ICALEPCS2017
Barcelona · Spain, October 8-13 · Palau de Congressos de Catalunya

European XFEL

Securing Light Source SCADA Systems

Leonce Mekinda, Valerii Bondar, Sandor Brockhauser, Cyril Danilevski, Wajid Ehsan, Sergey Esenov, Hans Fangohr, Gero Flucke, Gabriele Giovanetti, Steffen Hauf, David Gareth Hickin, Anna Klimovskaia, Luis Maia, Thomas Michelat, Astrid Muennich, Andrea Parenti, Hugo Santos, Kerstin Weger, Chen Xu.
European XFEL GmbH

Barcelona, 12/10/2017

Overview

- The security of SCADA systems is an increasing concern as they interconnect a significant number of COTS computers via IP networks; support *de facto* standards like USB.
- What happens once attackers have been granted access to / broke into the Control Network?
  - Can they do everything?
  - Can they easily escalate their privileges?
- “We trust whoever has access to the Control Network”
  - Would you let your personal laptop unlocked 24/7 in a control room? If no, why should the control system be less protected than your laptop?
- We suggest to secure the SCADA system beyond the general IT infrastructure security
  - Device servers would authenticate and authorize users for every issued message.

The European X-ray Free Electron Laser

- The most brilliant X-ray light source: $5x10^{33}$ photons s^{-1} $mm^{-2}mrad^{-2}$ per 0.1% bandwidth
- 4.5 MHz pulse rate in burst mode, maximum electron energy of 17.5 GeV, 0.05 nm minimum wavelength
- 1.4 billion euro facility
- Unique pieces of technology: Adaptive Gain Integrating Pixel Detector, Large Pixel Detector, DEPSET Sensor with Signal Compression
- 15 TB of data per beam day

LPD detector in the FXE instrument hutch

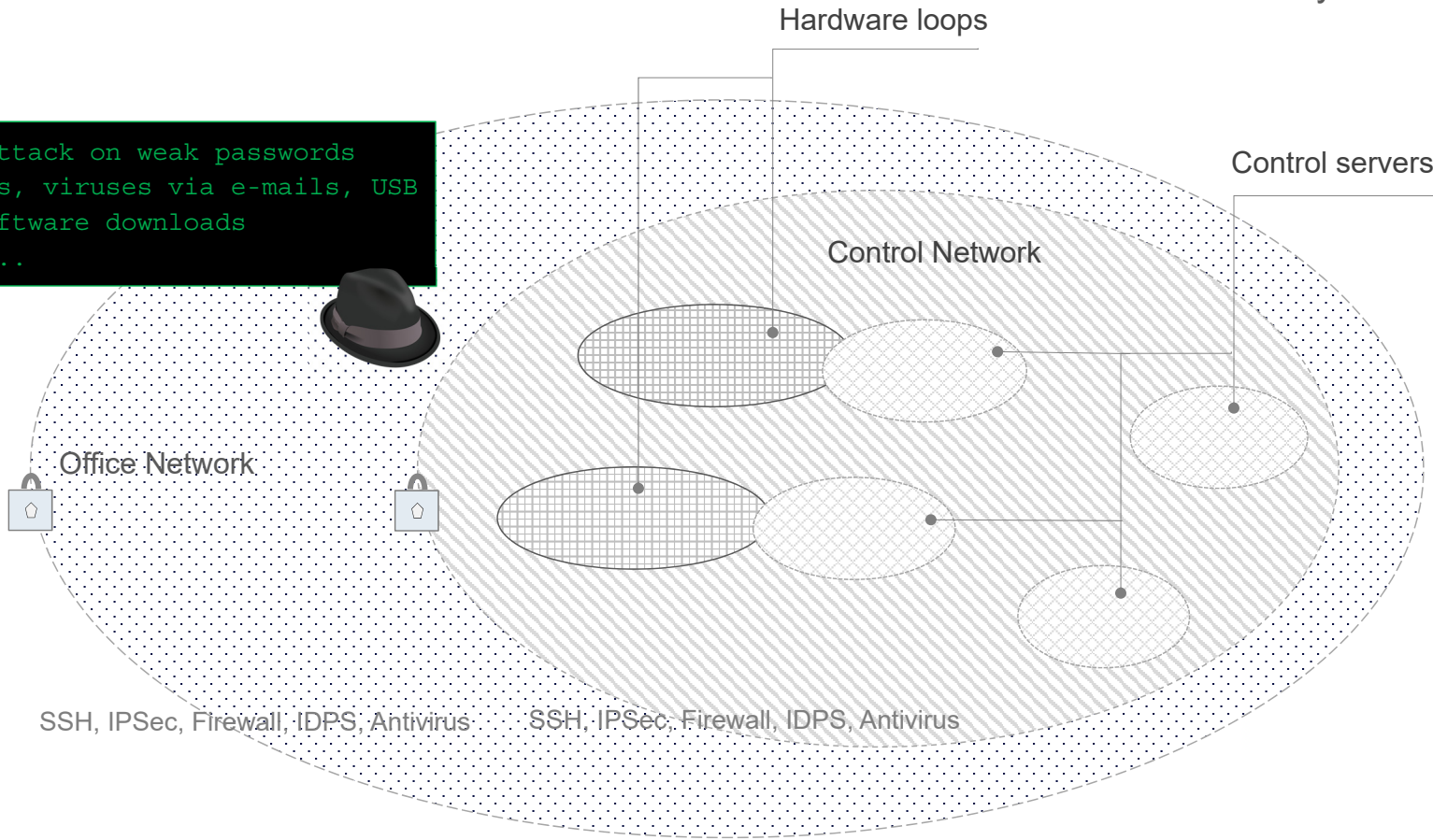
The security **onion**

```
# Dictionary attack on weak passwords
# Trojan horses, viruses via e-mails, USB
  sticks or software downloads
# Keyloggers ...
```

Any random compromised account often suffices.

The security **onion**

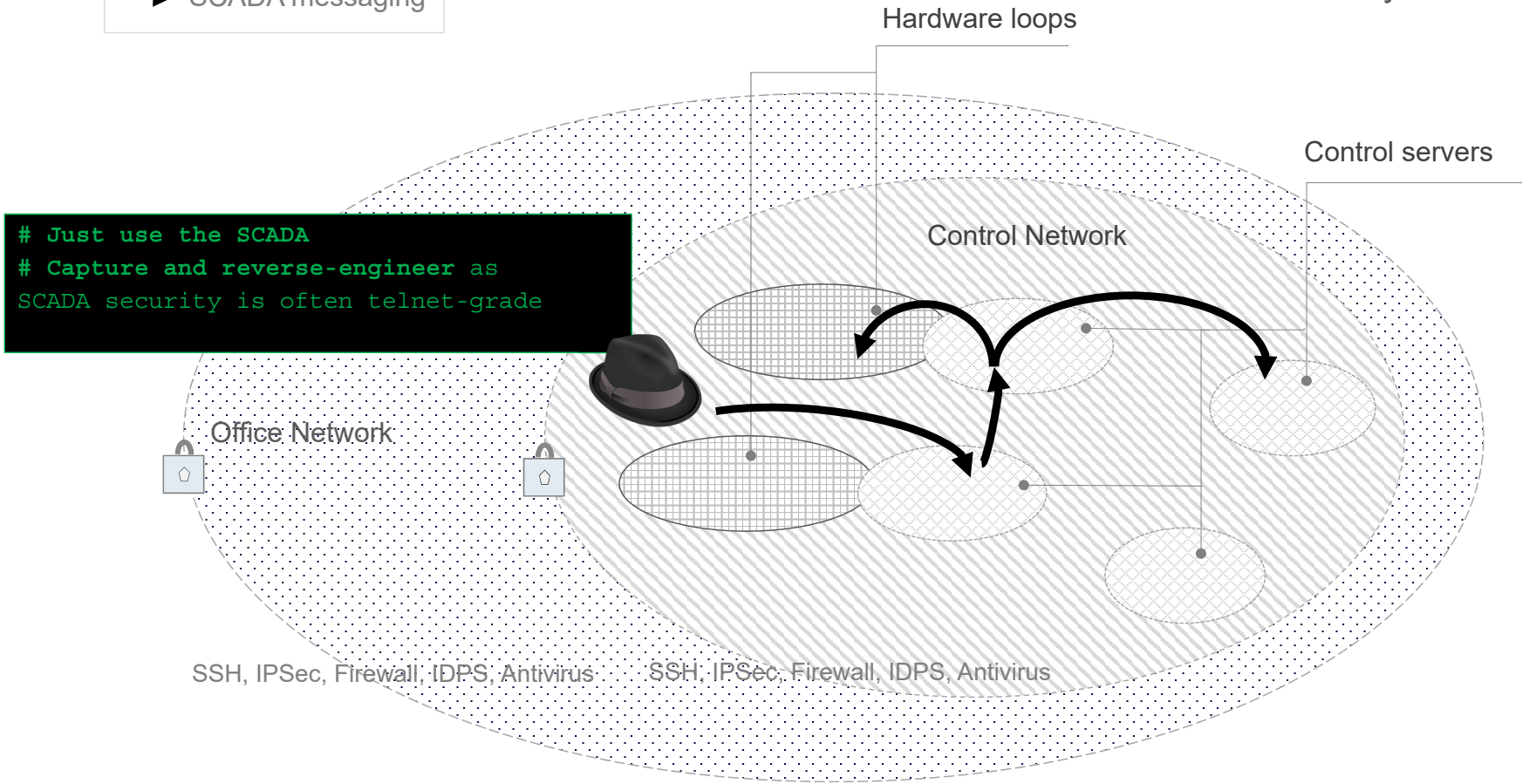
```
# Dictionary attack on weak passwords
# Trojan horses, viruses via e-mails, USB
  sticks or software downloads
# Keyloggers ...
```



Any random compromised account often suffices.

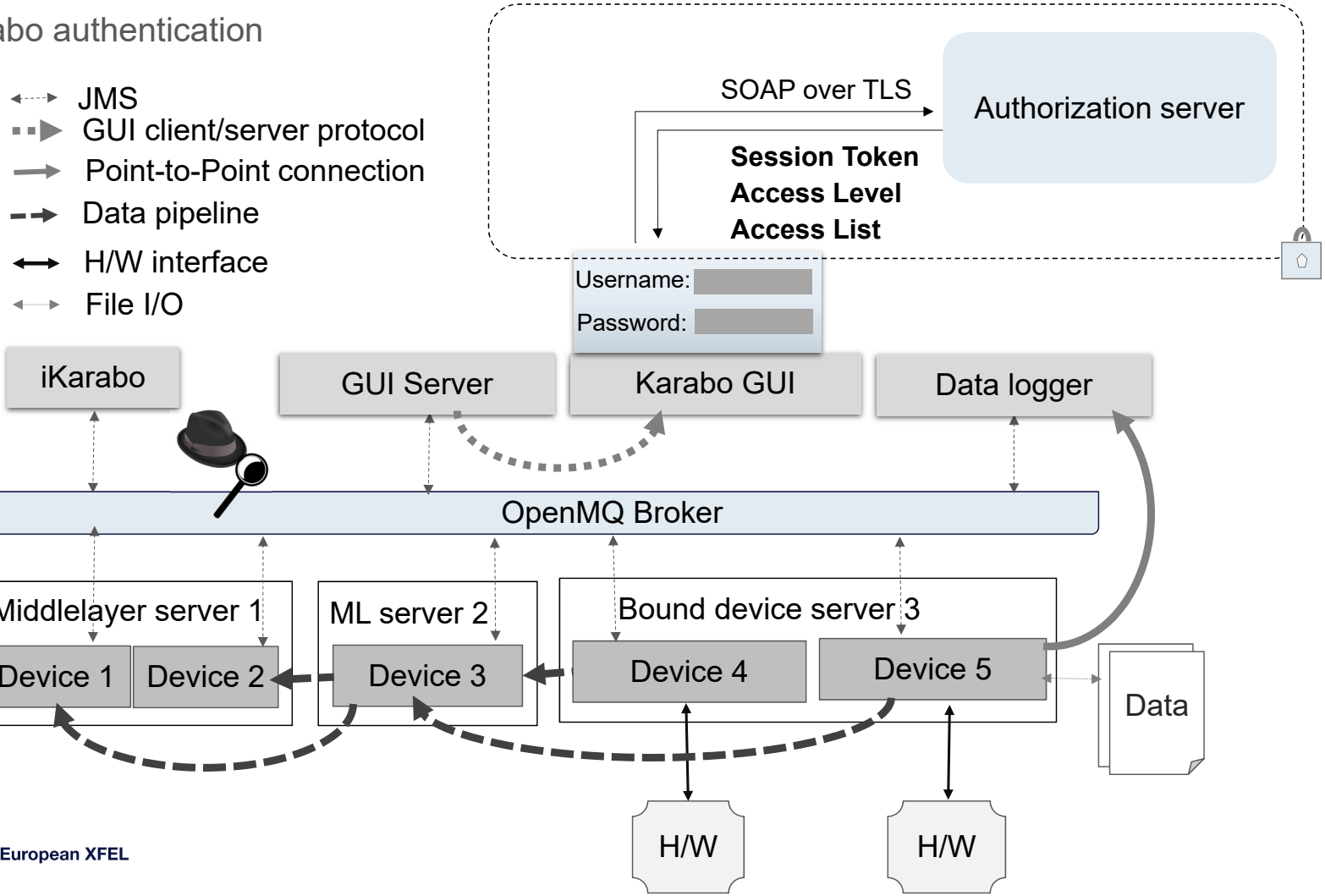
The security **onion**

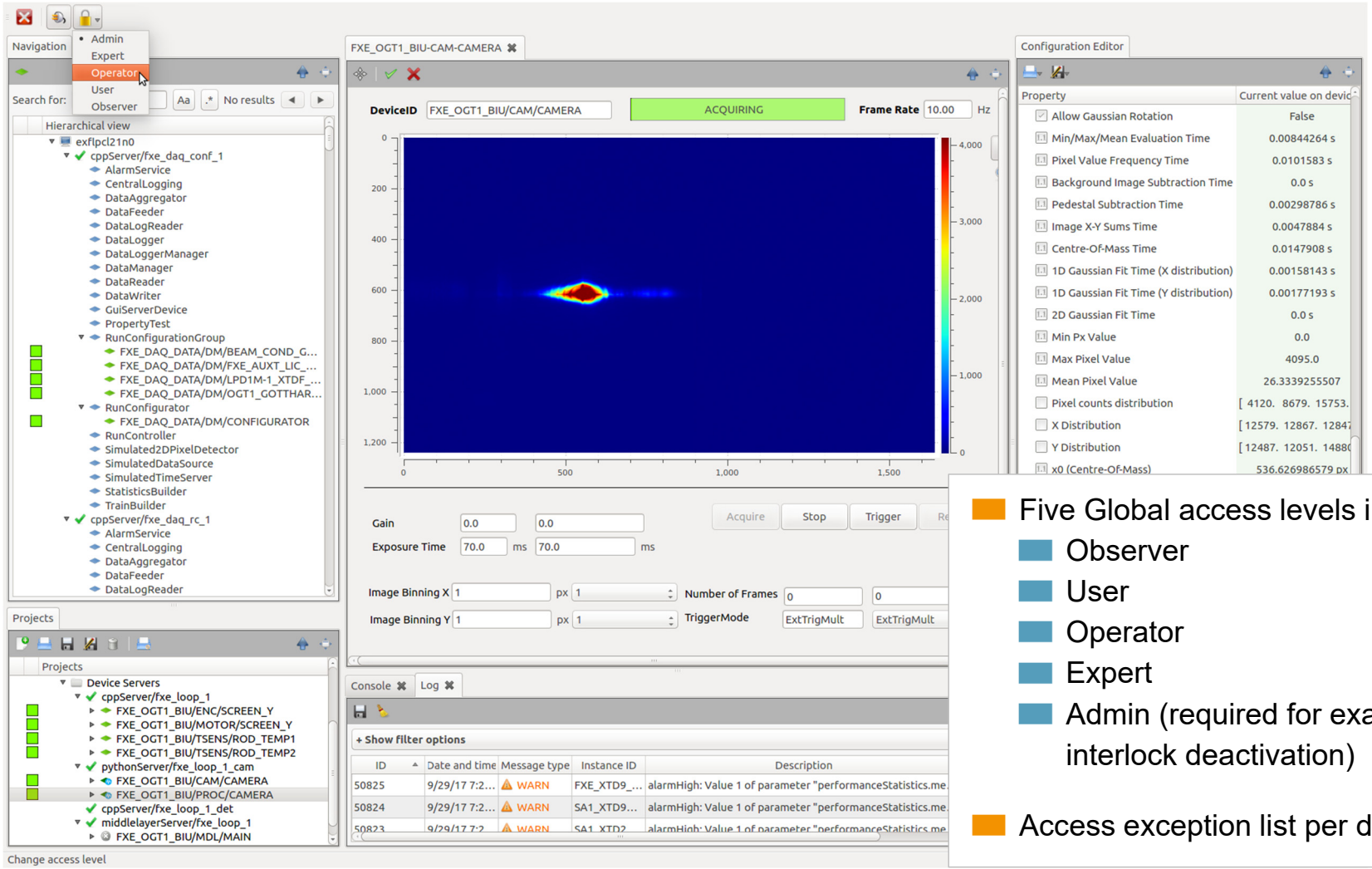
SCADA messaging



Example: **Stuxnet**(2010) attacking PLCs.

Basic Karabo authentication





- Five Global access levels in Karabo:
  - Observer
  - User
  - Operator
  - Expert
  - Admin (required for example for interlock deactivation)
- Access exception list per device.

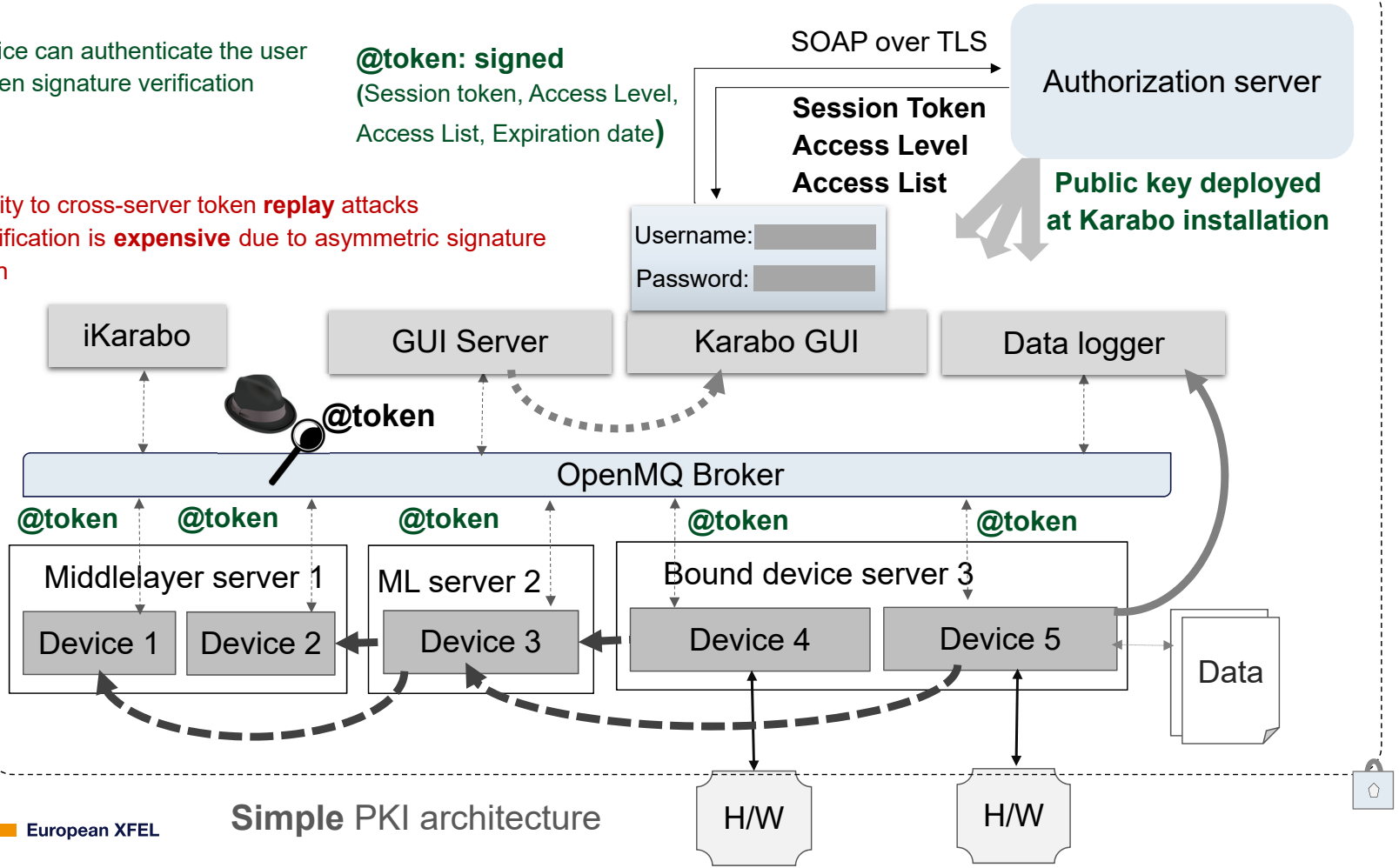
Benefits

- Every device can authenticate the user
- Offline token signature verification

Drawbacks

- Vulnerability to cross-server token **replay** attacks
- Token verification is **expensive** due to asymmetric signature verification

@token: signed
(Session token, Access Level, Access List, Expiration date)



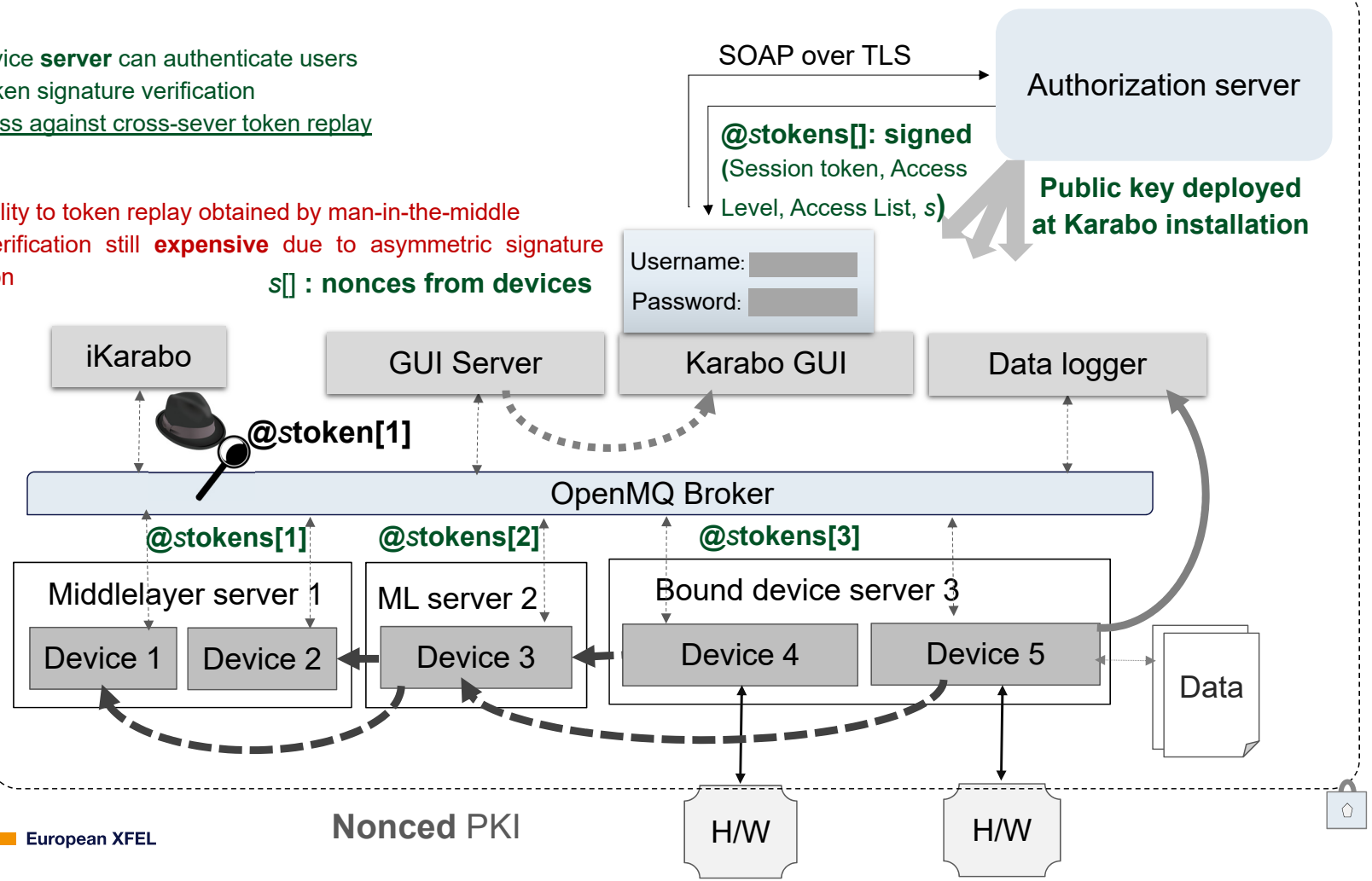
Simple PKI architecture

Benefits

- Every device **server** can authenticate users
- Offline token signature verification
- Robustness against cross-sever token replay

Drawbacks

- Vulnerability to token replay obtained by man-in-the-middle
- Token verification still **expensive** due to asymmetric signature verification



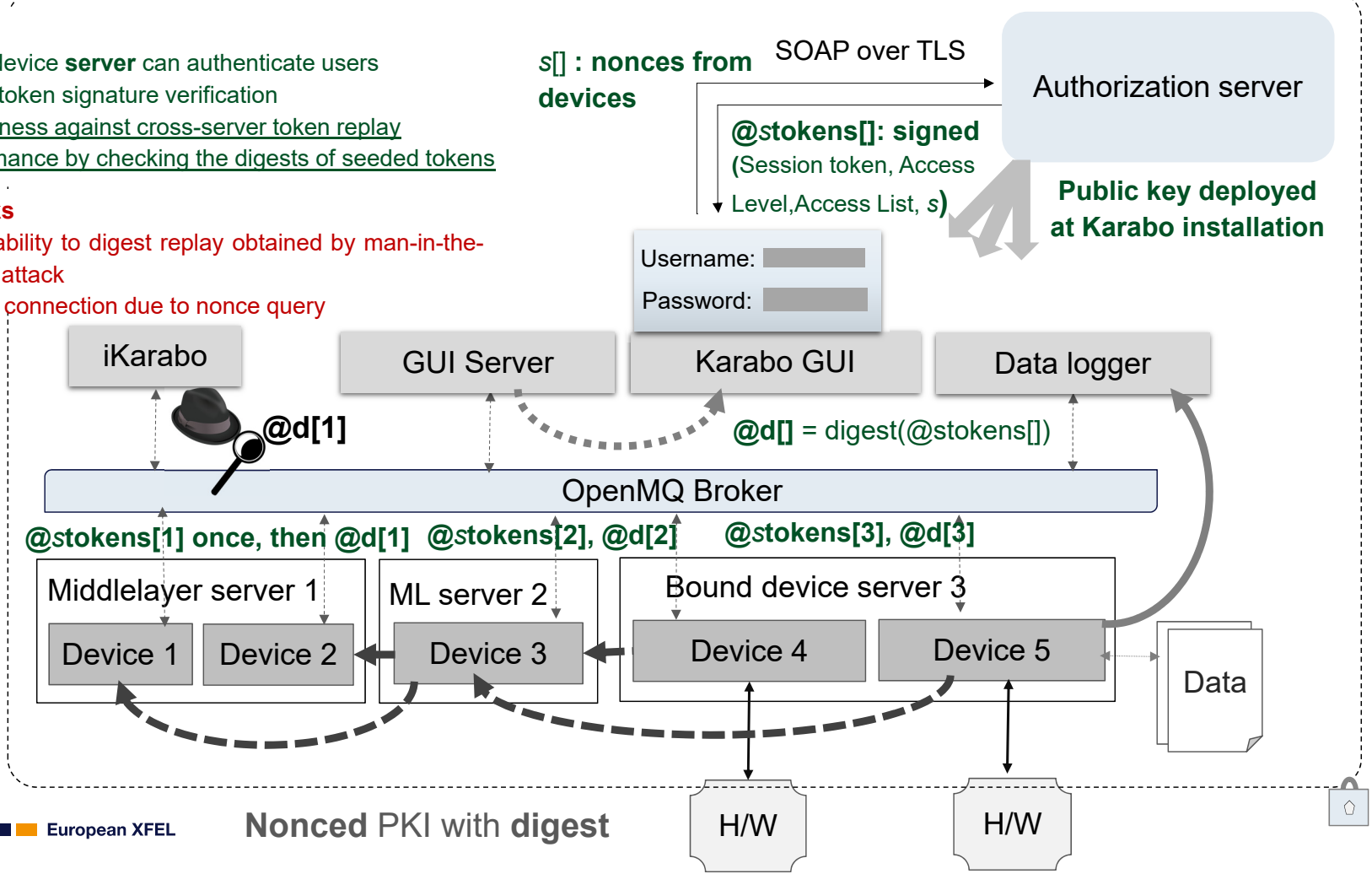
Nonced PKI

Benefits

- Every device **server** can authenticate users
- Offline token signature verification
- Robustness against cross-server token replay
- Performance by checking the digests of seeded tokens

Drawbacks

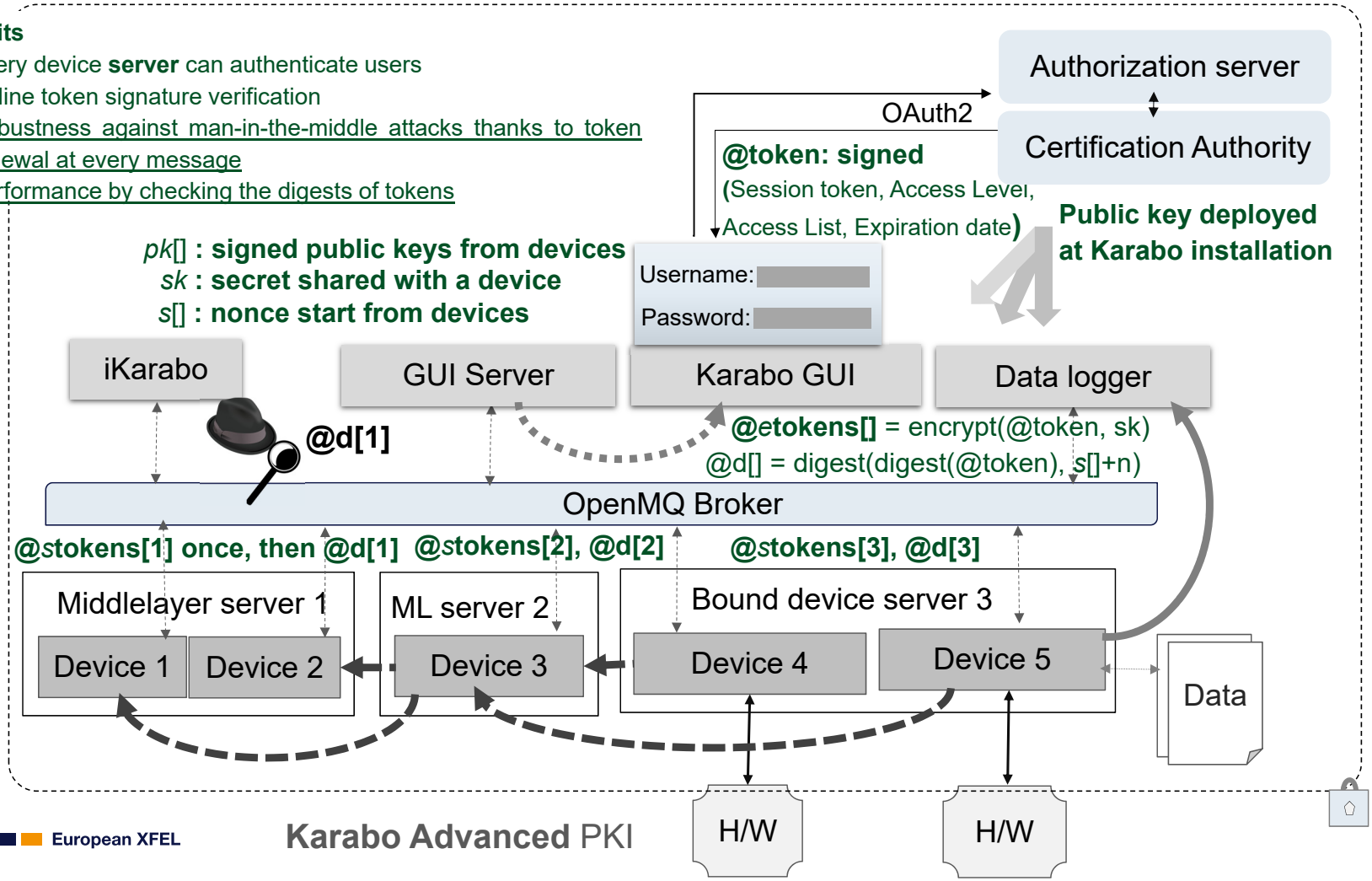
- Vulnerability to digest replay obtained by man-in-the-middle attack
- Slower connection due to nonce query



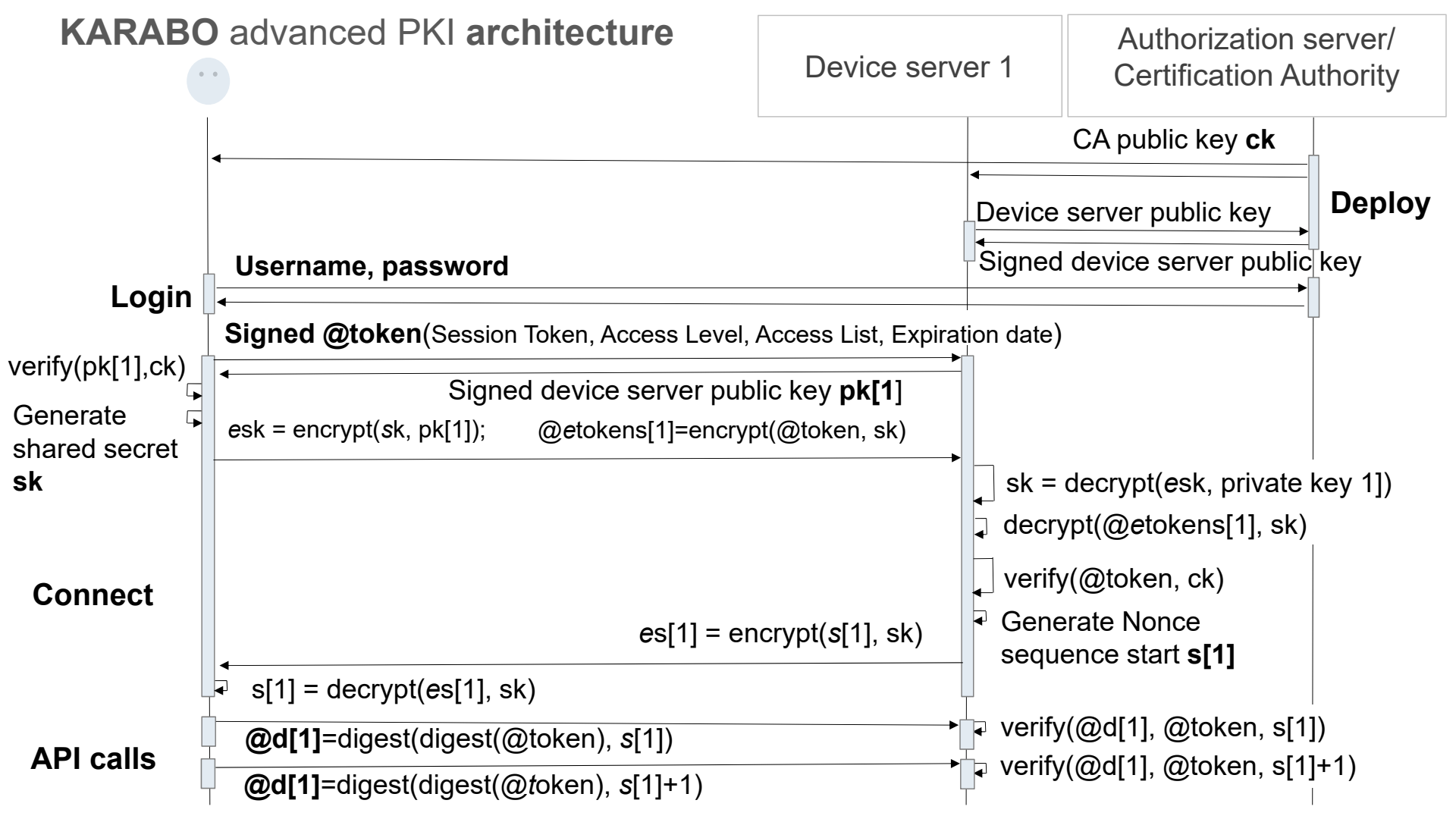
Nonced PKI with **digest**

Benefits

- Every device **server** can authenticate users
- Offline token signature verification
- Robustness against man-in-the-middle attacks thanks to token renewal at every message
- Performance by checking the digests of tokens



KARABO advanced PKI **architecture**



Conclusion

- We aim to protect the SCADA system **beyond general IT security**
- This prevents attackers from issuing valid SCADA messages or **escalading** their privileges
- Our Public-Key Infrastructure proposal for Karabo:
  - User shall access Karabo using a **token signed** by a Certification Authority
  - Device servers have their **public keys signed** by this same Certification Authority
  - Users communicate their session token only to certified device servers, encrypted with the device server public key.
  - The session token is only known to the CA, the user and the certified device servers
  - The nonce start is only known to the user and to a device server, encrypted by a **shared secret**
  - A **nonced digest** of the session token is sent within every API call, preserving performance.