JEDI: Many-to-Many End-to-End Encryption and Key Delegation for IoT

Sam Kumar, Yuncong Hu, Michael P Andersen, Raluca Ada Popa, David E. Culler

University of California, Berkeley

Accepted to USENIX Security 2019

What are the requirements of an end-to-end encryption (E2EE) protocol for large-scale IoT systems?

Large-Scale IoT Systems use *Publish/Subscribe*

- These systems **decouple senders from receivers**

Publish/Subscribe Example

Temperature Sensor

Write sodaHall/atrium/temp

Pub/Sub Broker

Read sodaHall/atrium/temp

Read sodaHall/atrium/temp

Person B

Cloud Services

Publish/Subscribe Example

Lighting

Read sodaHall/room410/lamp0/status

Write sodaHall/
room410/lamp0/status

Person A

Occupancy
Sensor

Write sodaHall/room465F/occupancy

Pub/Sub Broker

Read sodaHall/atrium/temp

Person B

Write sodaHall/atrium/temp

Read sodaHall/atrium/temp

Write sodaHall/atrium/hvac

Read sodaHall/
atrium/hvac

Temperature
Sensor

Heating, Ventilation,
and Cooling

Cloud Services

Publish/Subscribe Example

Lighting

Read sodaHall/room410/lamp0/status

Who is allowed to read which resources?

Person A

Write sodaHall/room465F/occupancy

Occupancy Sensor

Read sodaHall/atrium/temp

Person B

Write sodaHall/atrium/temp

Read sodaHall/atrium/temp

Write sodaHall/atrium/hvac

Read sodaHall/ atrium/hvac

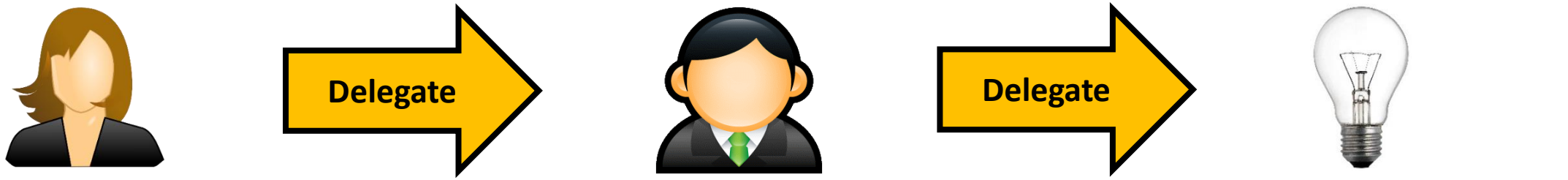
Pub/Sub Broker

Temperature Sensor

Heating, Ventilation, and Cooling

Cloud Services

IoT Systems use *Decentralized Delegation*



- Decentralized delegation is an old idea (SPKI/SDSI [CECF01])
- Recently, Vanadium [TS16] and BOSSWAVE [AKCCK17] introduced decentralized delegation in smart buildings
- It is the state-of-the-art for access control in large-scale IoT systems

JEDI: **J**oining **E**ncryption and **D**elegation for **I**oT

Joining **E**ncryption and **D**elegation for **I**oT

JEDI is an *end-to-end encryption* (E2EE) protocol that:

- Allows senders and receivers to be decoupled as in publish/subscribe
- Supports decentralized delegation
- Can run on resource-constrained IoT devices

Roadmap

1. **Requirements of E2EE for IoT**

2. JEDI's approach
   a) Decoupled senders/receivers and decentralized deelgation
   b) Expiry
   c) Resource-constrained devices

3. Evaluation results

Roadmap

1. Requirements of E2EE for IoT

2. **JEDI's approach**
   a) Decoupled senders/receivers and decentralized delegation
   b) Expiry
   c) Resource-constrained devices

3. Evaluation results

JEDI also supports **anonymous signatures** and **revocation**

- Not described in this talk
- See the paper for details

E2EE for Publish/Subscribe

Lighting

Read sodaHall/room410/lamp0/status

Occupancy Sensor

Write sodaHall/room465F/occupancy

Write sodaHall/atrium/temp

Temperature Sensor

Read sodaHall/ atrium/hvac

Heating, Ventilation, and Cooling

Pub/Sub Broker

Write sodaHall/ room410/lamp0/status

Person A

Read sodaHall/atrium/temp

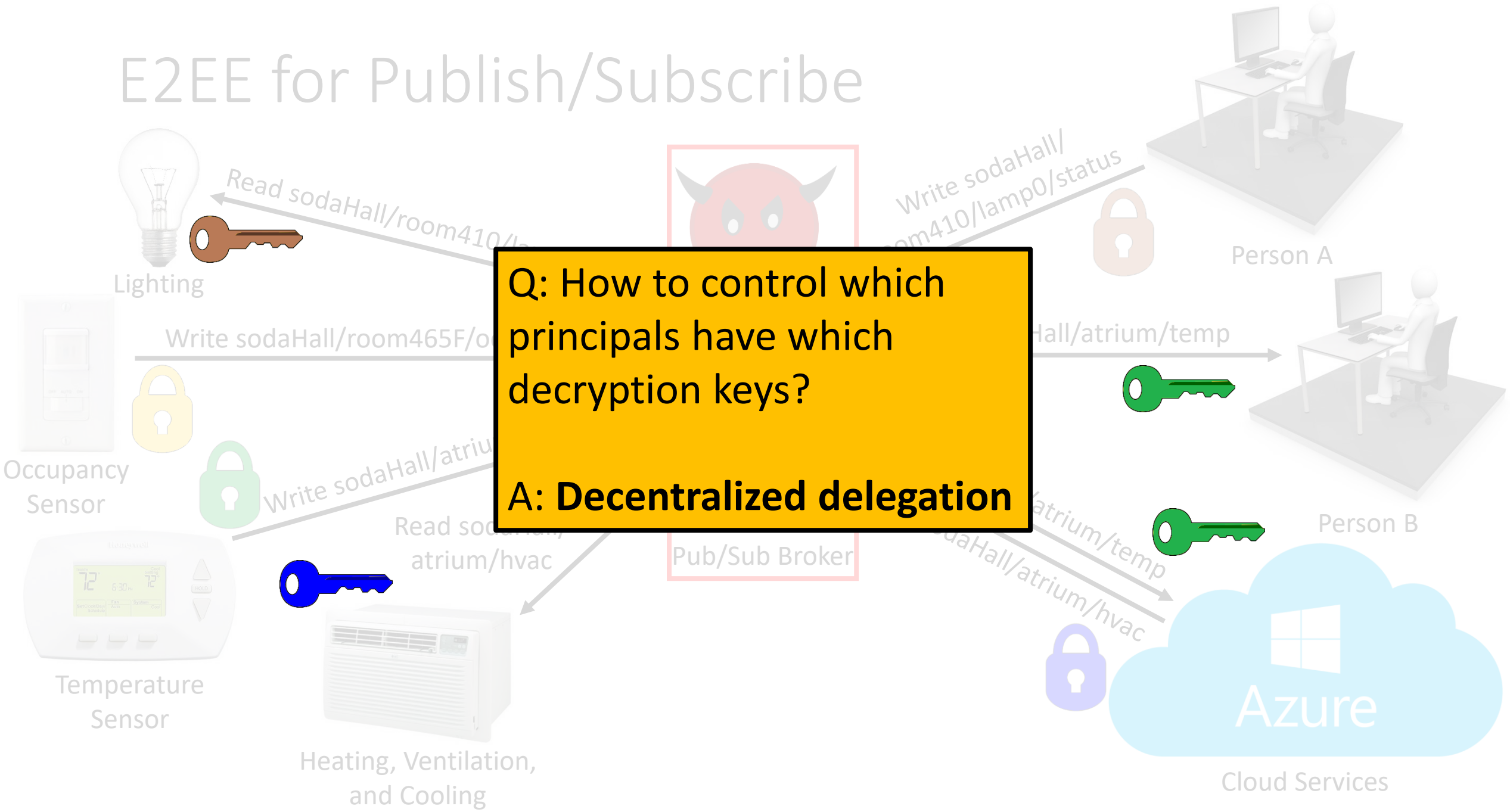
Person B

Read sodaHall/atrium/temp

Write sodaHall/atrium/hvac

Cloud Services

E2EE for Publish/Subscribe



Q: How to control which principals have which decryption keys?

A: **Decentralized delegation**

Decentralized Delegation Example

Campus Facilities Manager

Building Manager can read sodaHall/*

Building Manager

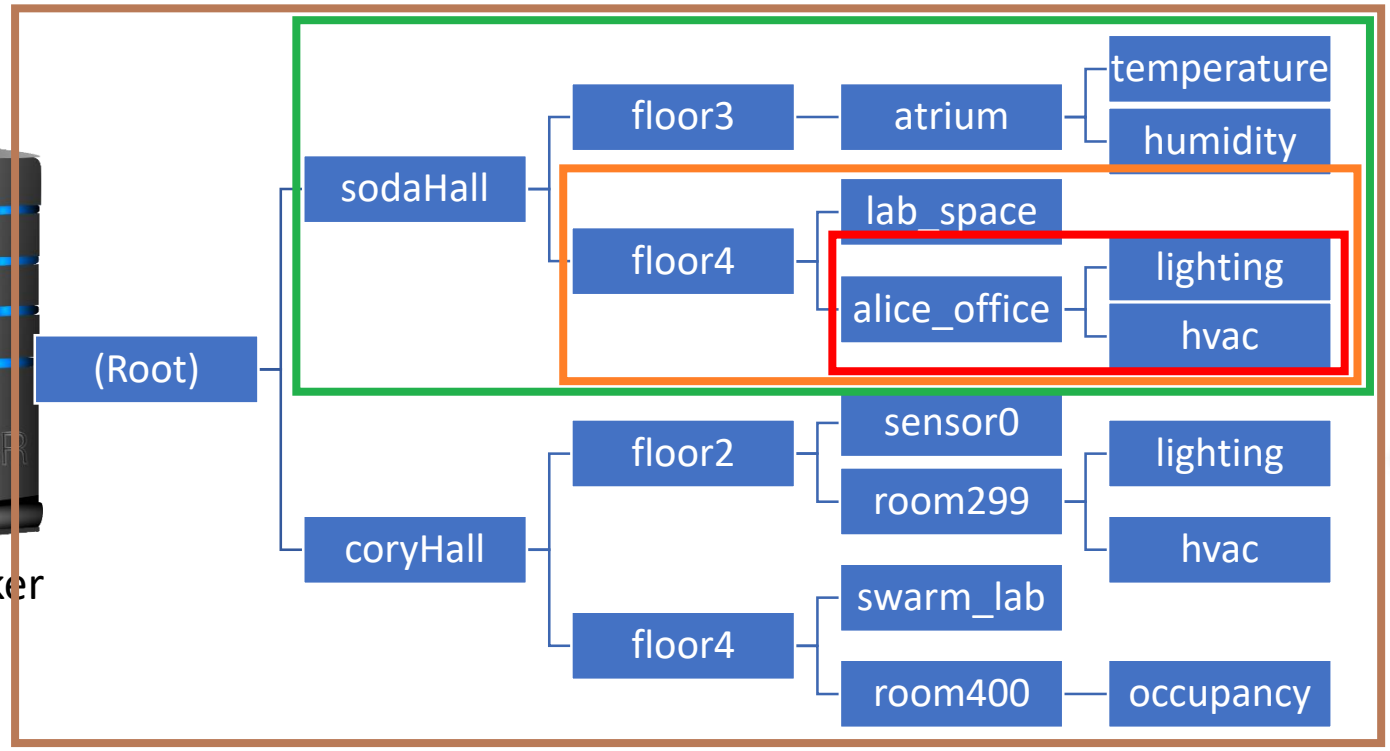
Lab Director can read sodaHall/floor4/*

Lab Director

Pub/Sub Broker

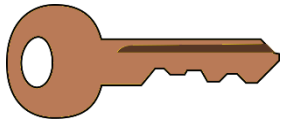
Alice

Alice can read sodaHall/floor4/alice_office/*



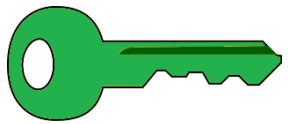
E2EE with Decentralized Delegation

Campus Facilities Manager



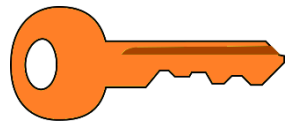
Key for *

Building Manager



Key for sodaHall/*

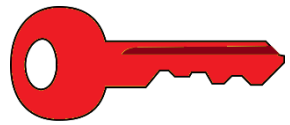
Lab Director



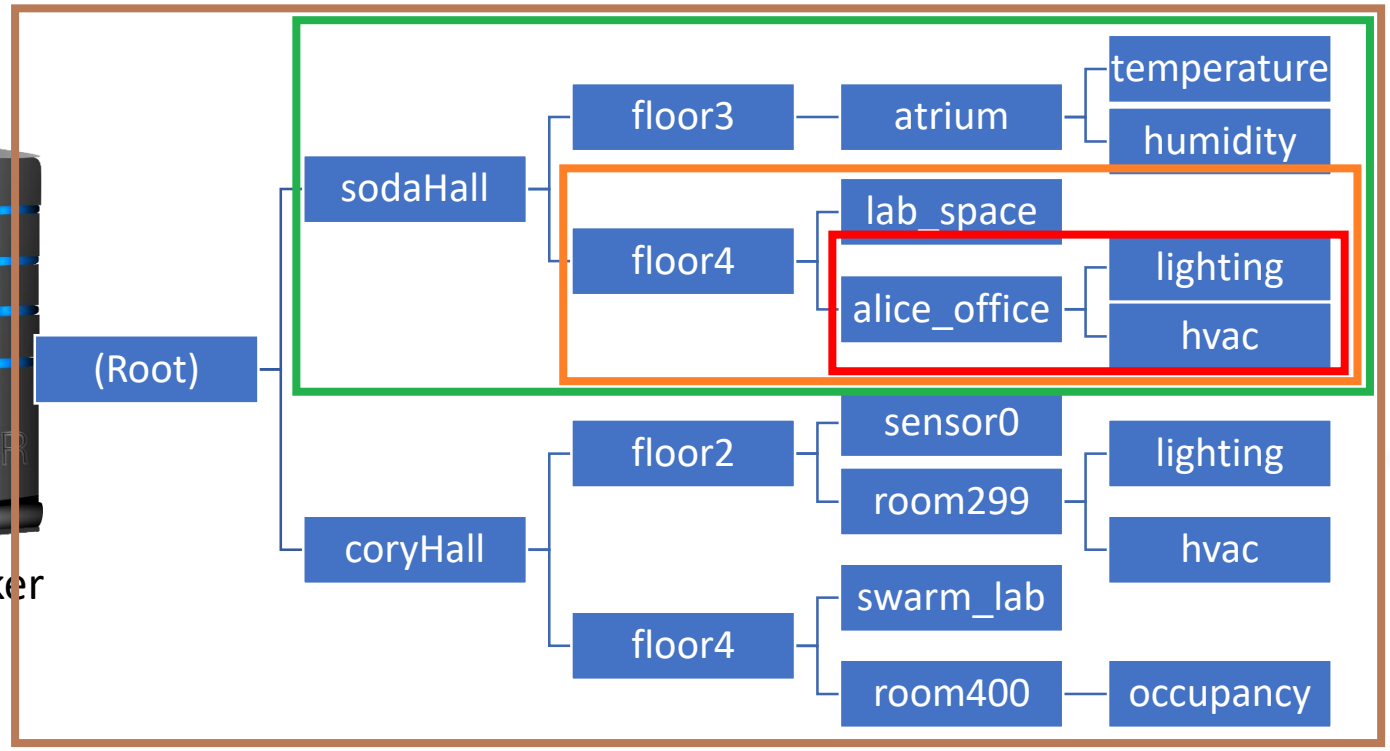
Key for sodaHall/floor4/*

Pub/Sub Broker

Alice

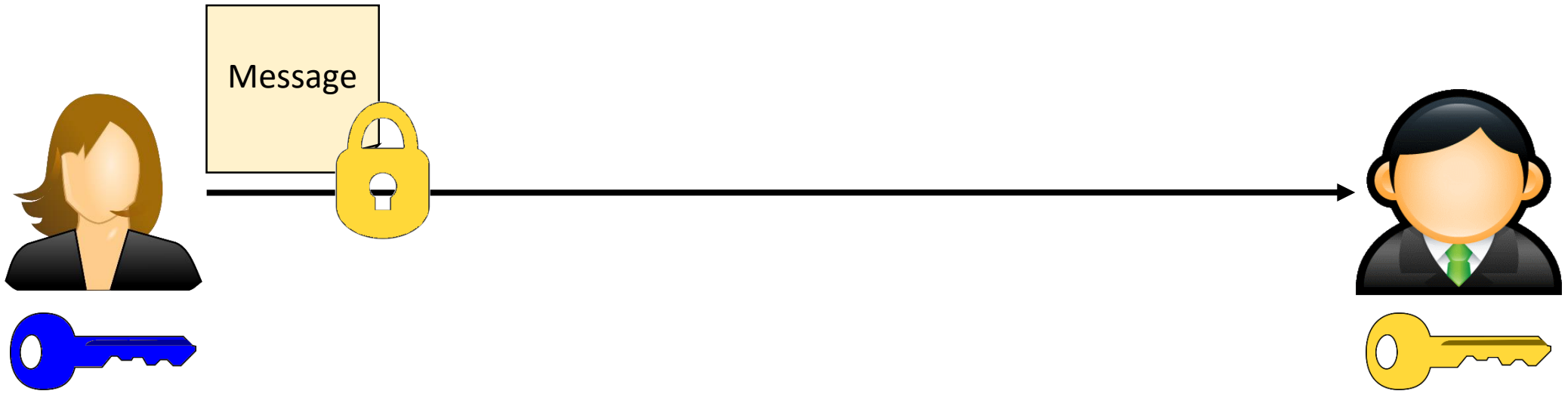


Key for sodaHall/floor4/alice_office/*



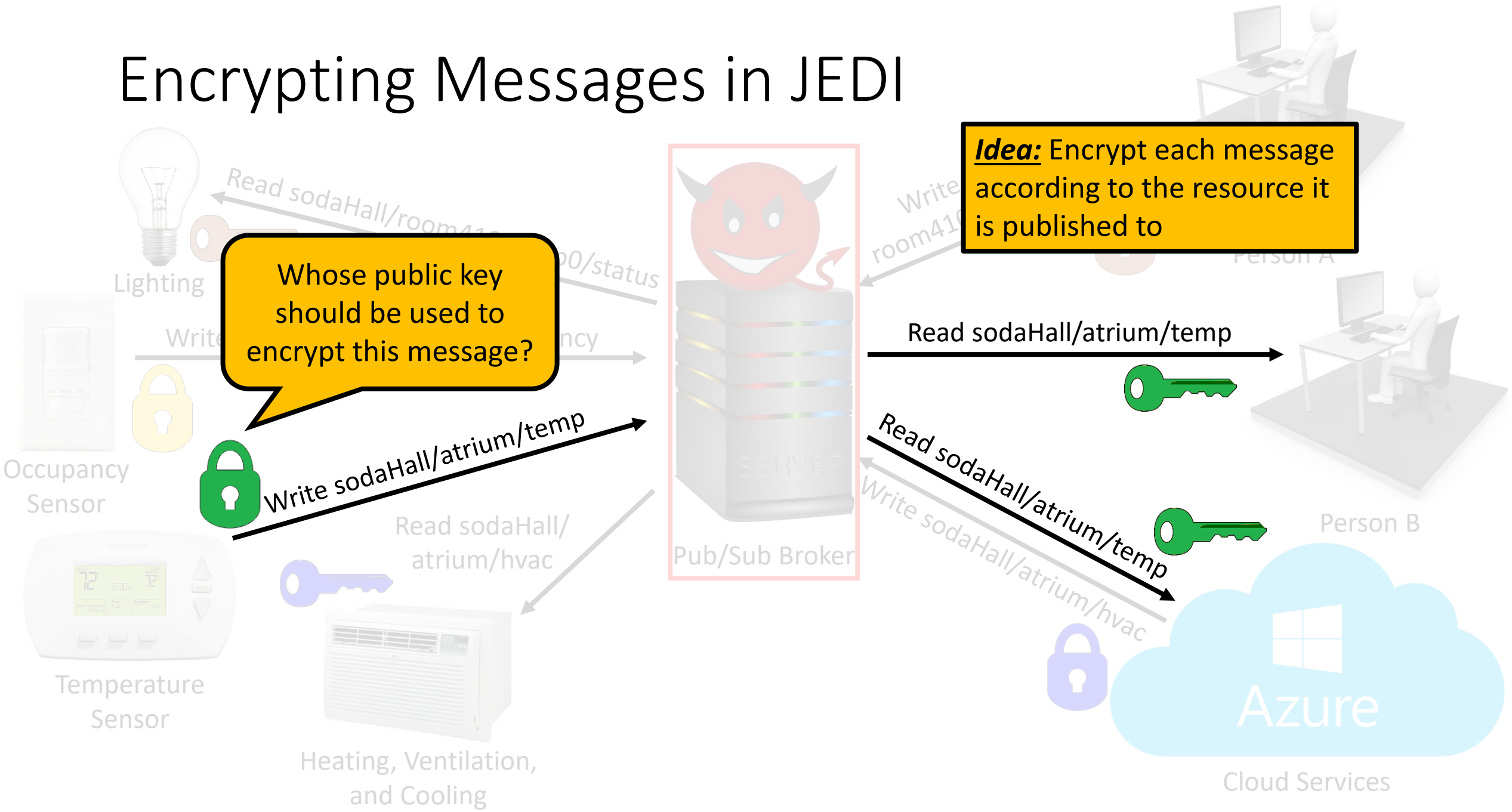
How to actually encrypt data?

Strawman #1: Existing protocols (e.g., SSL/TLS)



- Supports *one-to-one* communication between two parties
- Problem: not suitable for *many-to-many* communication

Encrypting Messages in JEDI

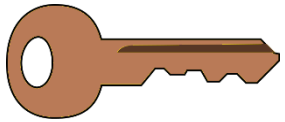


Strawman #2: Public Key for each Resource

- Generate an independent public-private keypair for each resource
- Problem: it requires the resource hierarchy to be fixed in advance

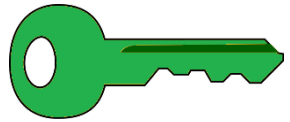
Problems with Strawman #2

Campus Facilities Manager



Key for *

Building Manager

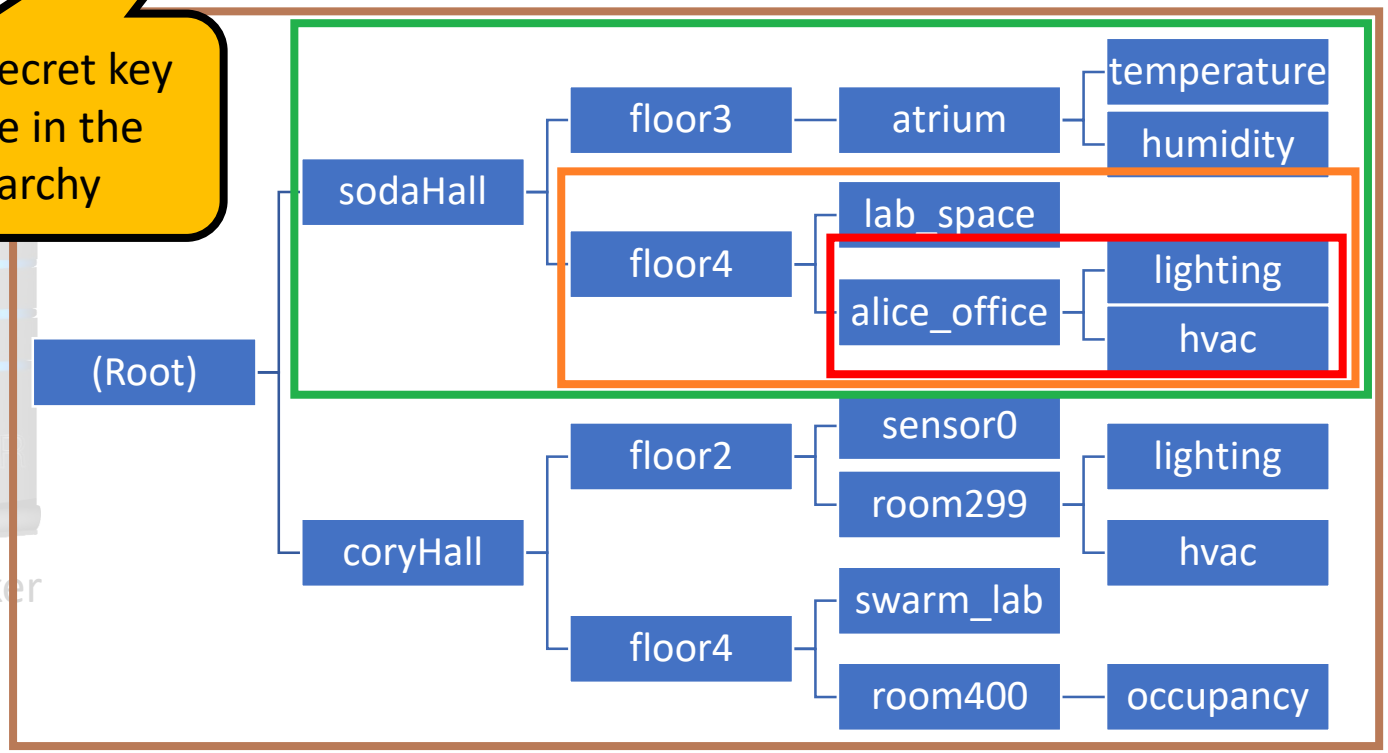


Key for sodaHall/*

Contains the secret key for each node in the entire hierarchy

Contains the secret key for each node in the sodaHall/* subtree

Instead, the Building Manager should be able to choose the structure of the sodaHall/* subtree



Pub/Sub Broker

Key for sodaHall/floor4/*

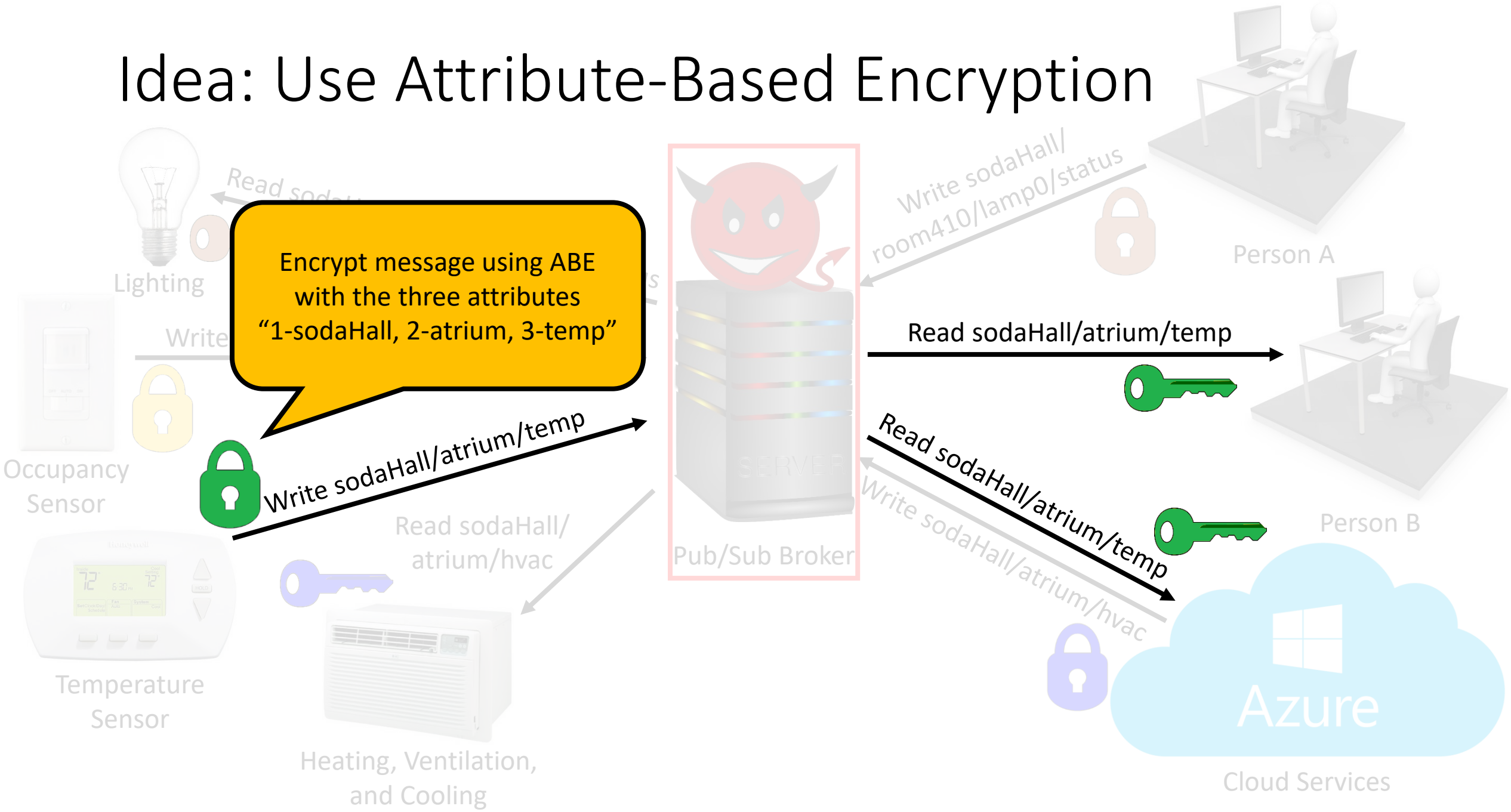
alice_office/*

Alice

Idea: Use Attribute-Based Encryption

- In Attribute-Based Encryption (**ABE**) [GPSW06]:
  - Messages are encrypted with a set of arbitrary strings called *attributes*
  - Keys correspond to *policies*
  - A key can decrypt ciphertexts whose attributes satisfy its policy
- A key for a policy can be used to generate a key for a more restrictive policy
- This is not efficient—we will improve the efficiency later

Idea: Use Attribute-Based Encryption

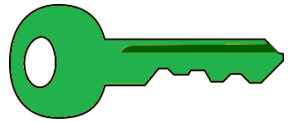


Idea: Use Attribute-Based Encryption

Campus Facilities Manager

Key for *

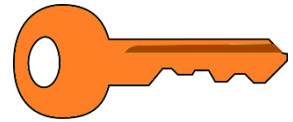
Building Ma



Key for sodaHall/*

ABE key for the policy "1-sodaHall"

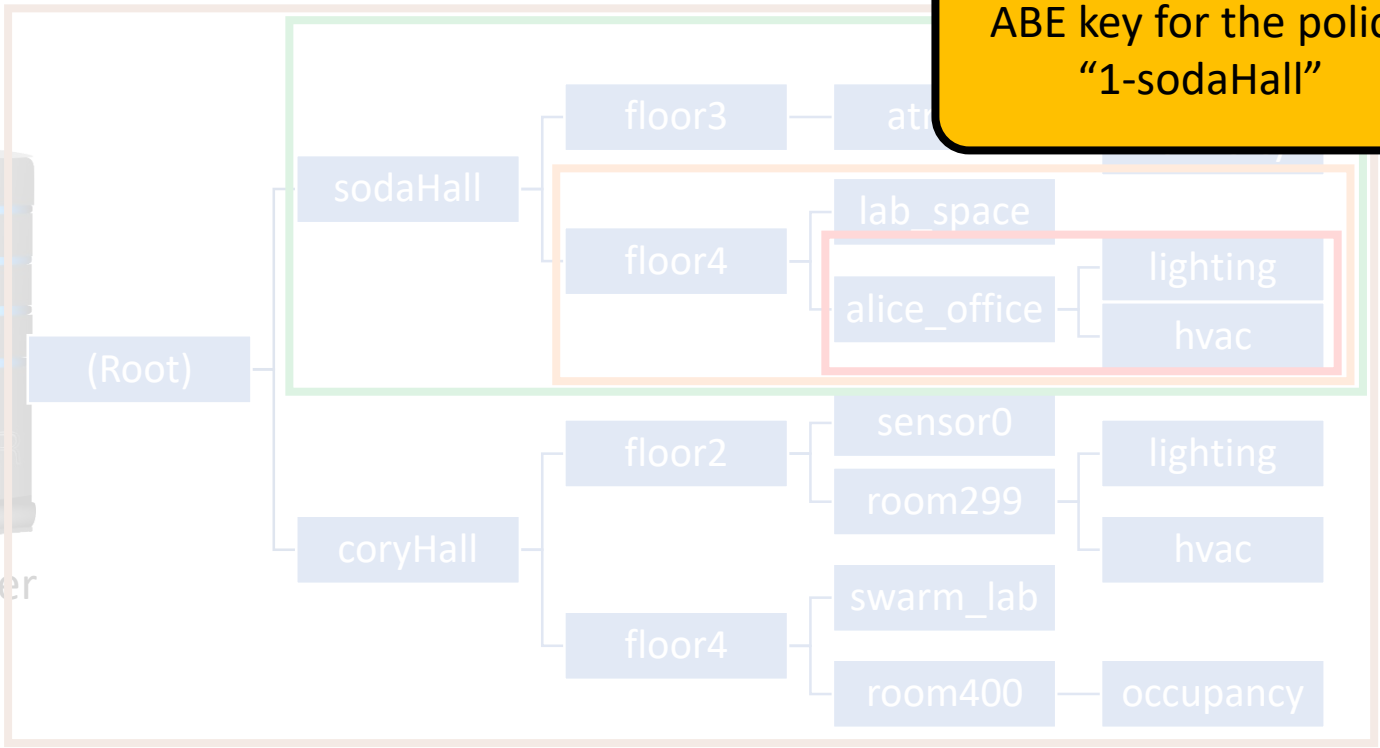
Lab



Key for sodaHall/floor4/*

ABE key for the policy "1-sodaHall AND 2-floor4"

Pub/Sub Broker



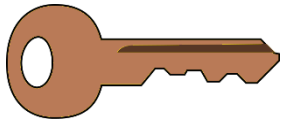
Alice

Key for sodaHall/floor4/alice_office/*

Expiry

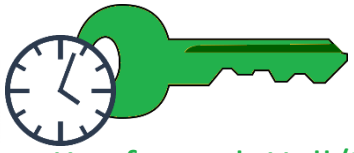
Expiry

Campus Facilities Manager



Key for *

Building Manager



Key for sodaHall/*
Expires Dec 2021

Lab Director

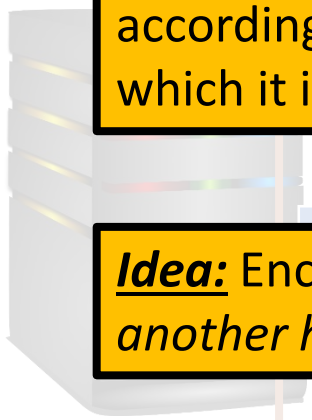
Key for sodaHall/floor4/*
Expires Jun 2020

Cannot decrypt messages published after December 2021

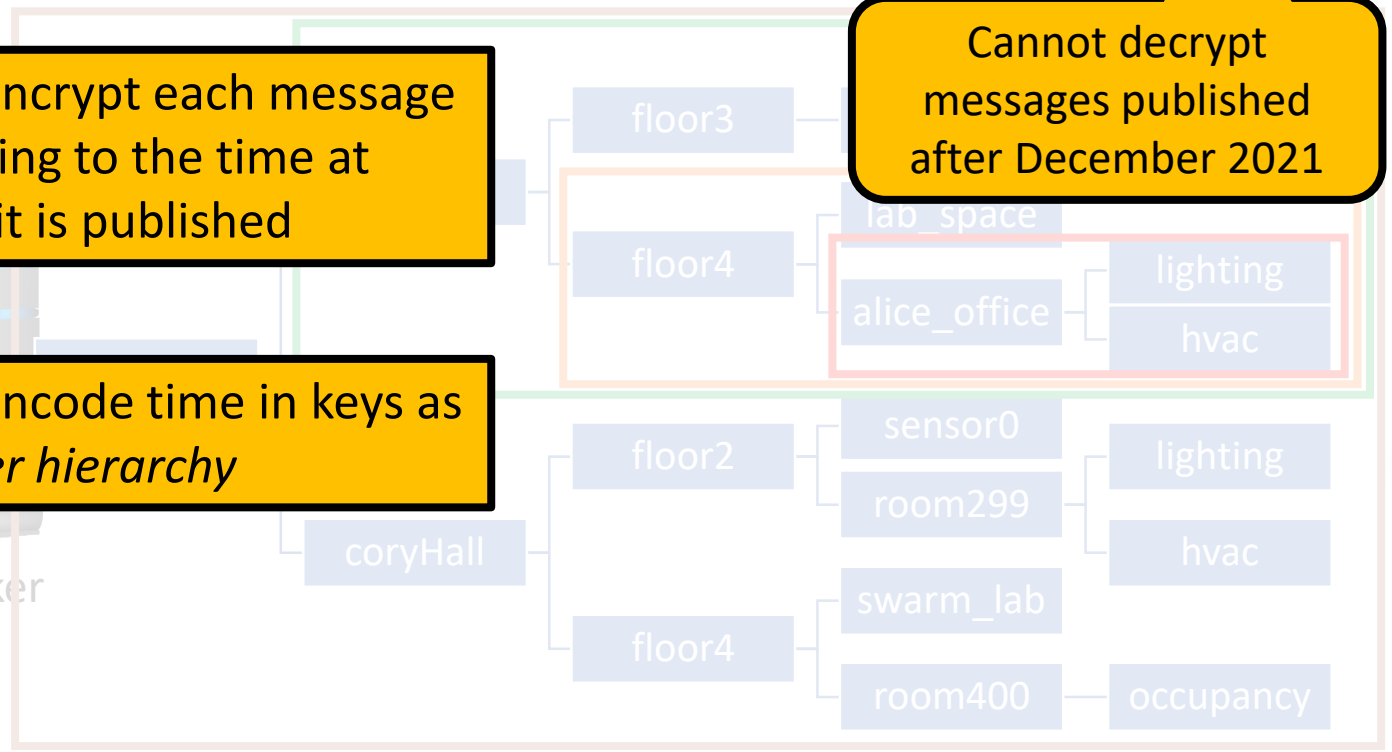
Expiry time can be restricted at each delegation

Idea: Encrypt each message according to the time at which it is published

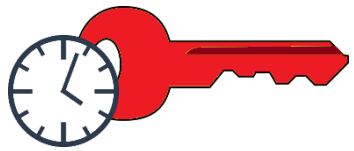
Idea: Encode time in keys as *another hierarchy*



Pub/Sub Broker

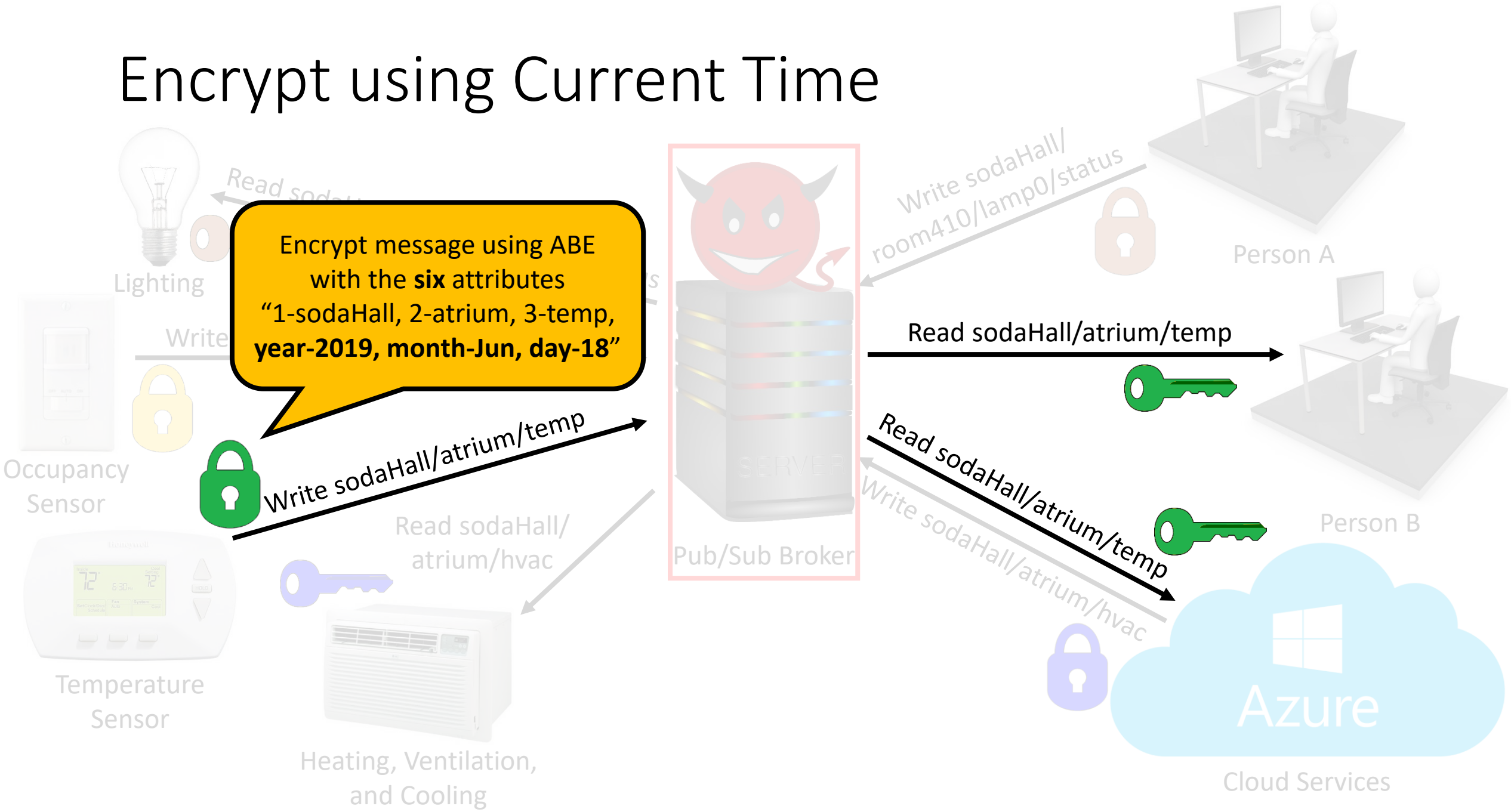


Alice

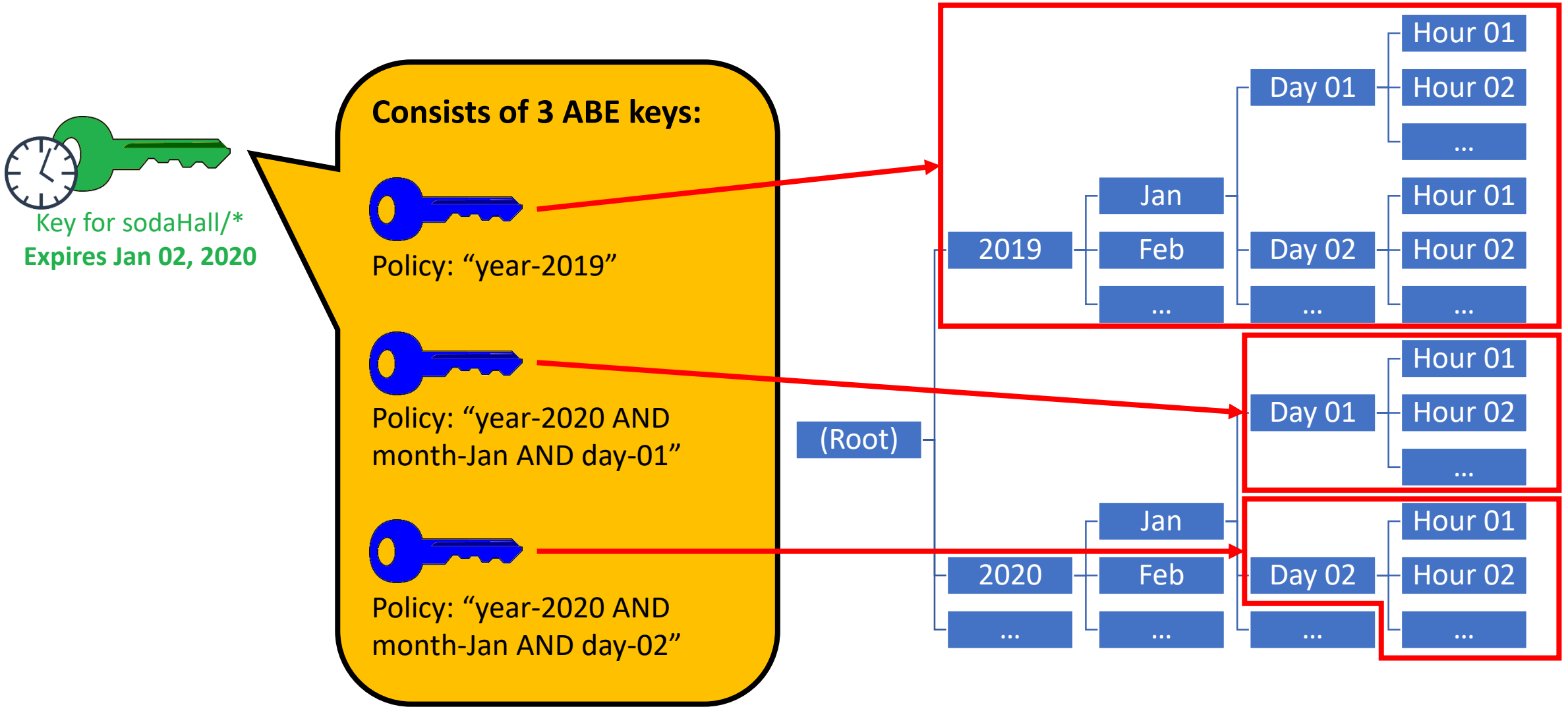


Key for sodaHall/floor4/alice_office/*
Expires Aug 2019

Encrypt using Current Time



Time is another Hierarchy



Support for Resource-Constrained Devices

Resource Constraints

We tested JEDI on Hamilton [KACKZMC18], a low-power sensor

Laptop

Smartphone

Smart Home Appliance

Wearable

Ultra Low-Power Deeply Embedded Sensor

More Powerful

100,000 DMIPS
10 GiB RAM

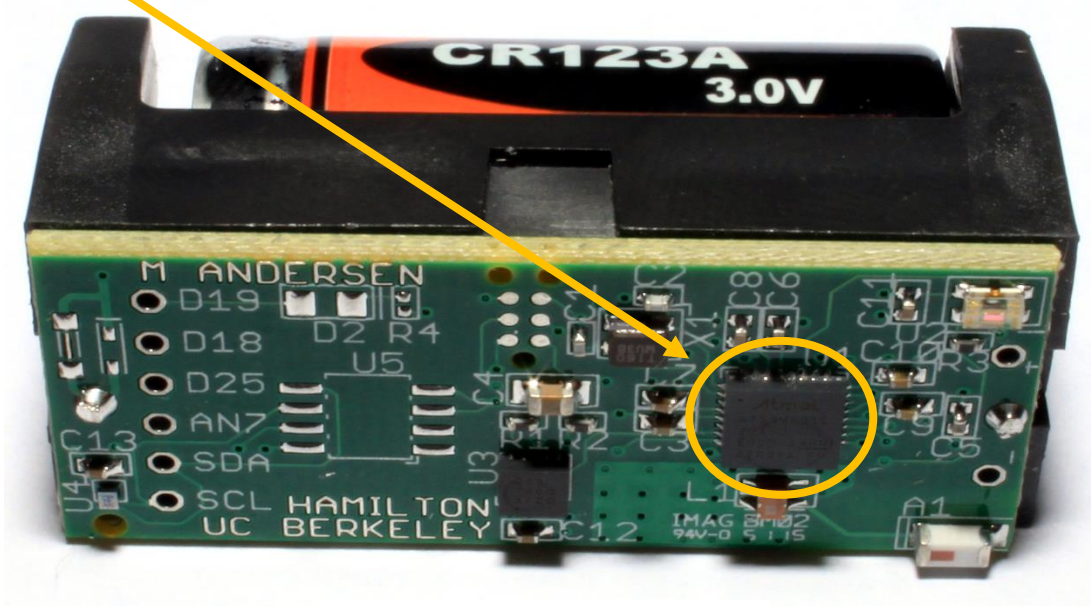
Less Powerful

50 DMIPS
32 KiB RAM
Power Constraints

- JEDI must work across the spectrum of devices…
- … without sacrificing security on all devices for the less powerful ones

Hamilton platform [KACKZMC18]

- Based on the Atmel SAMR21 SoC
  - 32-bit ARM Cortex M0+ @ 48 MHz
  - 32 KiB Data Memory (RAM)
- Transducers (accelerometer, magnetometer, temperature, humidity, and infrared)
- CR123A Lithium battery with 252 J (1400 mAh @ 3.0 V)



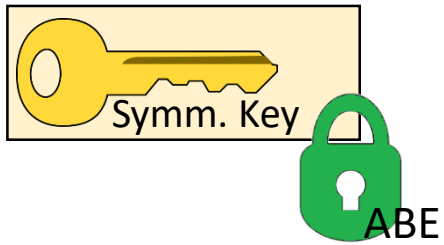
≈ 5 centimeters

How can we reduce the cost of ABE for constrained platforms?

Hybrid Encryption

Temperature Sensor

Write sodaHall/atrium/temp



Pub/Sub Broker

Read sodaHall/atrium/temp

Person B

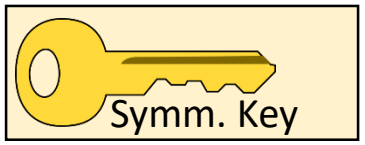
Hybrid Encryption

Temperature Sensor

Write sodaHall/atrium/temp

Pub/Sub Broker

Read sodaHall/atrium/temp



Person B

Hybrid Encryption

Temperature Sensor

Write sodaHall/atrium/temp

Message

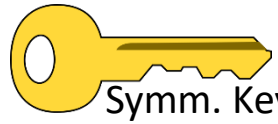
Symm.

Pub/Sub Broker

Read sodaHall/atrium/temp

Person B

ABE Key



Symm. Key

Hybrid Encryption

Temperature Sensor

Write sodaHall/atrium/temp

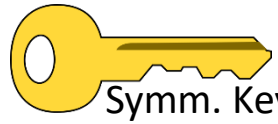
Pub/Sub Broker

Read sodaHall/atrium/temp

Person B

Message

ABE Key



Symm. Key

Rotate keys whenever a principal may lose access (e.g., end of each hour)

Energy Cost of ABE

- Due to hybrid encryption, we invoke ABE *rarely*
- Regardless, **ABE dominates power consumption**

- A single ABE [GPSW06] encryption would take 4 minutes on Hamilton
  - Active CPU current is 10.2 mA, so this is **0.68 mAh**
- Battery capacity is 252 J
  - Supply voltage is 3.0 V, so this is **1400 mAh**

- At one ABE encryption per hour, a device won't even last 100 days

Cheaper Alternatives to ABE [GPSW06]

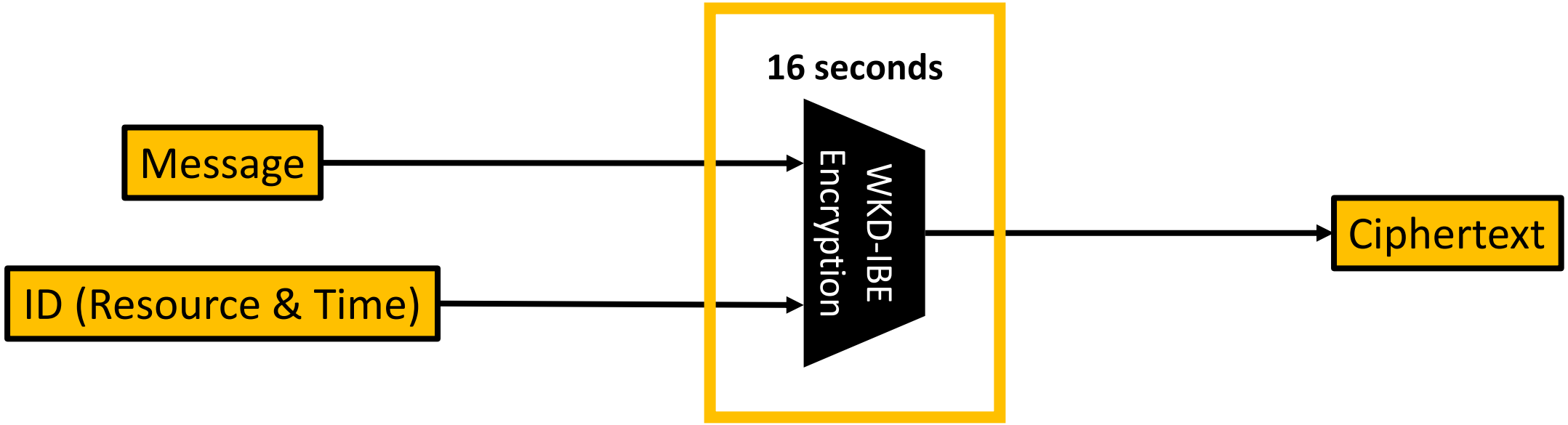
- Identity-Based Encryption (IBE) [Shamir84, BF01]
  - Cannot support delegation
- Hierarchical IBE (HIBE) [HL02, GS02, BBG05]
  - Can support either a resource hierarchy or time hierarchy
  - Cannot support both hierarchies simultaneously
- Wildcard Key Derivation IBE (WKD-IBE) [AKN07]
  - Enhanced HIBE with more general key delegation within a single hierarchy
  - We can use WKD-IBE in a nonstandard way to realize JEDI

Energy Cost of WKD-IBE

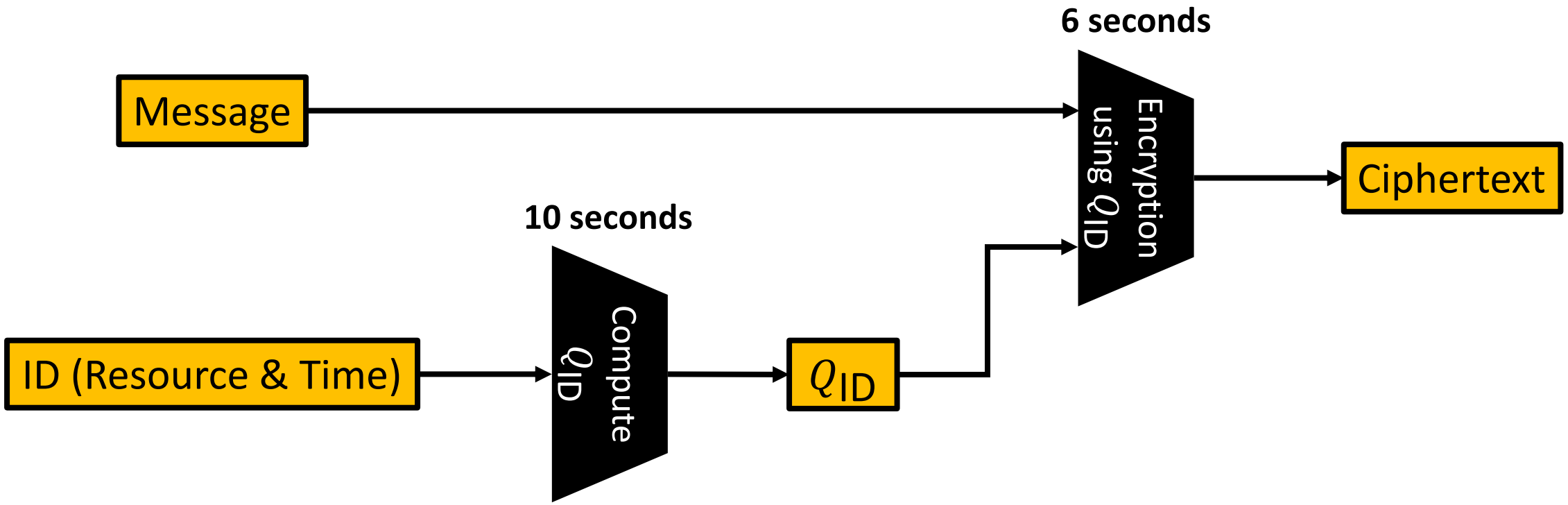
- WKD-IBE is an order of magnitude more efficient than ABE
  - Encryption time decreases from 240 s (ABE) to 16 s (WKD-IBE)

- Another 2-3x speedup comes from *non-black-box* usage of WKD-IBE

WKD-IBE Encryption

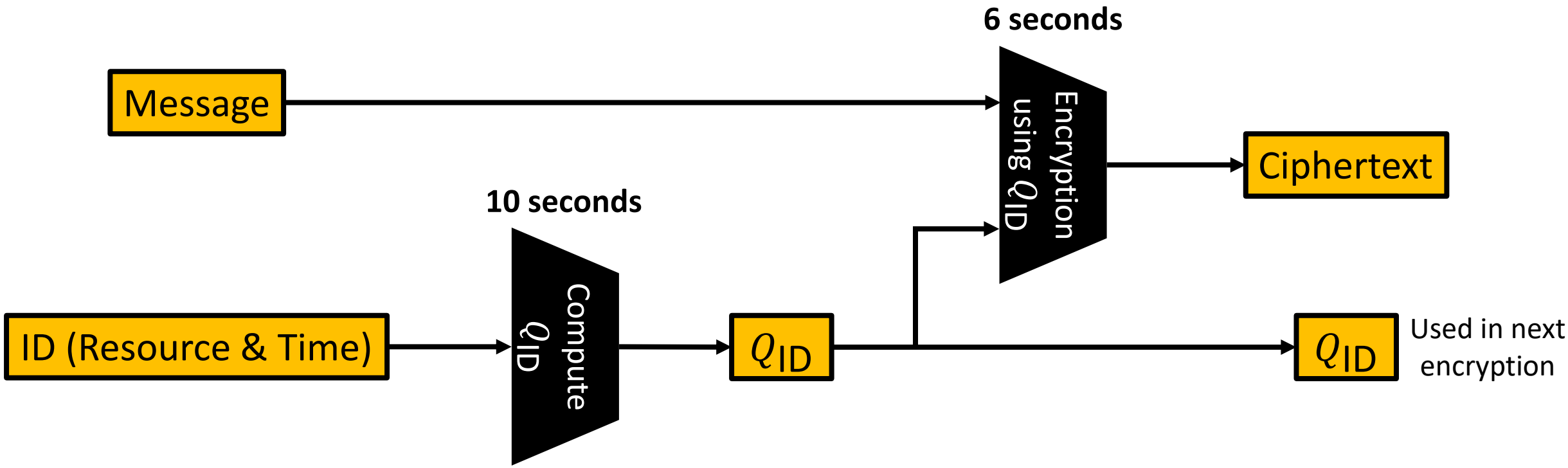


WKD-IBE Encryption



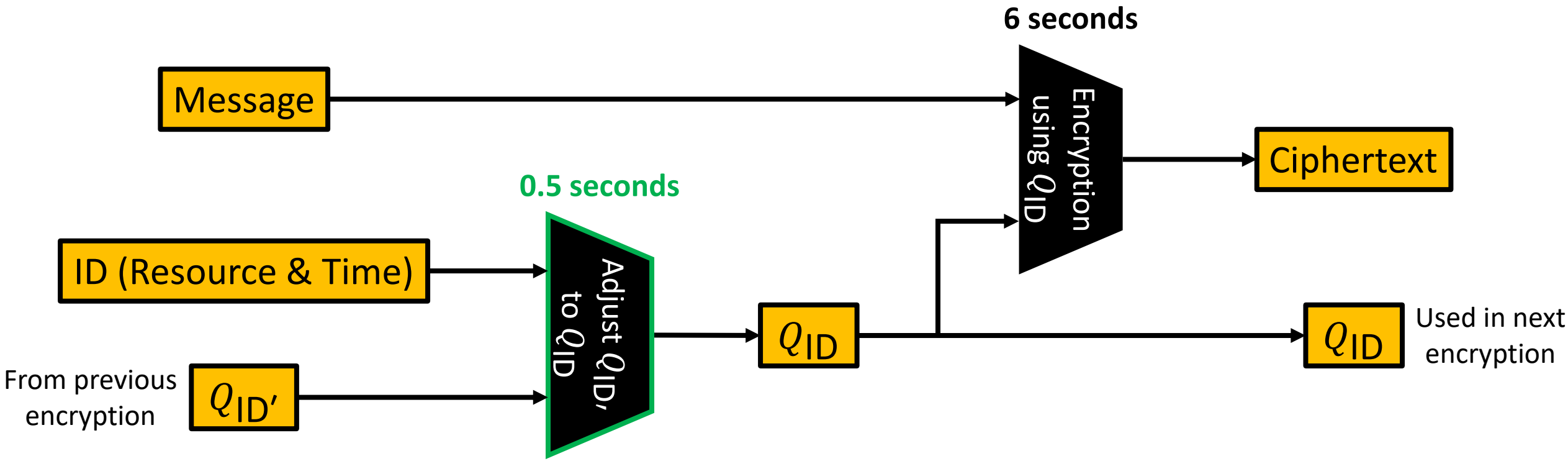
Observation: IDs in adjacent encryptions differ in only a few attributes

Non-Black-Box Usage of WKD-IBE



Observation: IDs in adjacent encryptions differ in only a few attributes

Non-Black-Box Usage of WKD-IBE



Observation: IDs in adjacent encryptions differ in only a few attributes

Overall Improvement

- 37x overall improvement (240 s → 6.5 s) from:
  - Using WKD-IBE instead of ABE
  - Making non-black-box use of WKD-IBE

- Full paper (on arXiv) formally explains the crypto optimizations and shows why they are secure

Roadmap

1. Requirements of E2EE for IoT

2. **JEDI's approach**
   a) Decoupled senders/receivers and decentralized delegation
   b) Expiry
   c) Resource-constrained devices

3. Evaluation results

JEDI also supports **anonymous signatures** and **revocation**

- Not described in this talk
- See the paper for details

Roadmap

1. Requirements of E2EE for IoT

2. JEDI’s approach

3. **Evaluation results**
   a) Implementation
   b) Evaluation

Implementation

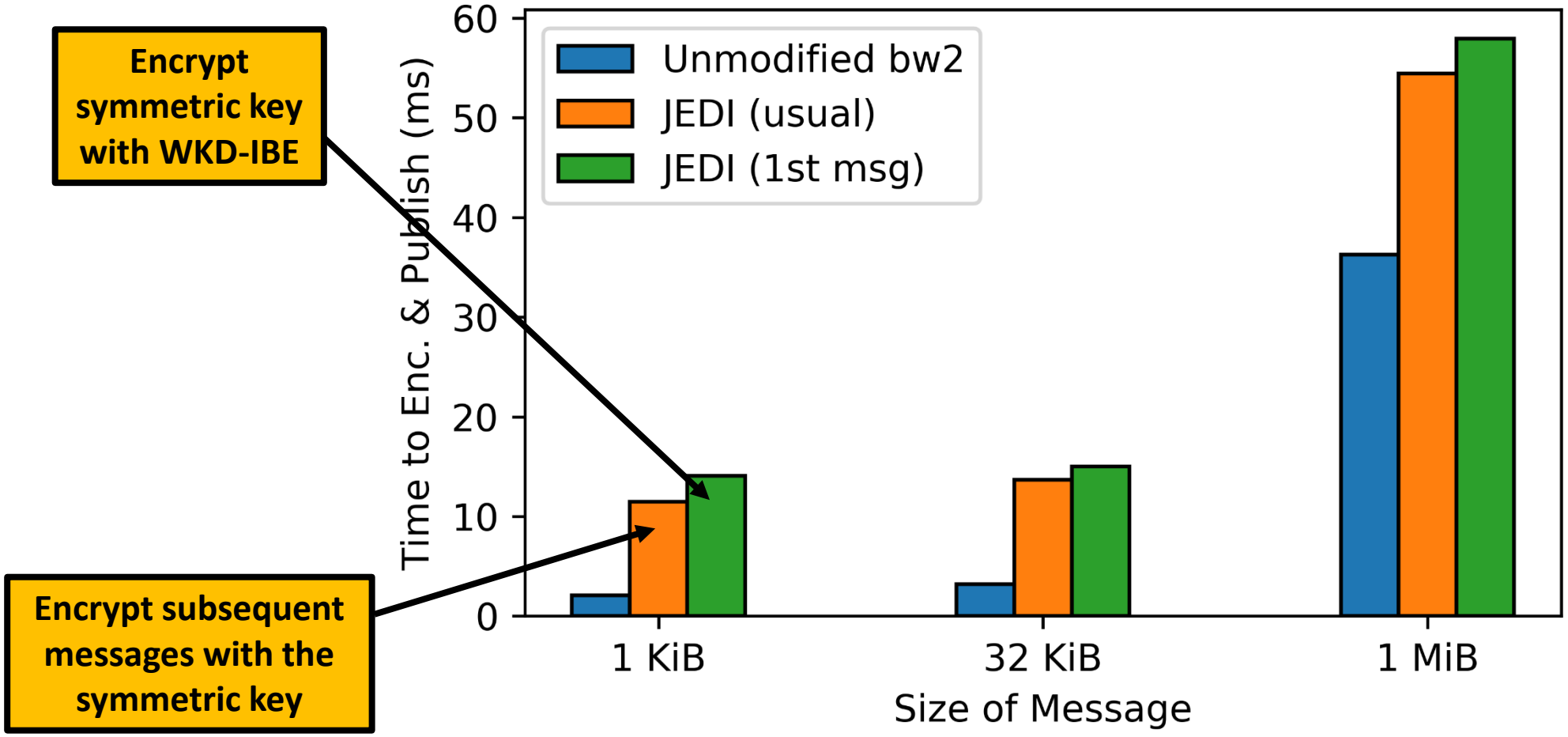
Two parts of JEDI's implementation:

1. JEDI Cryptography Library
   - Includes assembly optimizations for ARM Cortex-M0+
   - 4-5x performance improvement over pure C/C++

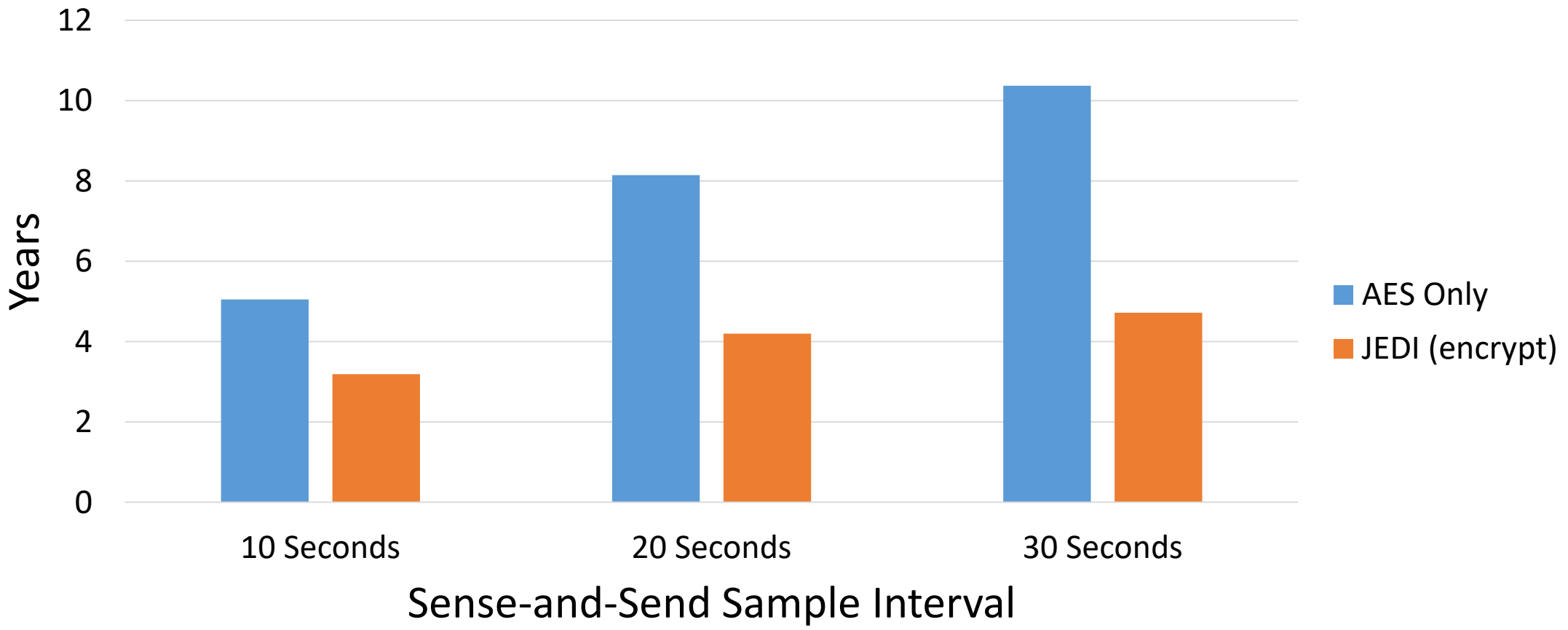
2. JEDI Protocol Prototype
   - Implemented for bw2 [AKCFCP17], a messaging system for smart cities

Both are open-source and available on GitHub

JEDI Applied to bw2 (running on a Laptop)



Estimated Battery Life on a Hamilton Sensor



Conclusion

JEDI is an end-to-end encryption protocol for large-scale IoT systems. It:

- Allows senders and receivers to be decoupled as in publish/subscribe
- Supports decentralized delegation with expiry
- Can run on devices across the spectrum of IoT resource constraints

Conclusion

Our JEDI implementation is open-source:

- https://github.com/ucbrise/jedi-pairing
- https://github.com/ucbrise/jedi-protocol

Preprint of our paper is available on arXiv:

- https://arxiv.org/abs/1905.13369

Any Questions?

This material is based on work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1752814. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.