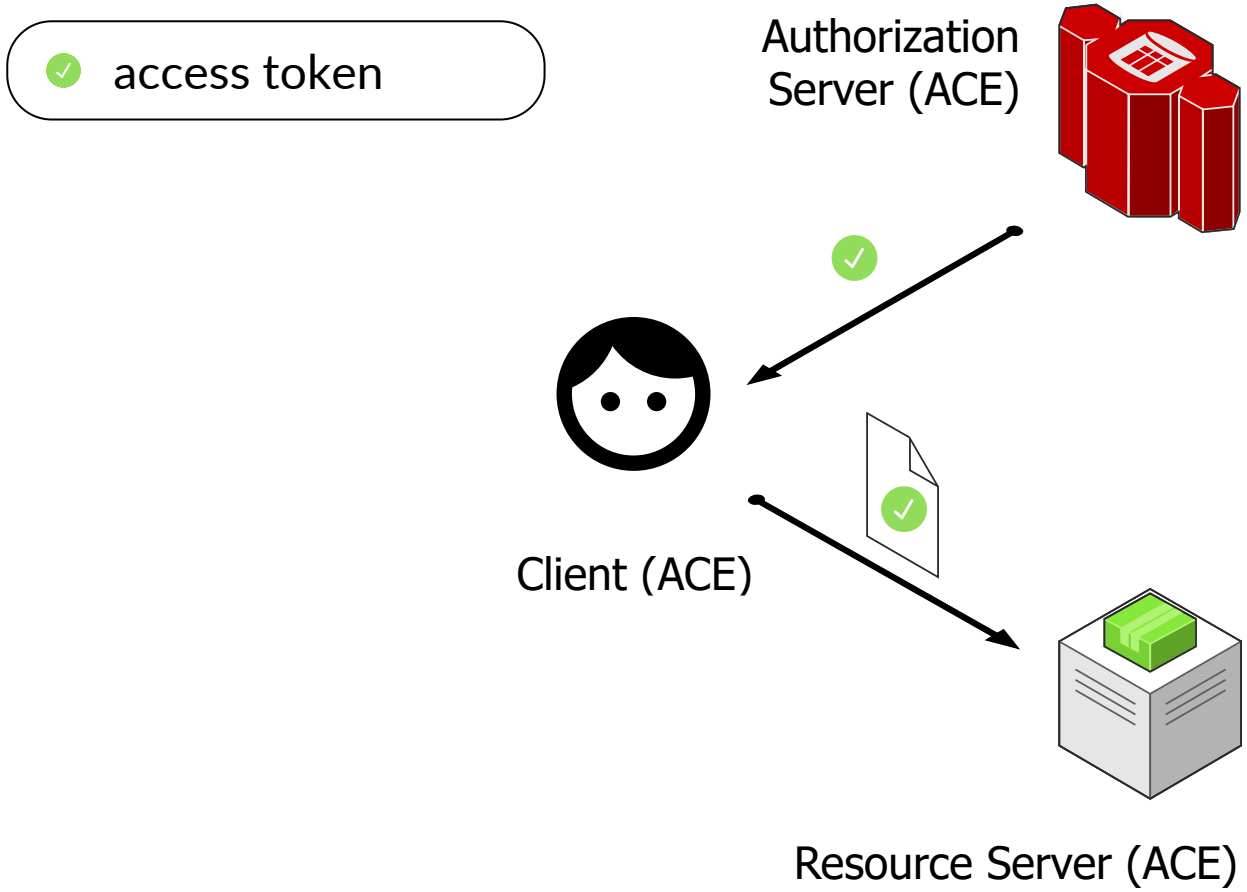
In a generic ACE scenario:

Clients request access to protected resources from an Authorization Server

AS grants the client a token, bound to:

- a secret key in the client's possession,
- a specific resource and
- an expiration date.

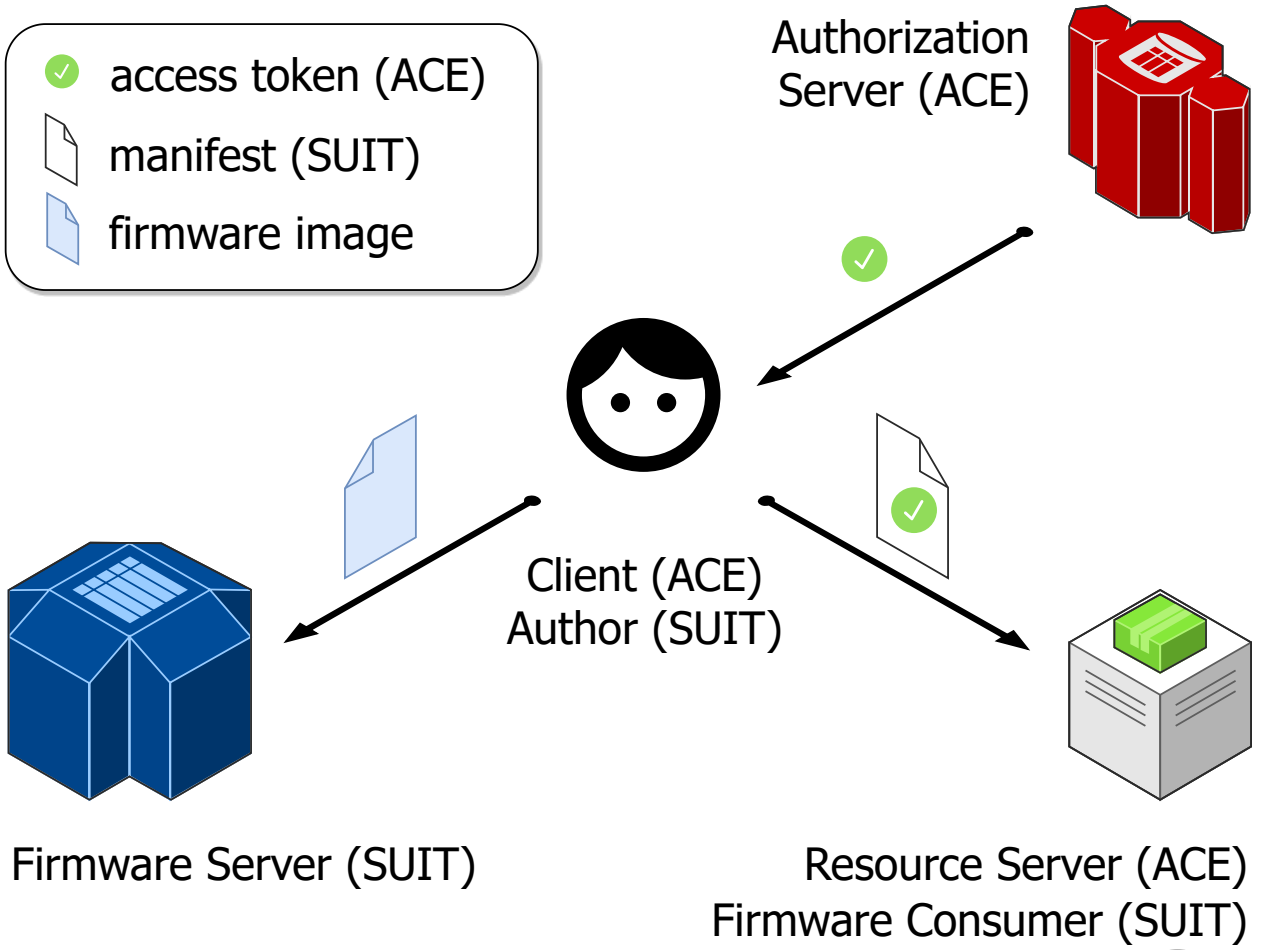
This token is then used as proof of authorization when accessing the Resource Server (RS).



System overview

In our proposed solution:

- The manifest author acts as the client
- The AS grants the author a token, which is attached to the manifest
- The IoT device acts as the RS, validating the token before accepting the manifest



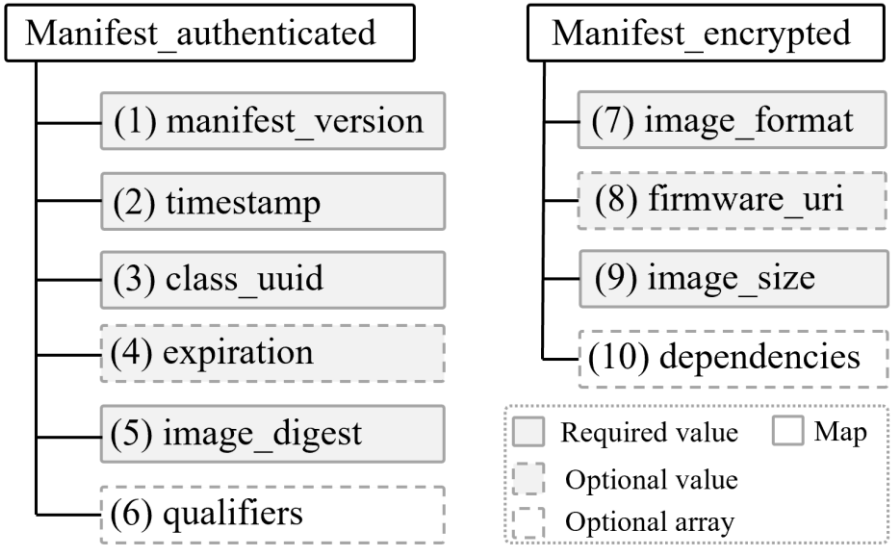
Firmware manifest structure

`class_uuid` – compared by IoT with `aud` field in the access token

`timestamp` – preventing rollback attacks

`dependencies` – allows differential updates

`recipients` in the manifest wrapper – used to contain the Content Encryption Key (CEK), enciphered with Key Encryption Keys (KEK) known only to the intended recipients



Information about image contents is encrypted

Concealing info that is useful to adversaries attempting to gain insights into the device sw.

Includes dependencies and exact firmware URI

Unencrypted part is used as AAD: any device can quickly ascertain if the update applies, without crypto operations

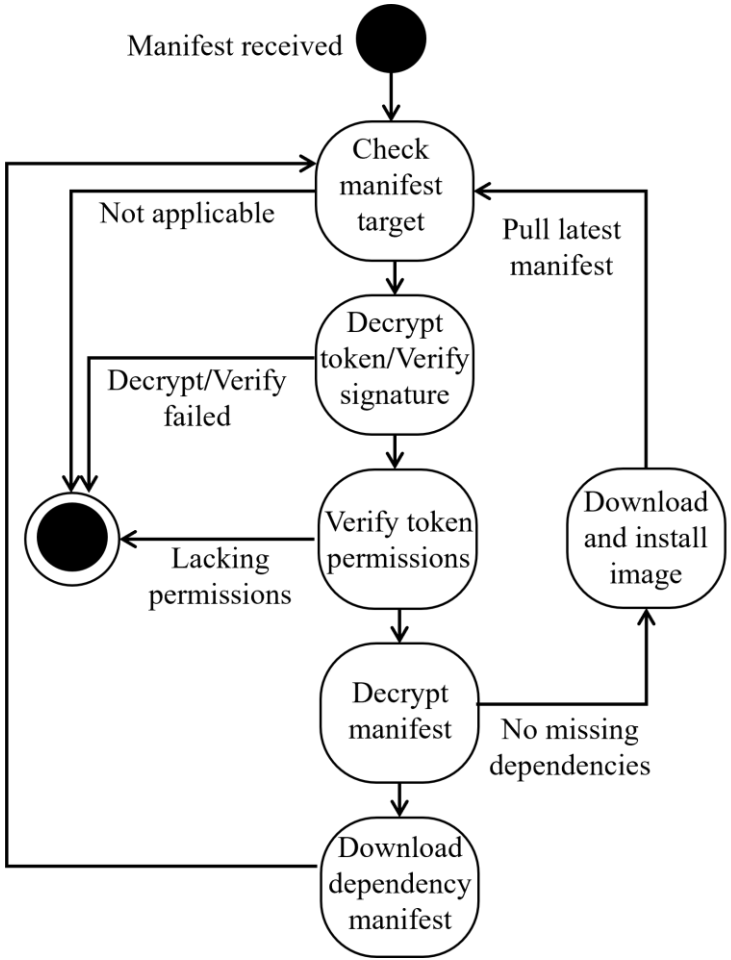
Manifest dependency tree traversal

Supports image-based & differential updates with dependencies.

Differential updates illustrated here.

To handle loss of state:

Query a known manifest distributor at startup, request latest manifest. The IoT device knows the update is complete when it receives manifest matching current firmware.

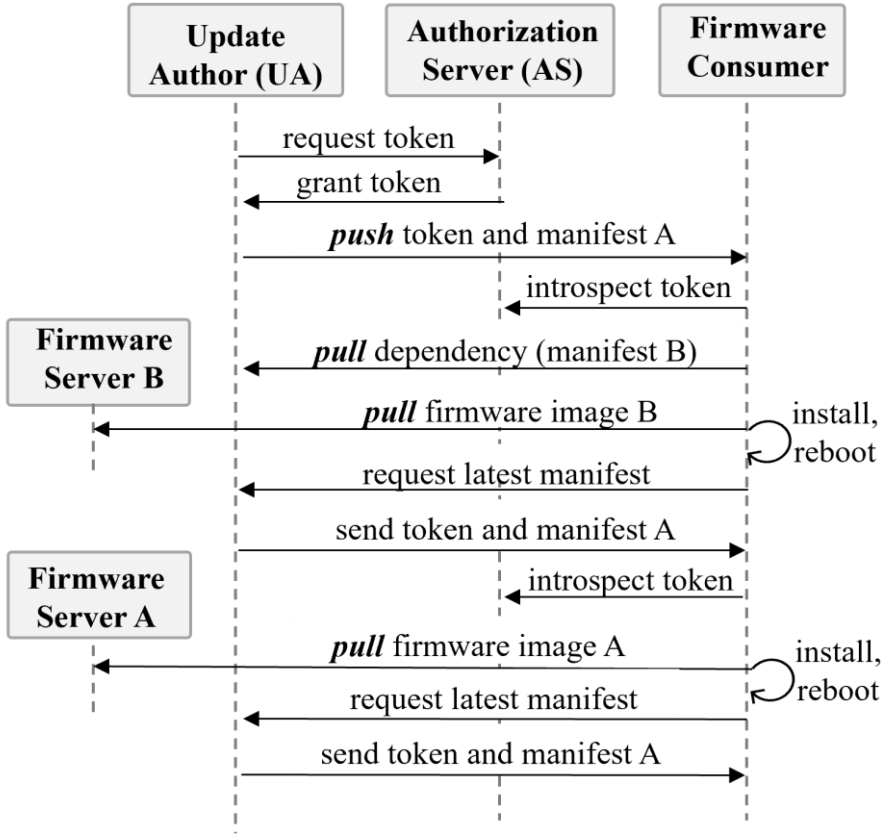


Update sequence

A push-based process can be turned into a pull based, by adding periodical polling by the IoT client

- Pull based is often needed, because a direct path to the IoT devices is not know

Introspection step is optional



Authentication options

	AEAD	ECDSA	ECDH	KDF
A	manifest, token			
B	manifest, token	token	manifest	
C	manifest	token	manifest	manifest
D	manifest	token	manifest	

A. Symmetric Pop Key with PSK

- One unique PoP key for each access token, issued by Authorization Server
- More vulnerable than the other options, only for the most constrained devices

B. Symmetric PoP Key

- AS generates PoP key and encrypts it with itself
- Avoids including any recipients in the token itself
- Breaks end-to-end security between the author and recipients – PoP key is used as CEK, and known to the AS

C. Asymmetric PoP Key, Direct Key Agreement

- UA generates unique CEK for each recipient

D. Asymmetric PoP Key, Key Wrap

- UA uses the key derived through ECDH as a Key Encryption Key (KEK) to encipher a randomly-generated ephemeral CEK.

Implementation

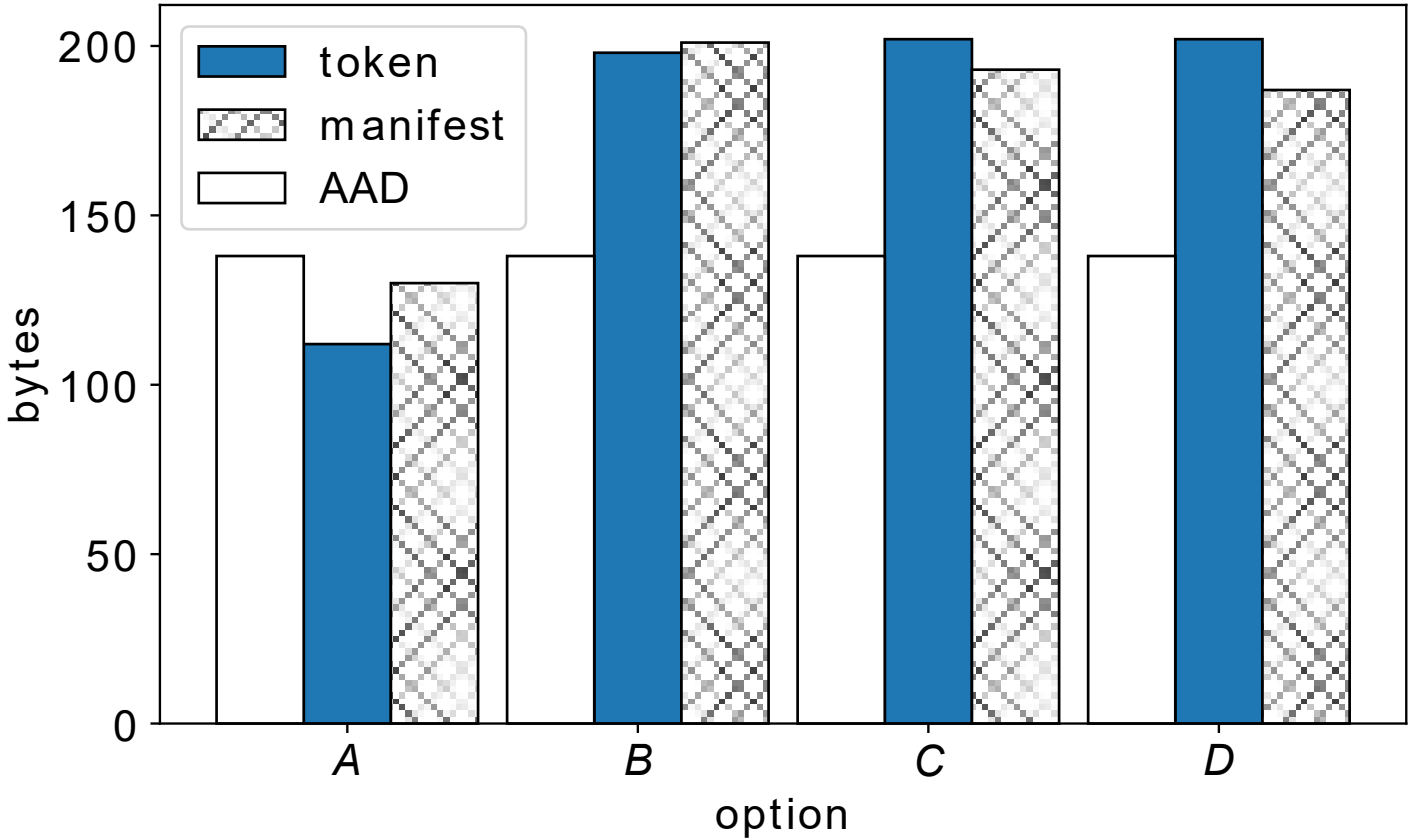
- Done in C, using COSE + CBOR
- ECDSA signatures with 256-bit keys
- AES-CCM with 128-bit keys + 13-byte nonce for content encryption
- AES 128-bit key wrap
- ECDH Ephemeral-Static (ES)
- HMAC-Based Extract-and-Expand Key Derivation Function (HKDF) with SHA-256

Take-home message:

Future proof key sizes, but lightweight enough to be feasible for modern IoT devices

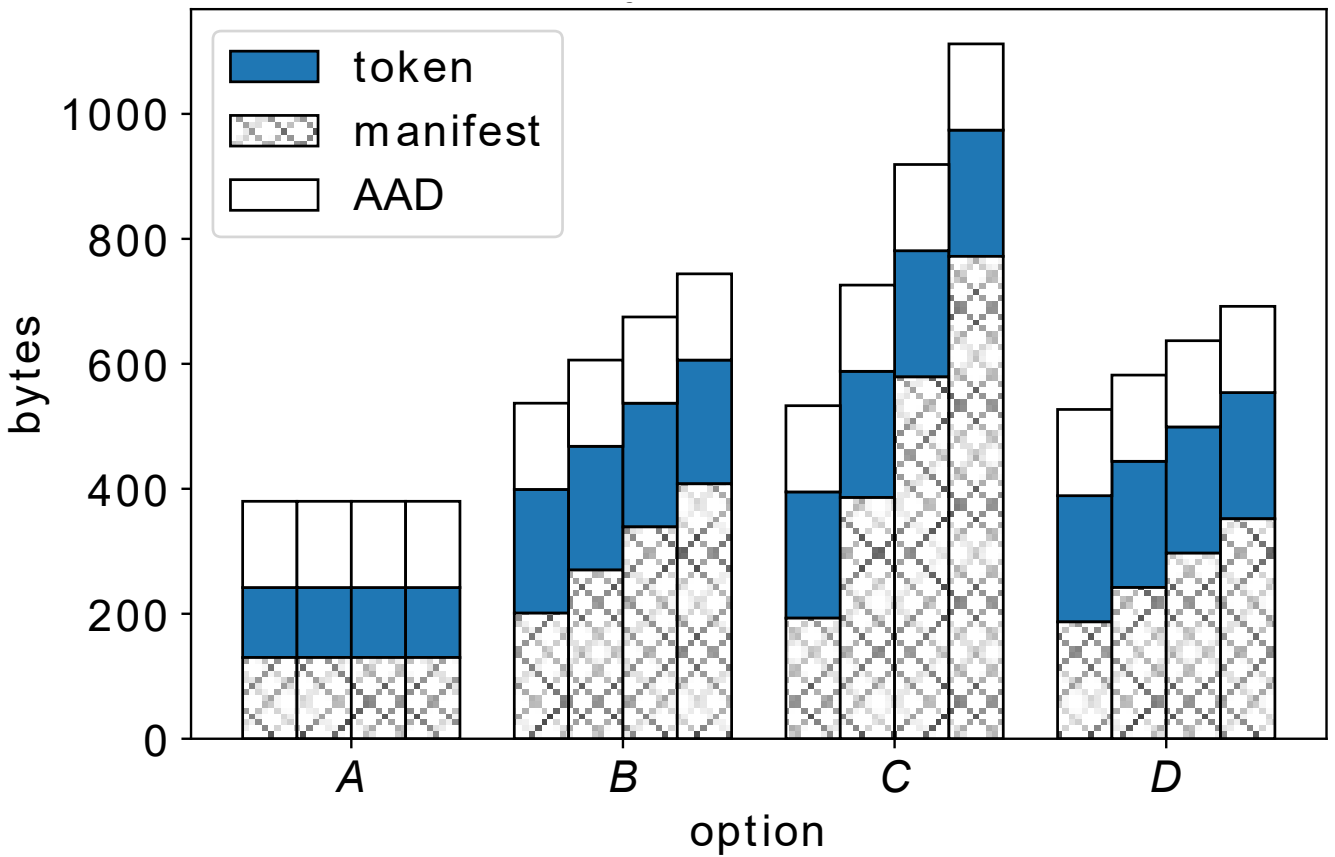
Evaluation: encoded object sizes

Single recipient



Evaluation: encoded object sizes

Multiple recipients



Comparison with ongoing SUIT proposals

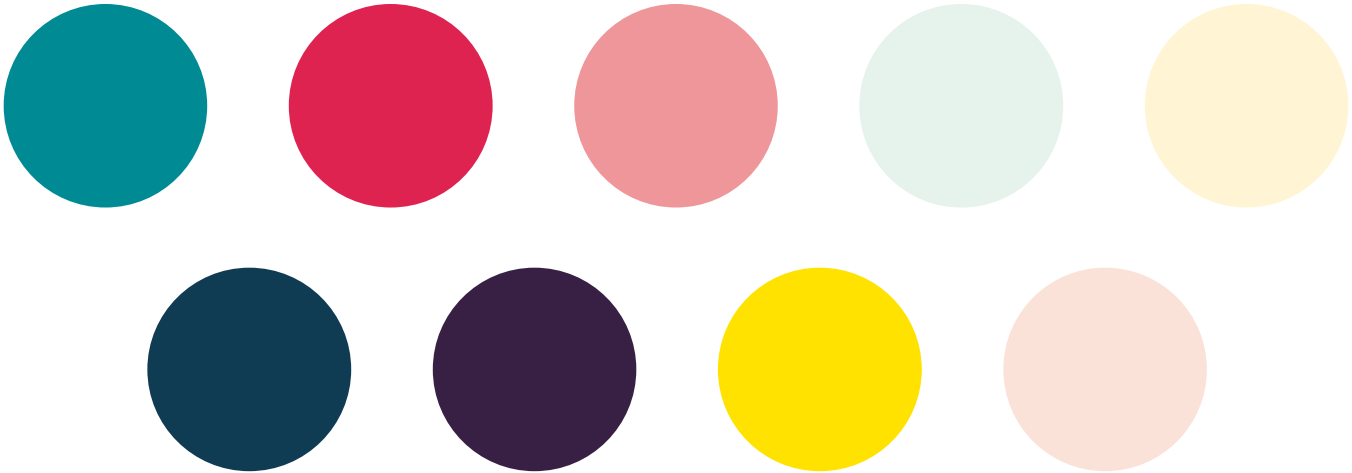
- Comparable data sizes
- Our proposed solution addresses authorization

Security considerations

- Avoid PSK (A) whenever possible!
- Avoid symmetric PoP keys (B) if you want to keep end-to-end security
  - Asymmetric PoP keys have a similar overhead
- For option C: using Ephemeral-Static key derivation lowers the risks, and the usability of leaked private keys

Conclusion

A standard compliant, low overhead secure token-based authorization schema for SUIT updates is feasible.



Joel Höglund

joel.hoglund@ri.se
+4670-775 16 09

Shahid Raza

shahid.raza@ri.se