Attestation in the Internet of Things (IoT)

Mobile Systems Security 2017

Optional lecture

February 16th, 2017

Andrew Paverd
Aalto University
andrew.paverd@aalto.fi
*(***Contributors: N. Asokan, Lucas Davi)**

Aalto University

UNIVERSITY OF HELSINKI

You will learn about:

- Why do we need (remote) attestation in IoT?
- What technologies can we use for this?

Motivating example

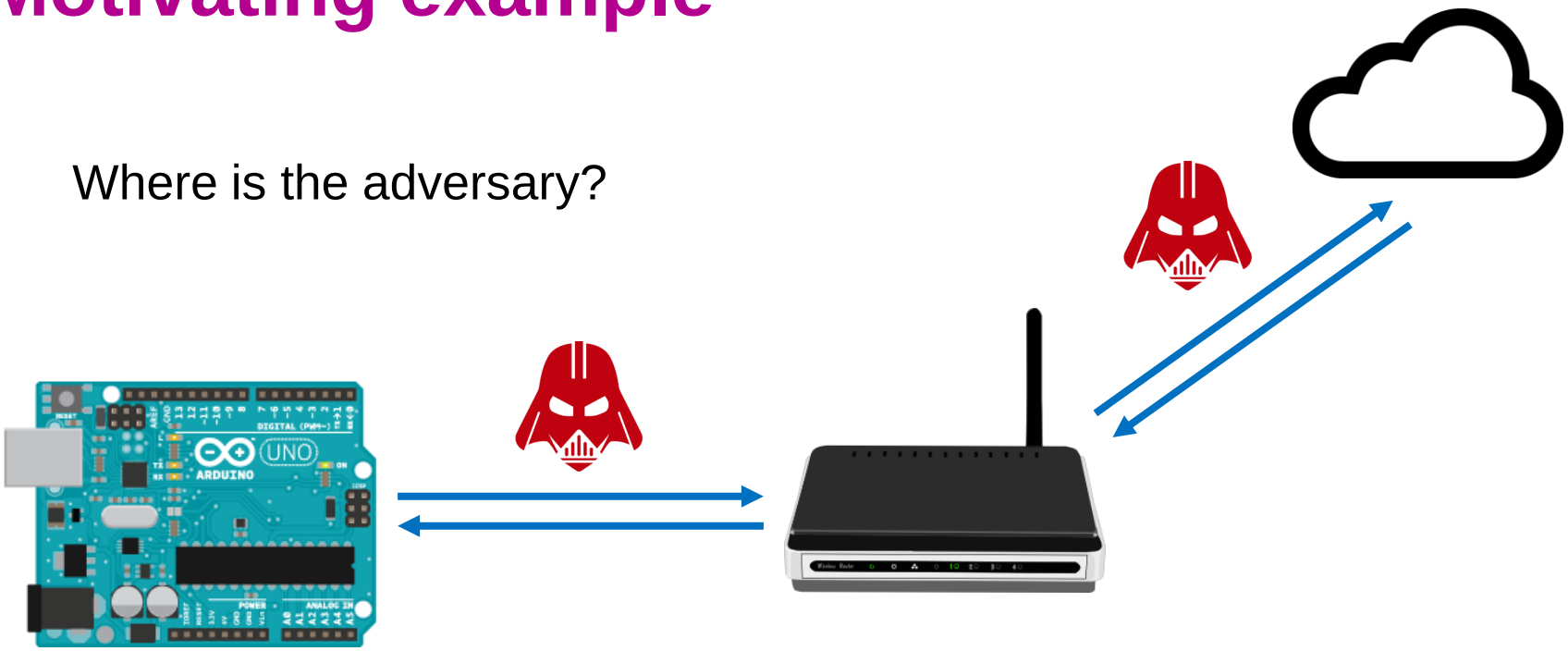
You receive the following message:

```
{
        "name": temperature,
        "value": 23.5,
        "timestamp": 1430905326.2
}
```

What does it mean?

Motivating example

Where is the adversary?



Network adversary: read, modify, falsify communication

Motivating example

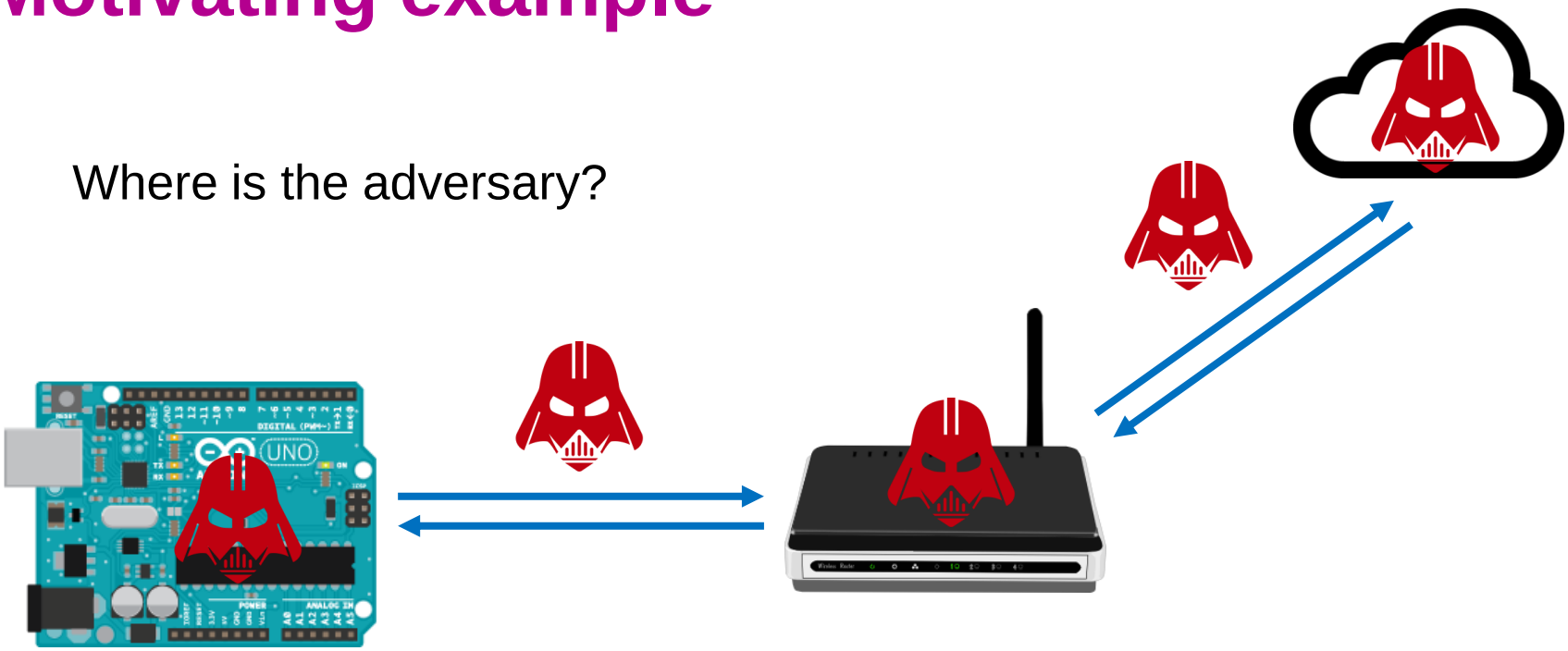
You receive the following message over an authenticated, integrity-protected communication channel:

```
{
        "name": temperature,
        "value": 23.5,
        "timestamp": 1430905326.2
}
```

What does it mean?

Motivating example

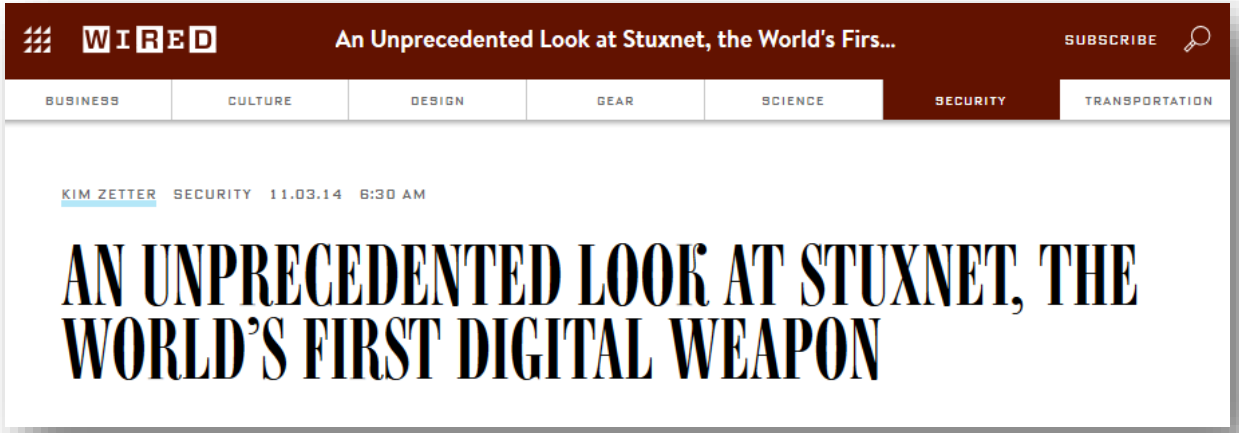
Where is the adversary?



Network adversary: read, modify, falsify communication

Malware: extract secrets, change state, modify behaviour

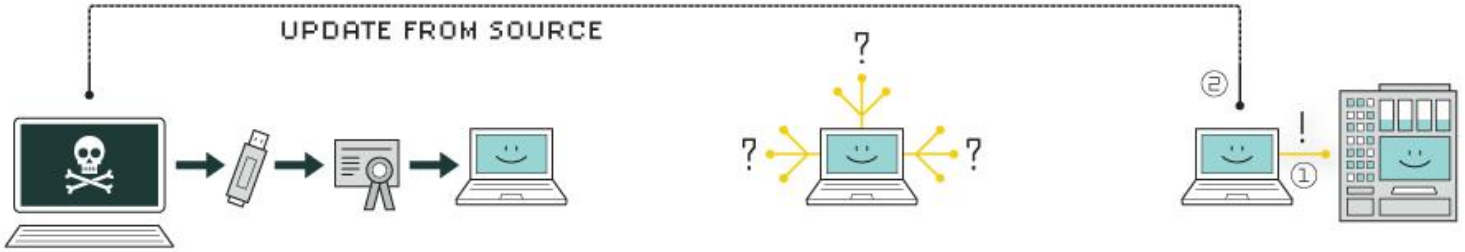
IoT malware: Stuxnet



Aalto University

UNIVERSITY OF HELSINKI

IoT malware: Stuxnet



1. infection
Stuxnet enters a system via a USB stick and proceeds to infect all machines running Microsoft Windows. By brandishing a digital certificate that seems to show that it comes from a reliable company, the worm is able to evade automated-detection systems.

2. search
Stuxnet then checks whether a given machine is part of the targeted industrial control system made by Siemens. Such systems are deployed in Iran to run high-speed centrifuges that help to enrich nuclear fuel.

3. update
If the system isn't a target, Stuxnet does nothing; if it is, the worm attempts to access the Internet and download a more recent version of itself.

4. compromise
The worm then compromises the target system's logic controllers, exploiting "zero day" vulnerabilities-software weaknesses that haven't been identified by security experts.

5. control
In the beginning, Stuxnet spies on the operations of the targeted system. Then it uses the information it has gathered to take control of the centrifuges, making them spin themselves to failure.

6. deceive and destroy
Meanwhile, it provides false feedback to outside controllers, ensuring that they won't know what's going wrong until it's too late to do anything about it.

Illustration: L-Dopa
http://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet

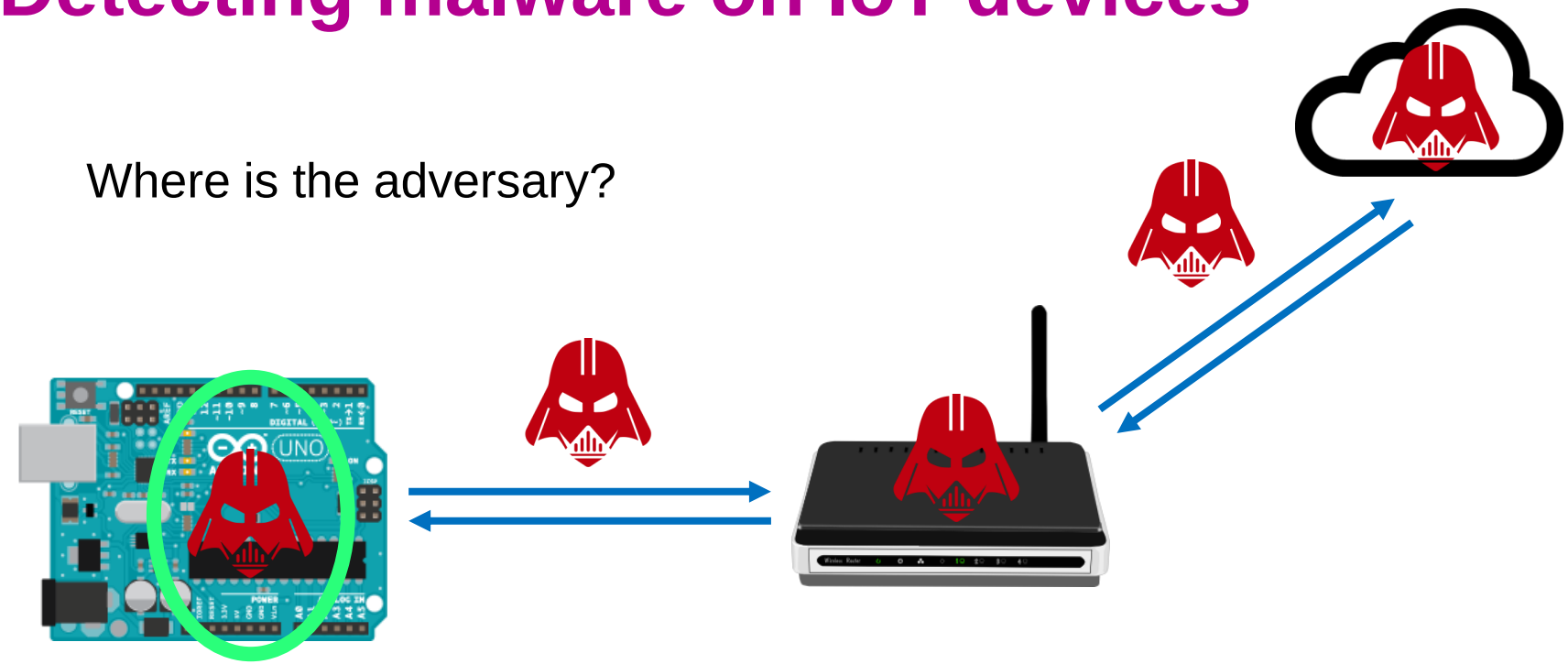
IoT malware

Malware is not a new threat, but IoT…

- Broadens the attack surface
  - cost-constrained and/or resource-constrained devices
  - many more interconnected devices
- Amplifies the impact
  - access to detailed personal information
  - control of physical environment

Detecting malware on IoT devices

Where is the adversary?

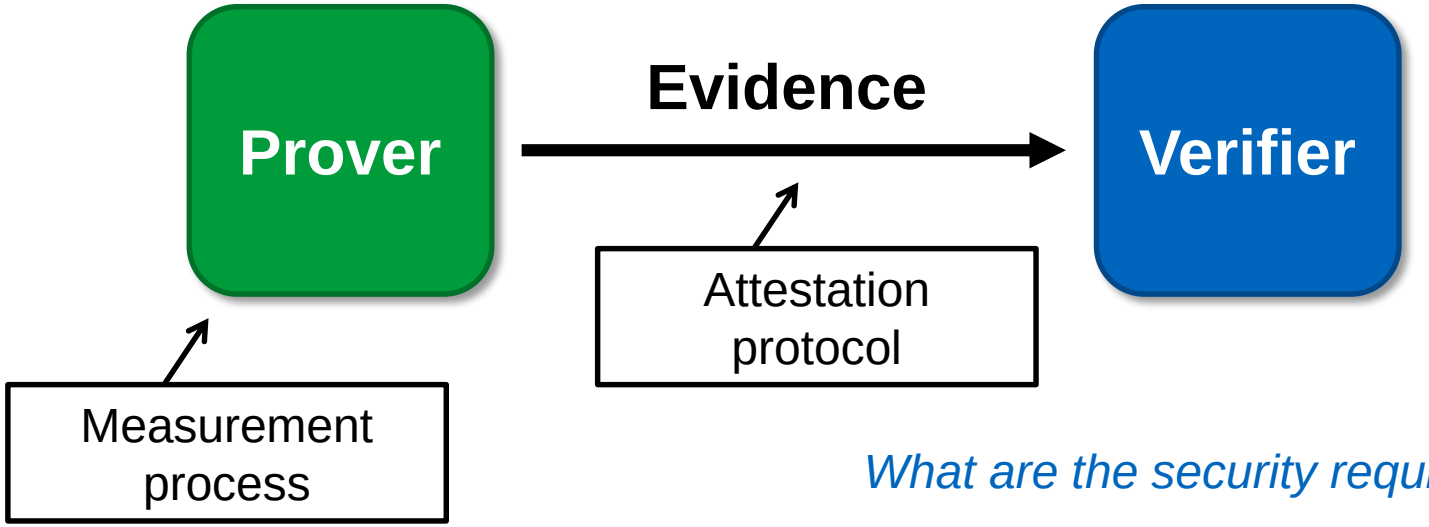


Network adversary: read, modify, falsify communication

Malware: extract secrets, change state, modify behaviour

Attestation

- An interaction between two parties through which the *verifier* ascertains the current state and/or behaviour of the *prover*.



What are the security requirements?

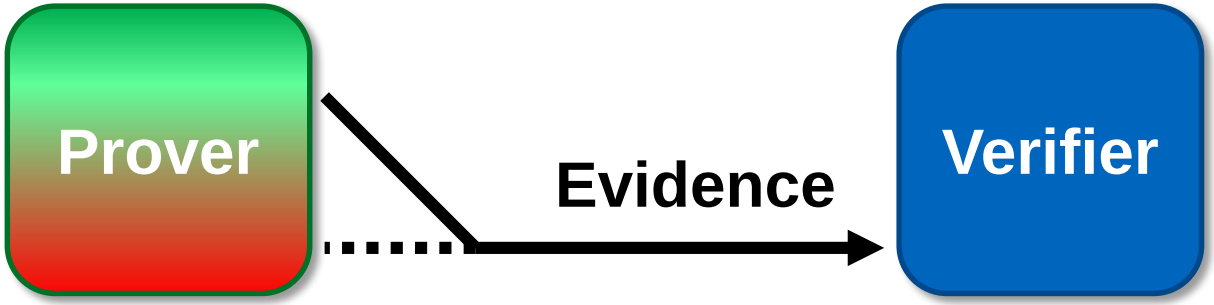
Attestation Requirements

1. Authenticity
   - representation of the *real* state of the system

Attestation Requirements

1. Authenticity
   - representation of the *real* state of the system

2. “Timeliness”
   - representation of the *current* state



TPM Attestation (Lecture 4)

- TPM Platform Configuration Registers (PCRs)
  - store cryptographic hash
  - cannot be over-written by software, only *extended*

PCR_1 = 0x00

tpm_extend(PCR_1 $hash_1$)

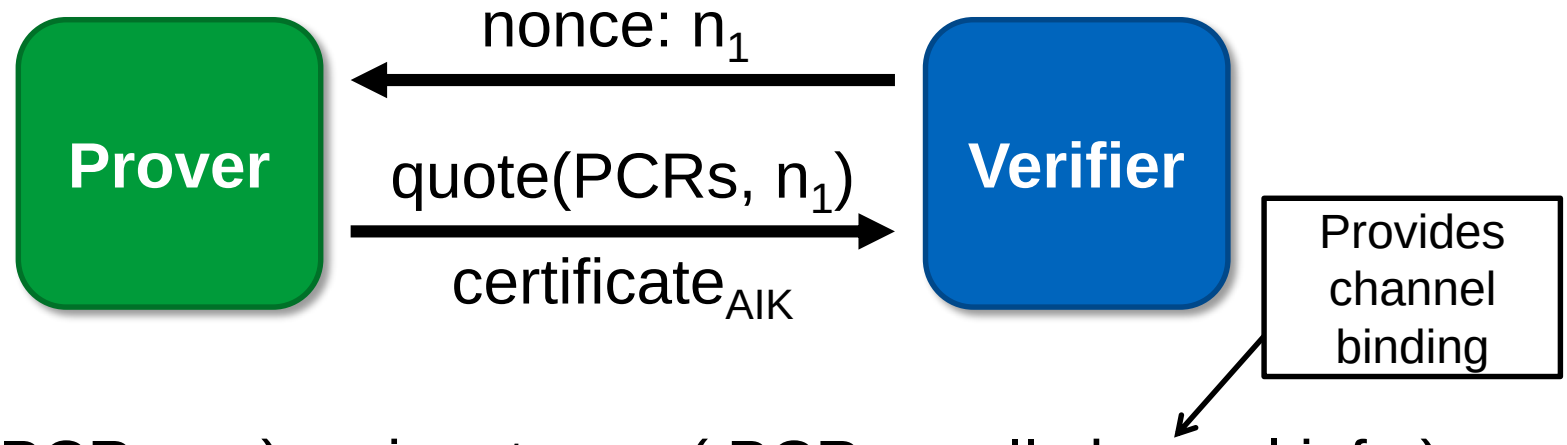
PCR_1 = hash(0x00 || $hash_1$)

tpm_extend(PCR_1 $hash_2$)

PCR_1 = hash(hash(0x00 || $hash_1$) || $hash_2$)

TPM Attestation (Lecture 4)

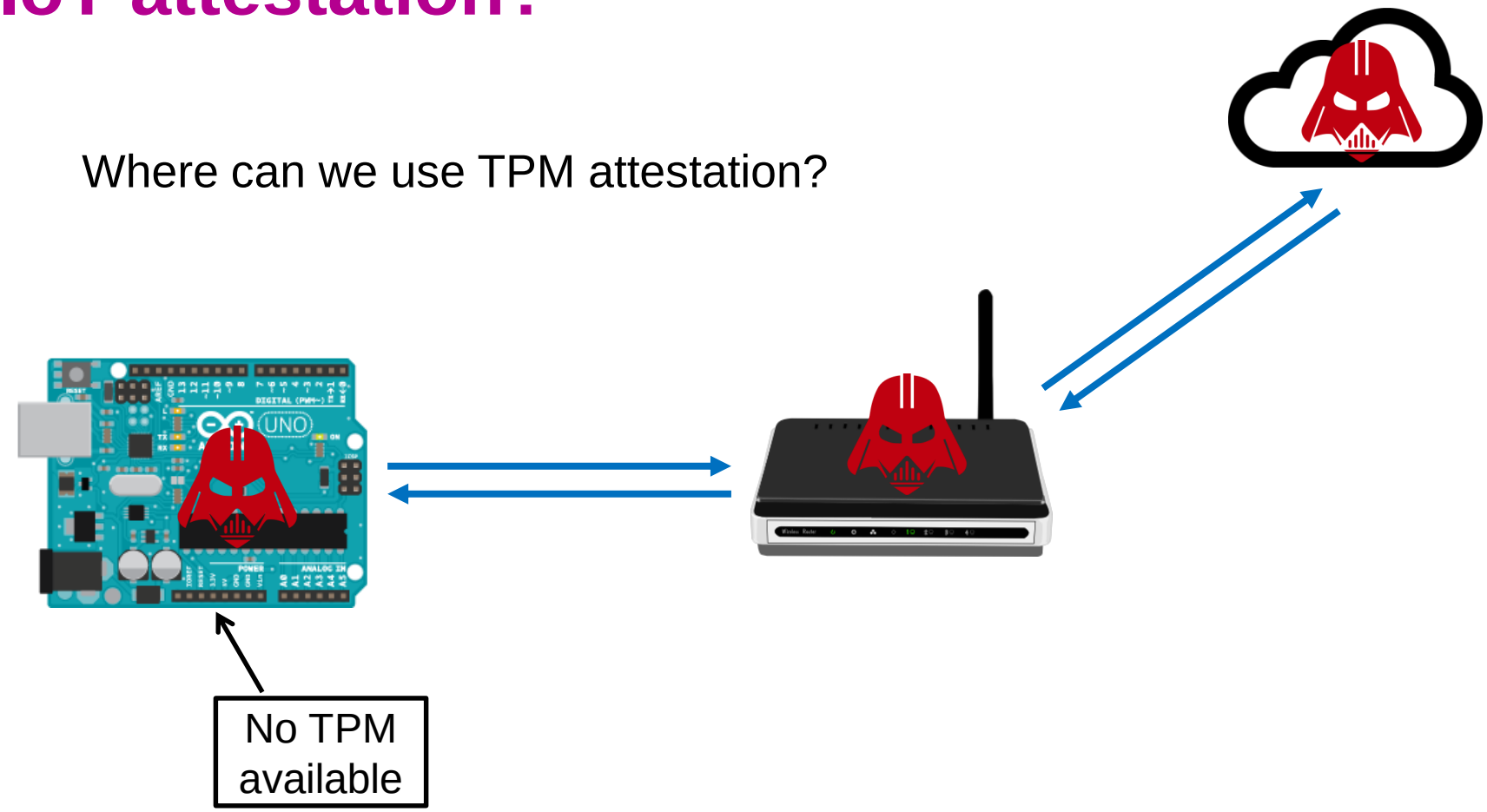
- TPM Quote
  - PCR values signed by TPM-bound Attestation Identity Key (AIK)
  - includes nonce to ensure timeliness



quote(PCRs, n_1) = signature$_{AIK}$(PCRs, n_1 || channel info)

IoT attestation?

Where can we use TPM attestation?



TPM Attestation “Costs”

- Additional hardware
  - takes up space
  - uses power
  - increases hardware cost (TPM + integration)
- Additional software
  - requires driver and software library

TPM Attestation Limitations

- Covers only the initial loading of software
- Deals with only one prover and one verifier
- *“Decision of trustworthiness”* does not scale
  - measurements change with every software update

Attestation of Things (AoT)

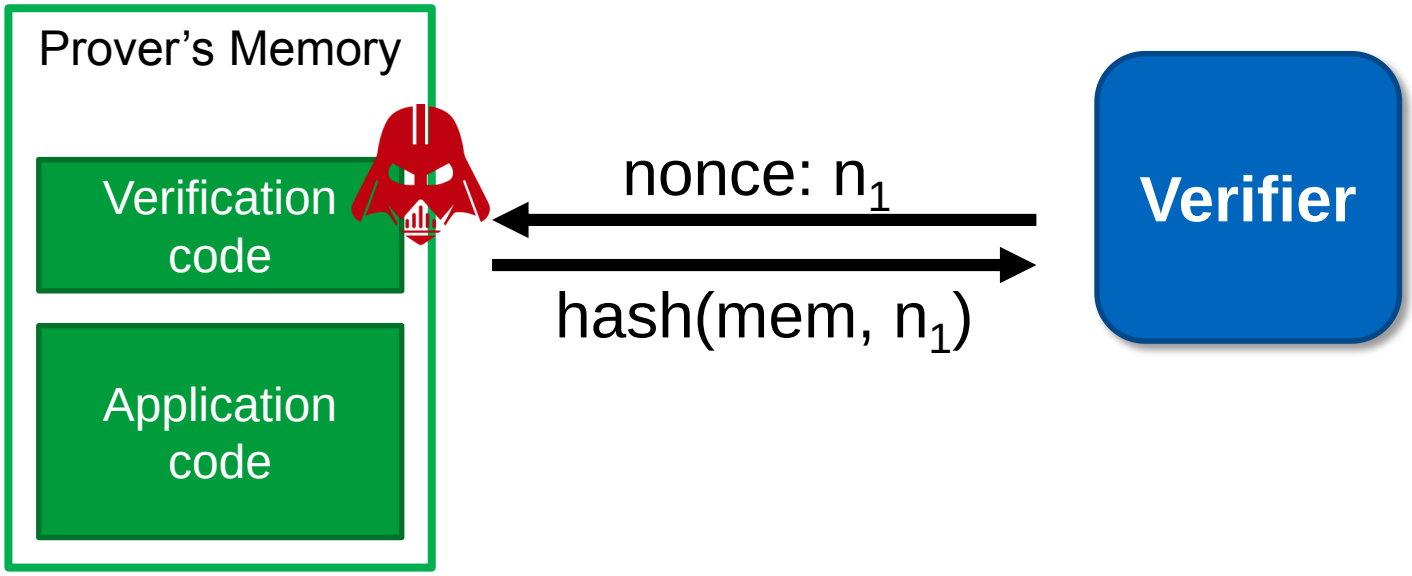
- Software-based attestation
- Hybrid attestation
- Scalability of attestation
- Run-time attestation

Attestation of Things (AoT)

- Software-based attestation
- Hybrid attestation
- Scalability of attestation
- Run-time attestation

Software-based Attestation

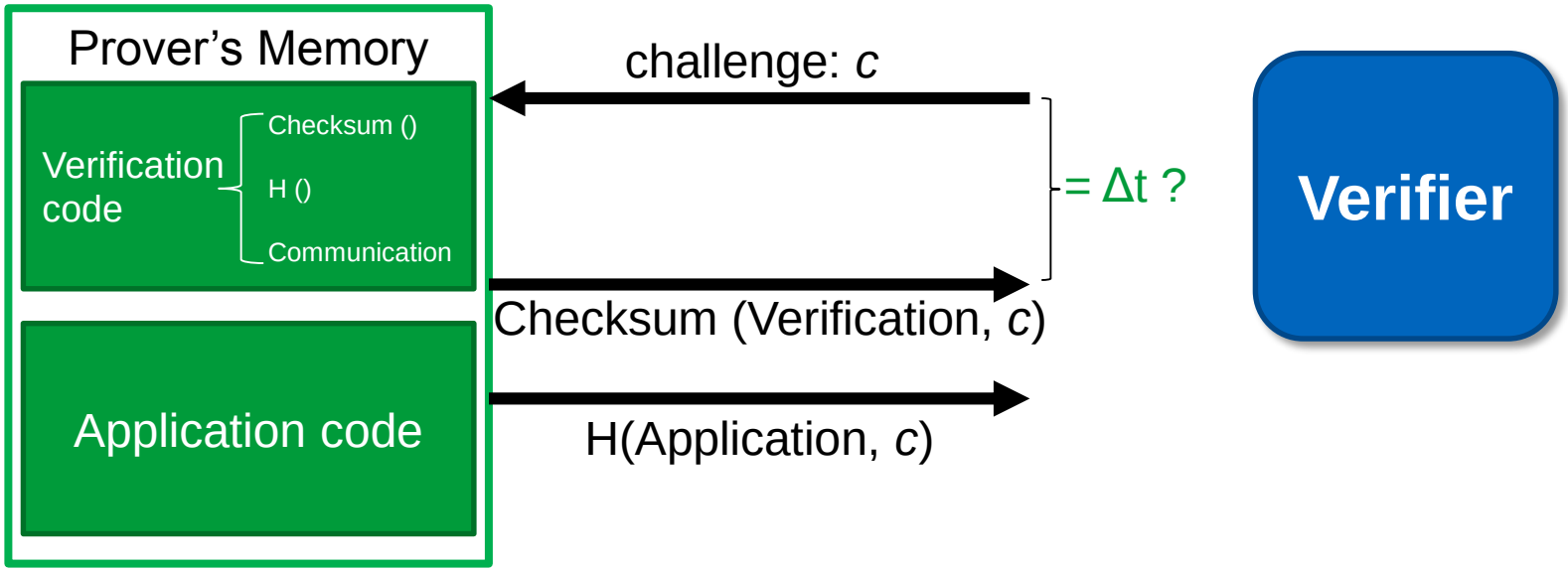
- Assumes no hardware features to support attestation
  - No secrets on prover (e.g. no AIK)
  - Cannot guarantee specific code being run



Software-based Attestation

Authenticity?

- Pioneer system
  - compute time-optimal checksum of verification code



A. Seshadri, M. Luk, E. Shi, A. Perrig, L. van Doorn, and P. Khosla. **Pioneer: Verifying integrity and guaranteeing execution of code on legacy platforms**. SOSP '05

Aalto University

UNIVERSITY OF HELSINKI

Software-based Attestation: summary

- Limitations of timing side channels
  - verifier must know exact hardware configuration
  - difficult to prove time-optimality
  - limited to “one-hop” networks
  - requires authenticated channel (e.g. physical connection)

Attestation of Things (AoT)

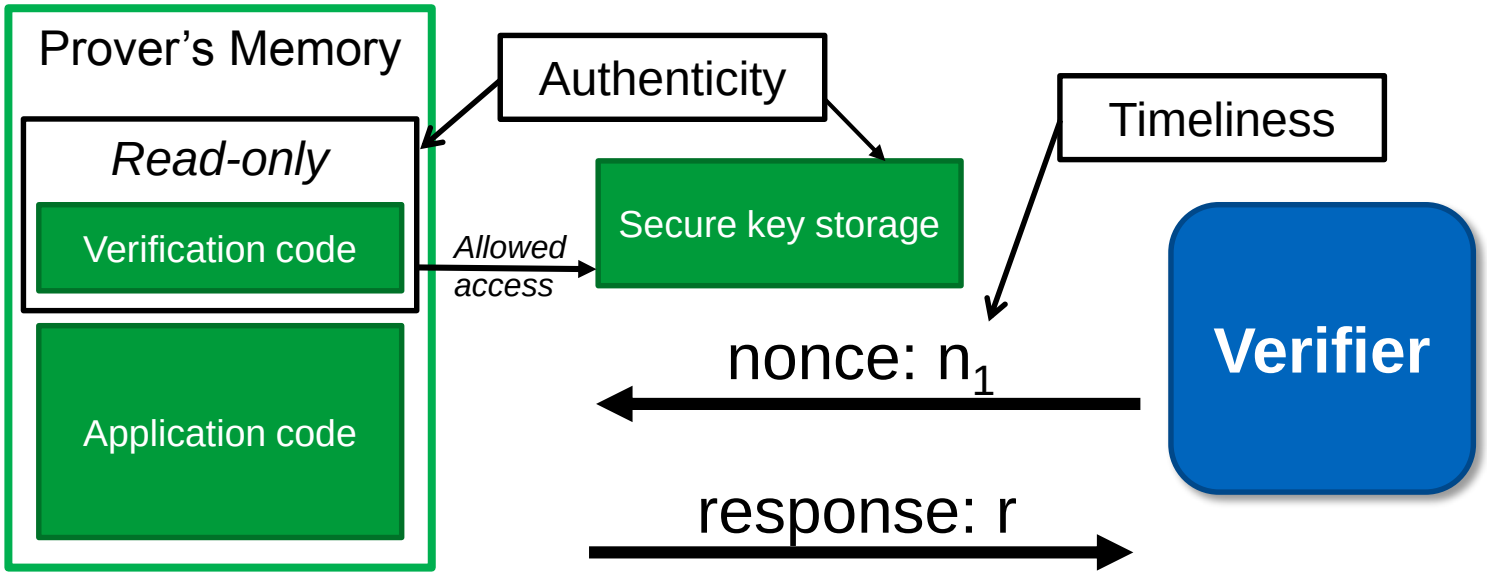
- Software-based attestation
- Hybrid attestation
- Scalability of attestation
- Run-time attestation

Hybrid Attestation

- Minimal trust anchors
  - small changes to hardware
  - “hardware/software co-design”

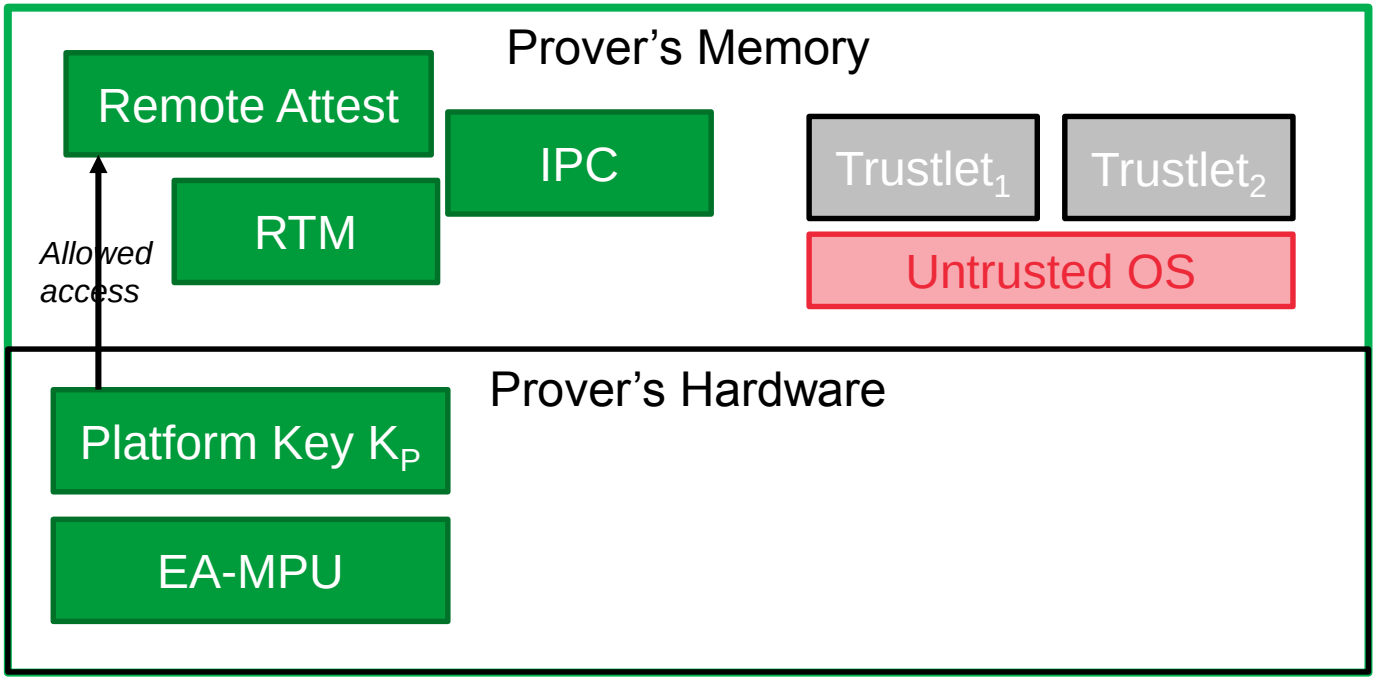
Hybrid Attestation: SMART

Read-only Verification code, secure key storage and atomicity of execution of Verification code



Hybrid Attestation: TrustLite & TyTAN

- Execution-Aware Memory Protection Unit (EA-MPU)
  - access control based on memory request target and origin



- P. Koeberl, et al. **TrustLite: A Security Architecture for Tiny Embedded Devices**. EuroSys '14
- F. Brasser, et al, **TyTAN: Tiny Trust Anchor for Tiny Devices,** DAC '15

Hybrid Attestation: summary

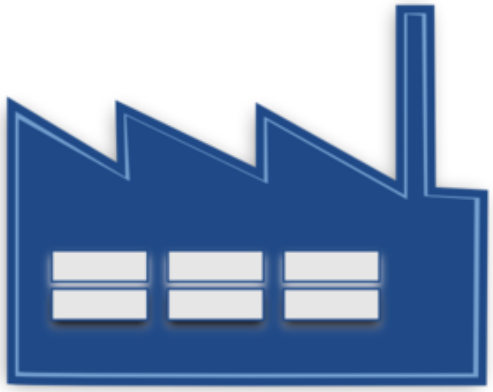
- Advantages of hybrid approaches
  - can be used across a network / over an untrusted channel
  - Verifier need not know prover's exact hardware configuration
- Drawbacks
  - Needs additional hardware support
  - But minimal MCU trust anchors soon available commercially
    - TrustZone-M (ARM v8), …

Attestation of Things (AoT)

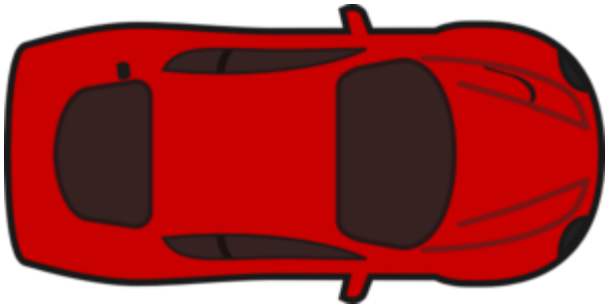
- Software-based attestation
- Hybrid attestation
- Scalability of attestation
- Run-time attestation

Scalability of Attestation

- Attestation protocols usually assume a single prover
  - but IoT scenarios may involve groups of (many) provers



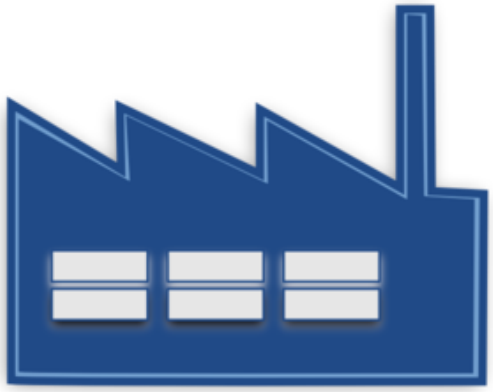
Smart factories



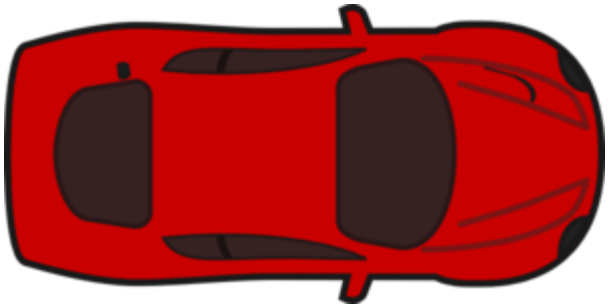
Smart vehicles

Scalability of Attestation

- Device *swarms*
  - dynamic topology: nodes move within swarm
  - dynamic membership: nodes join and leave the swarm



Smart factories

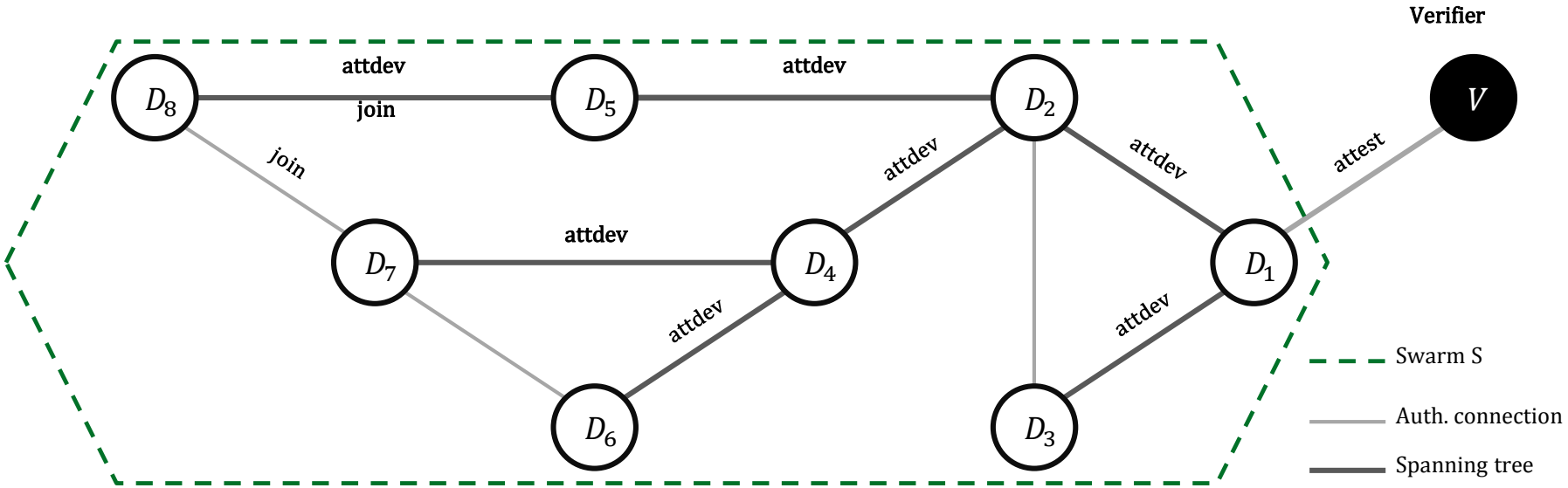


Smart vehicles

Scalability of Attestation

- SEDA: Scalable Embedded Device Attestation
  - more efficient than attesting each node individually
  - can use any type of measurement process
  - assumes homogeneous provers

Scalability of Attestation



Aalto University

N. Asokan, F. Brasser, A. Ibrahim, A.-R. Sadeghi, M. Schunter, G. Tsudik, and C. Wachsmann. **SEDA: Scalable Embedded Device Attestation**. CCS '15

UNIVERSITY OF HELSINKI

Scalability of Attestation: summary

How to extend SEDA to

- support highly dynamic swarms?
- be resilient to physical compromise of some devices?

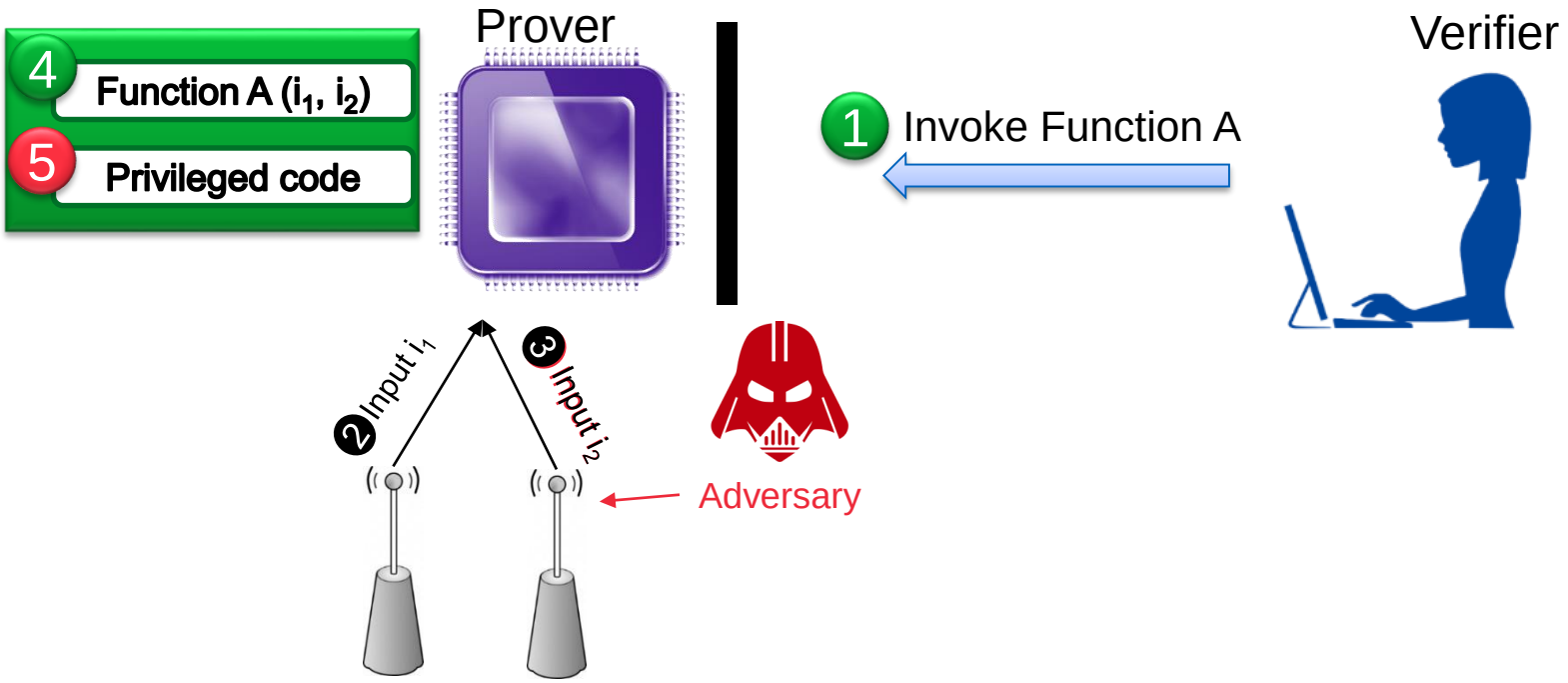
Attestation of Things (AoT)

- Software-based attestation
- Hybrid attestation
- Scalability of attestation
- Run-time attestation

Why run-time attestation?

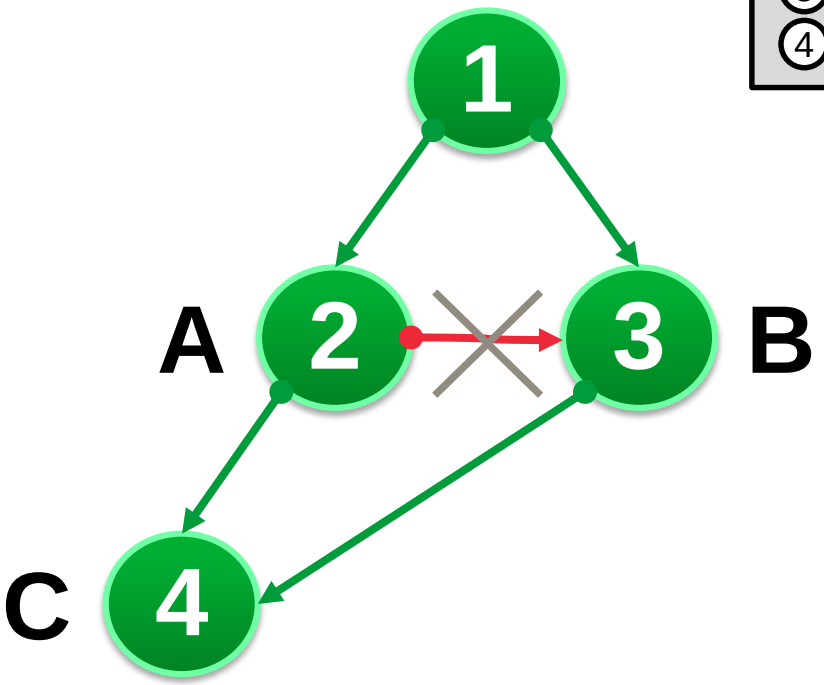
- Traditional attestation measures binaries at load time
- Cannot capture run-time attacks
  - return-oriented programming
  - control data attacks

Run-time attacks



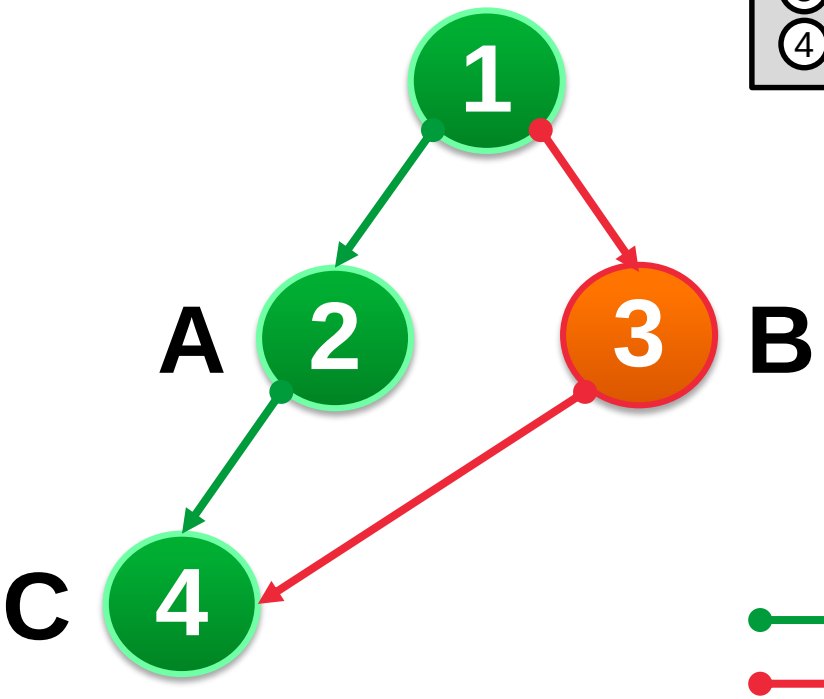
Control flow integrity (CFI)

```
① if (condition)
②  then: block A
③  else: block B
④ block C
```



Run-time attacks without violating CFI

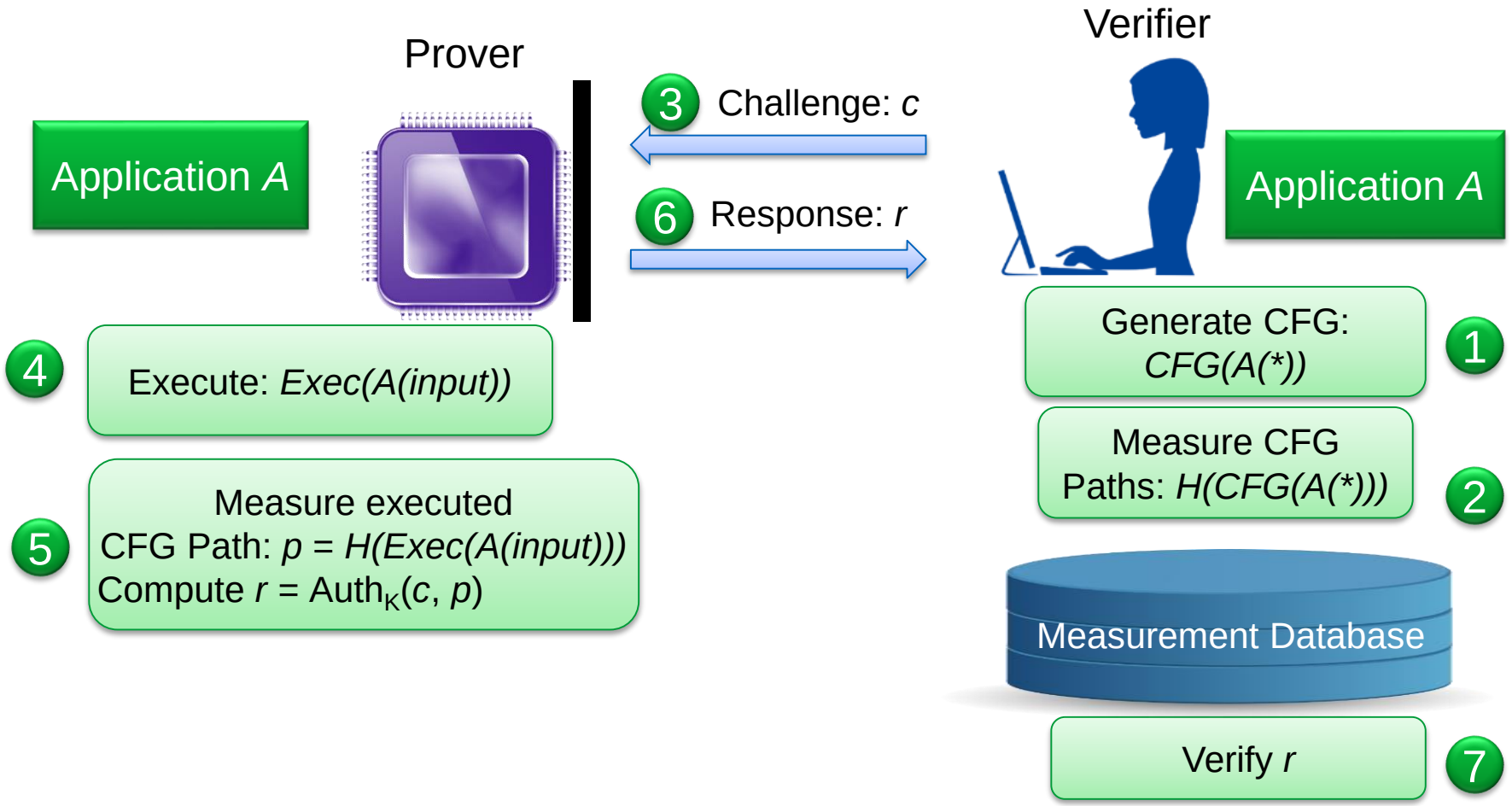
```
① if (condition)
②  then: block A
③  else: block B
④ block C
```



Normal operation

Privileged operation

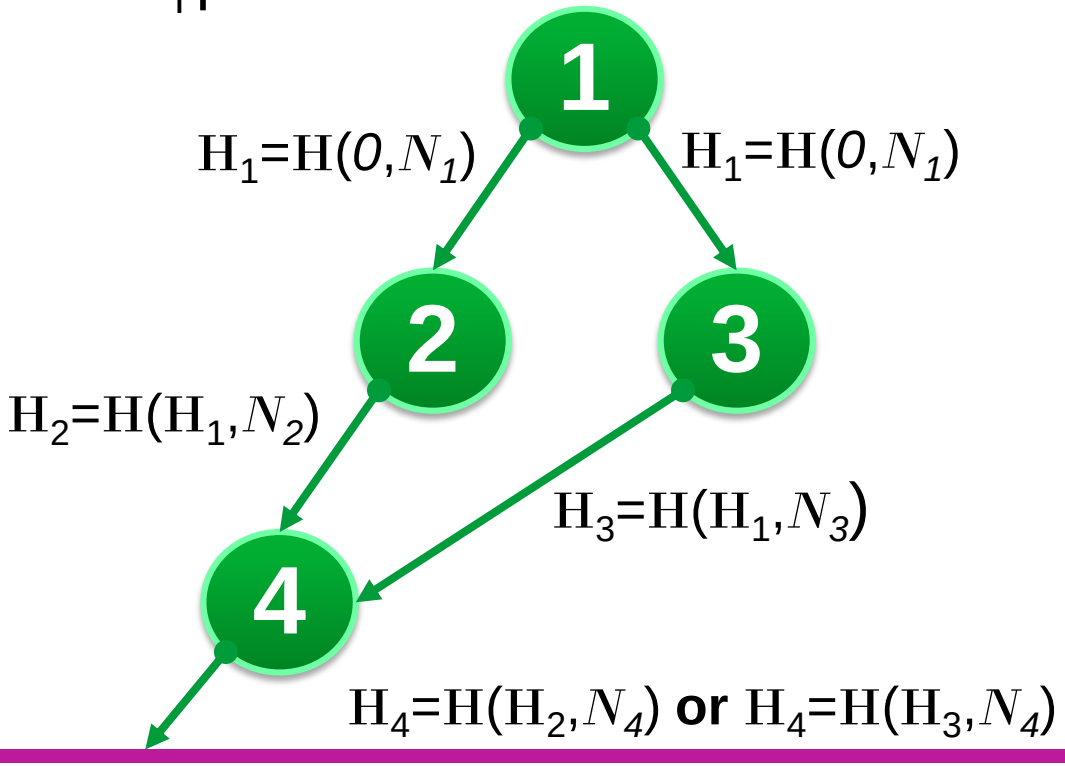
Control-Flow Attestation (C-FLAT)



T. Abera, N. Asokan, L. Davi, J-E. Ekberg, A. Paverd, T. Nyman, A-R. Sadeghi, G. Tsudik, **C-FLAT: Control Flow Attestation for Embedded Systems Software**, CCS '16. http://arxiv.org/abs/1605.07763,

C-FLAT: High-Level Idea

- Cumulative Hash Value: $H_j=H(H_i,N)$, where H_i previous hash result and N is the current node

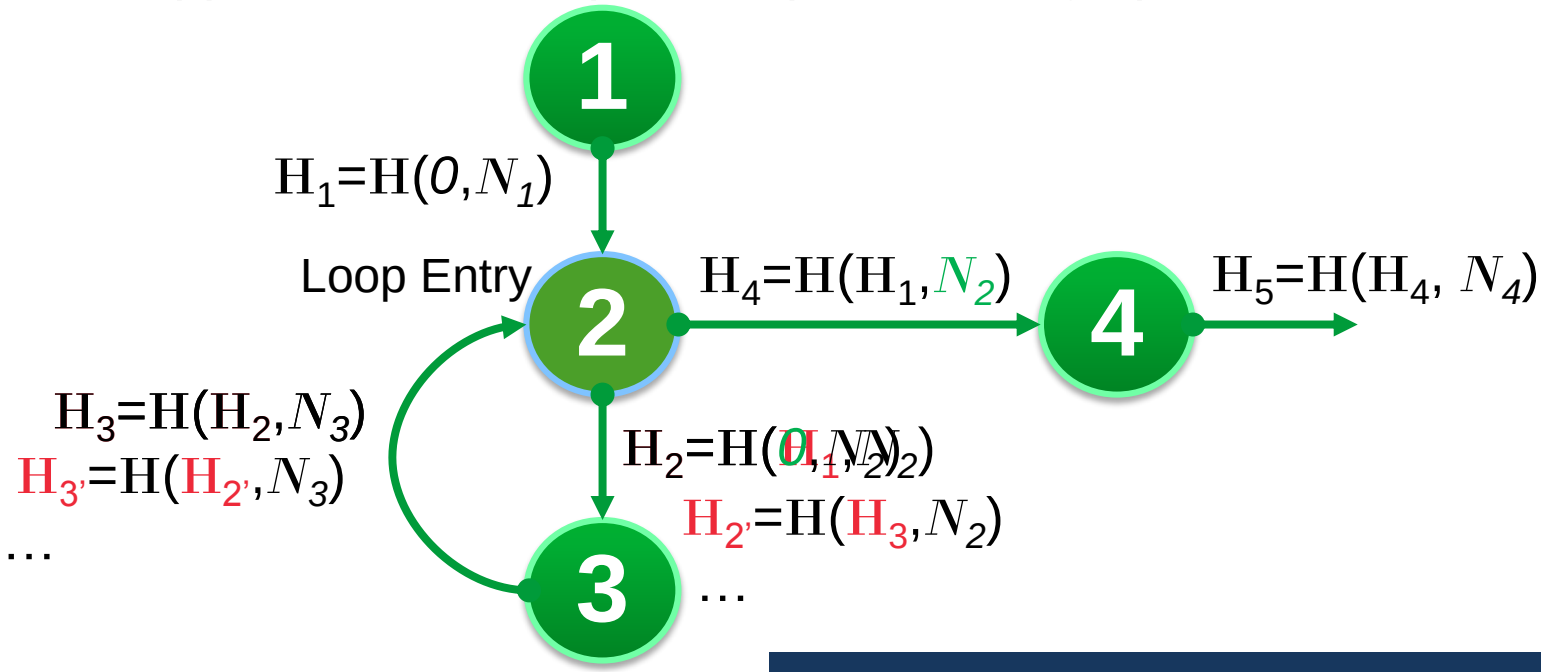


$p = H_4$

Handling Loops

H_x different for each loop iteration

- Different loop paths/iterations → many valid hash values
  - Our approach: treat loops as separate sub-graphs



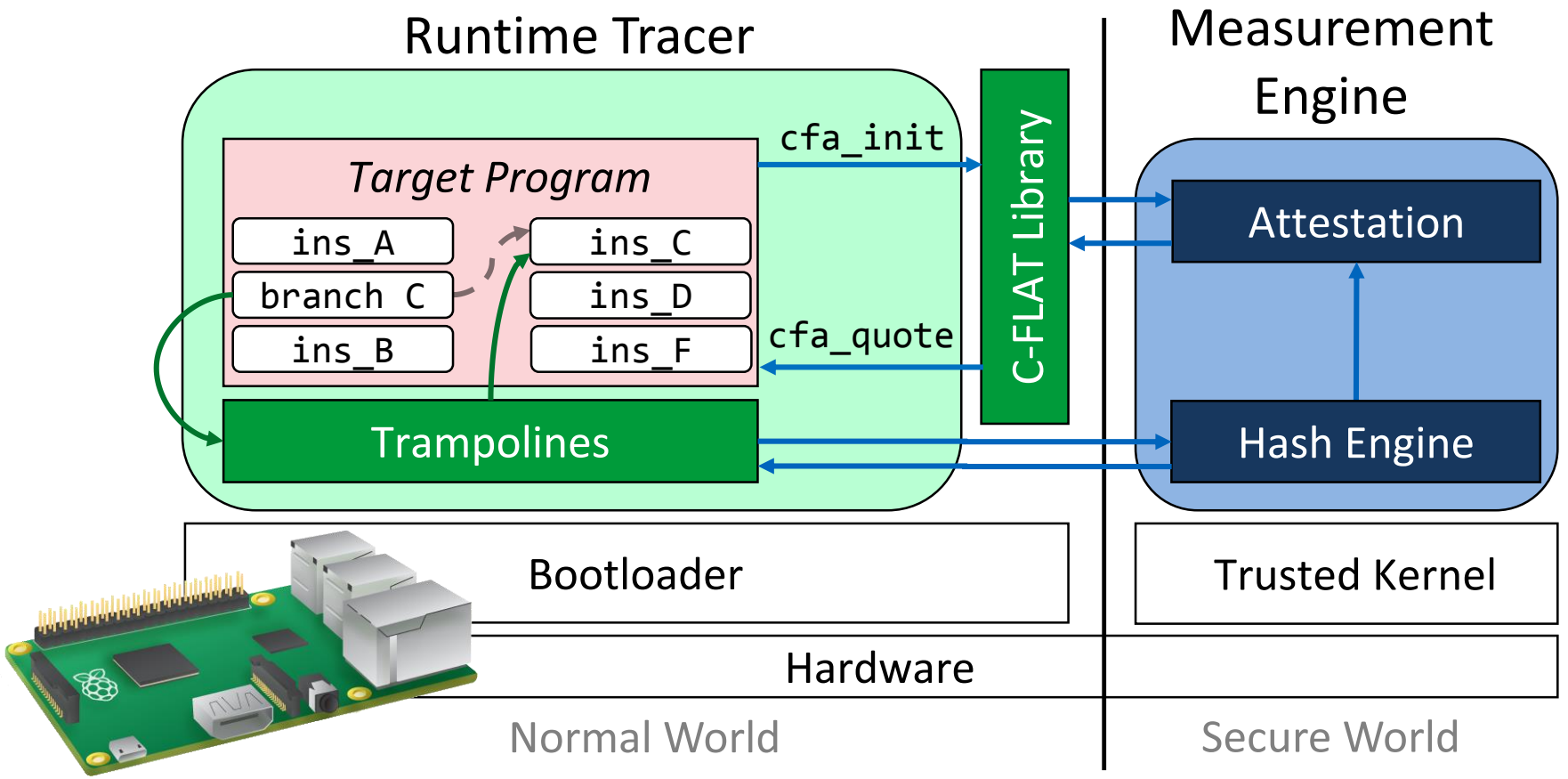
Proof-of-Concept Implementation

- Bare-metal prototype on Raspberry Pi 2
  - Single-purpose program instrumented using binary-rewriting
  - Runtime Monitor written in ARM assembler
  - Measurement Engine isolated in TrustZone-A Secure World

cfa_quote: 7c 16 d6 51 20 a2 a0 c7 90 f5 ef 04 0c 2e ba bc
loop[000]: 78 22 5b 62 92 41 ca 02 7b ff 29 57 c6 6f 9b a2
path[000]: 2f a5 8c dc 1b 35 41 29 ab dd 35 5c f2 69 08 37 (1)
loop[001]: d6 90 9e a0 8c ae 90 84 9e 66 09 f8 a6 7b 52 04
path[000]: 92 fb d1 e8 90 cb 02 e5 6c f2 65 8c 86 72 0e d3 (2)
….
loop[006]: 05 e3 92 40 95 ef 7b 46 13 7d 6e 8b 05 be bf 41
path[000]: 67 c6 5e d4 18 13 02 bc 4a 5d 60 a0 16 85 f4 ed (9)
path[001]: 78 19 af 09 0f d5 64 f4 39 b4 7a 0d 97 57 77 8c (2)

Source: https://github.com/control-flow-attestation/c-flat/blob/master/samples/syringe/syringe-auth.txt

Proof-of-Concept Implementation



T. Abera, N. Asokan, L. Davi, J-E. Ekberg, A. Paverd, T. Nyman, A-R. Sadeghi, G. Tsudik, **C-FLAT: Control Flow Attestation for Embedded Systems Software**, CCS '16. http://arxiv.org/abs/1605.07763,

Aalto University

UNIVERSITY OF HELSINKI

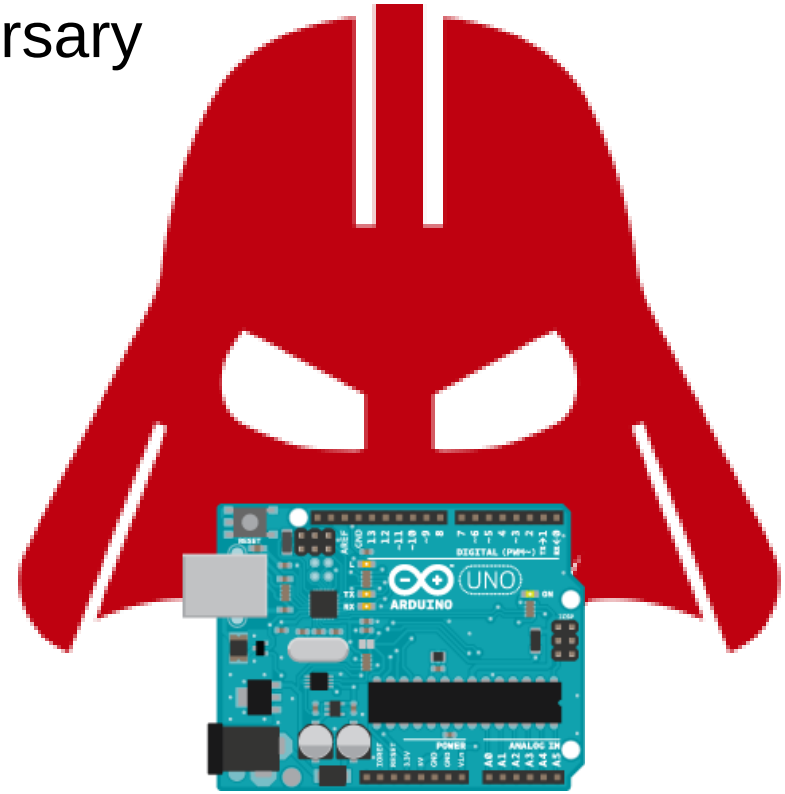
Run-time attestation: summary

- How can we scale control flow attestation?
  - Better ways to aggregate measurements?
  - Faster/simpler purpose-built hash functions?
  - Purpose-built hardware support?

- What next?
  - Attestation of properties rather than measurements?
    - from attestation to checking compliance with a (dynamic) policy?

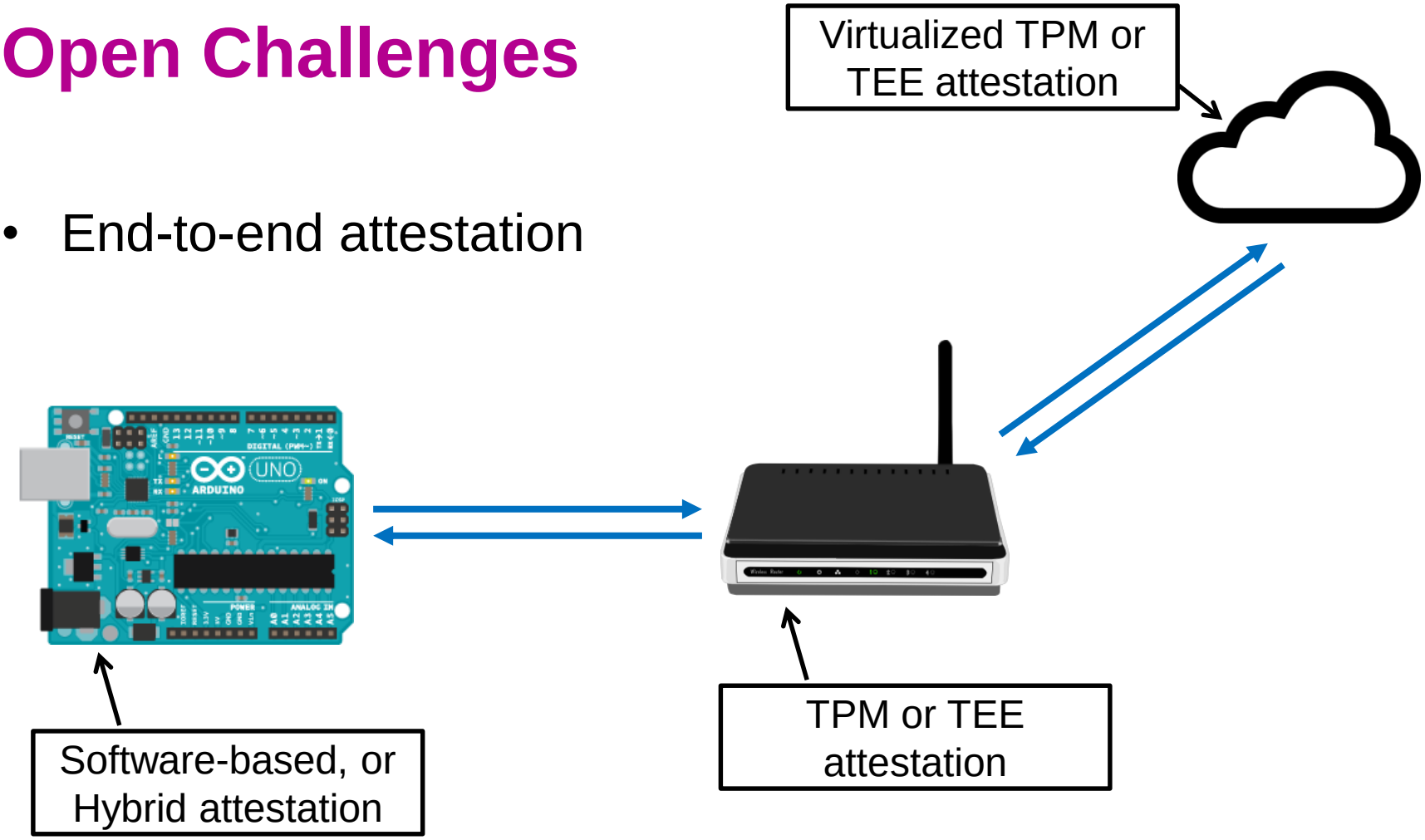
Open Challenges

- Physical adversary



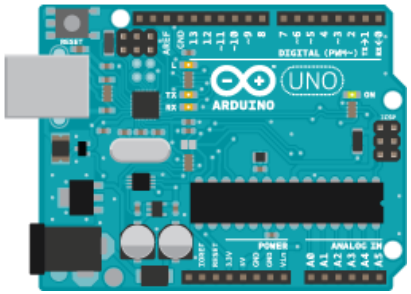
Open Challenges

- End-to-end attestation



Open Challenges

- Device heterogeneity



Did you learn about:

- Why we need attestation in IoT
- Some recent research towards achieving this

Attestation in the Internet of Things (IoT)

Mobile Systems Security 2017

Optional lecture

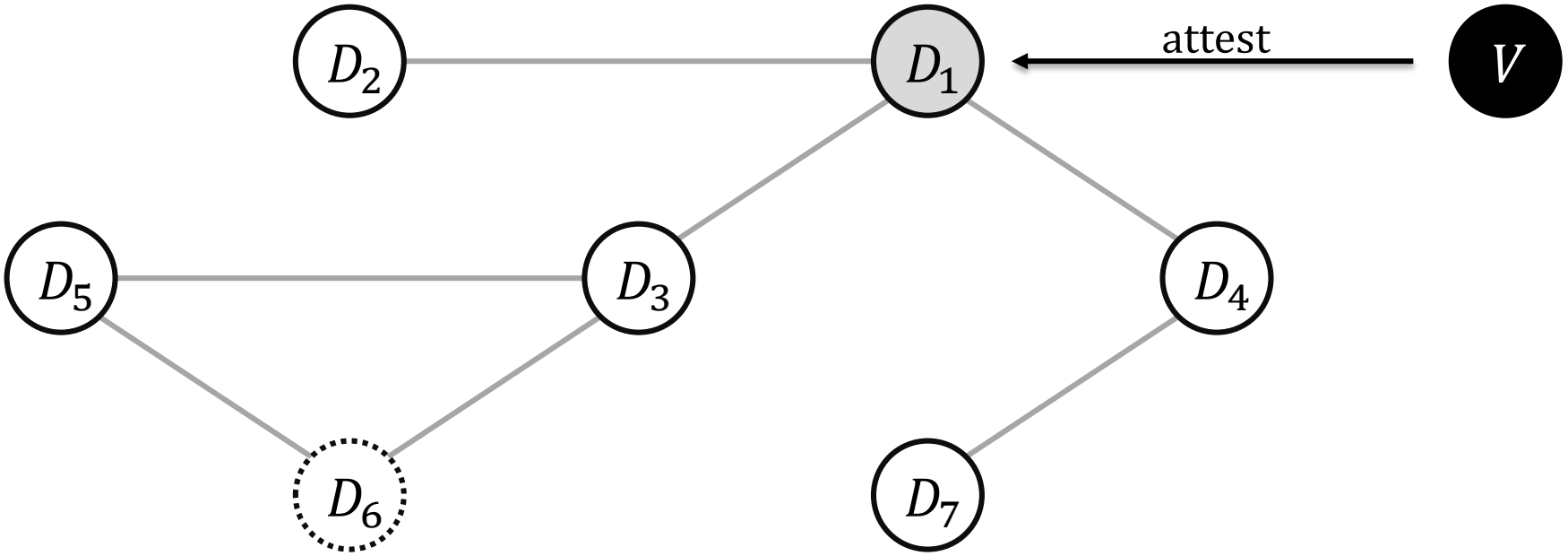
February 16th, 2017

Andrew Paverd
Aalto University
andrew.paverd@aalto.fi
*(***Contributors: N. Asokan, Lucas Davi)**

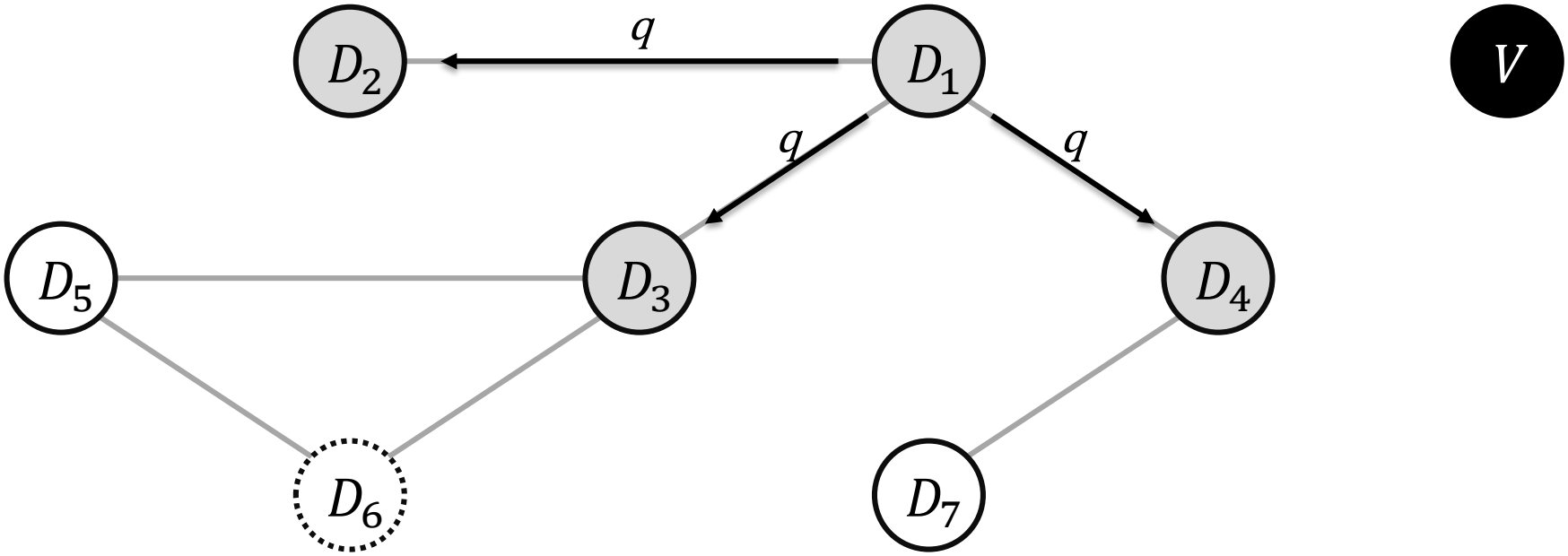
Aalto University

UNIVERSITY OF HELSINKI

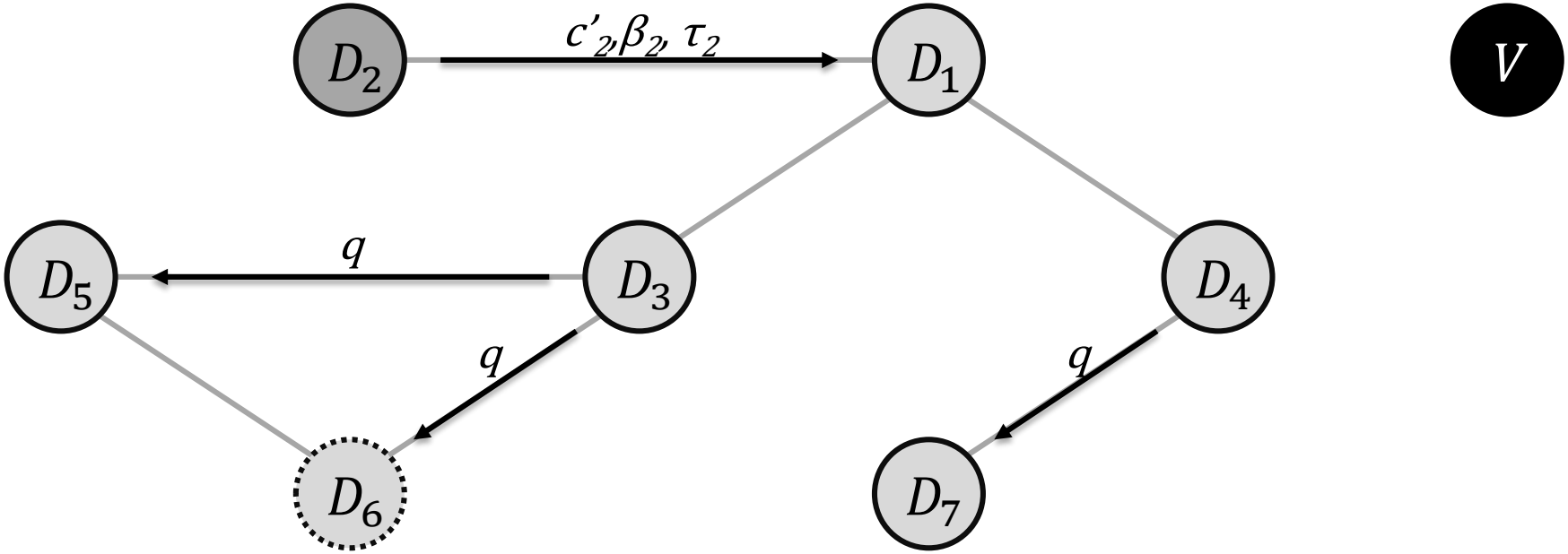
SEDA (in detail)



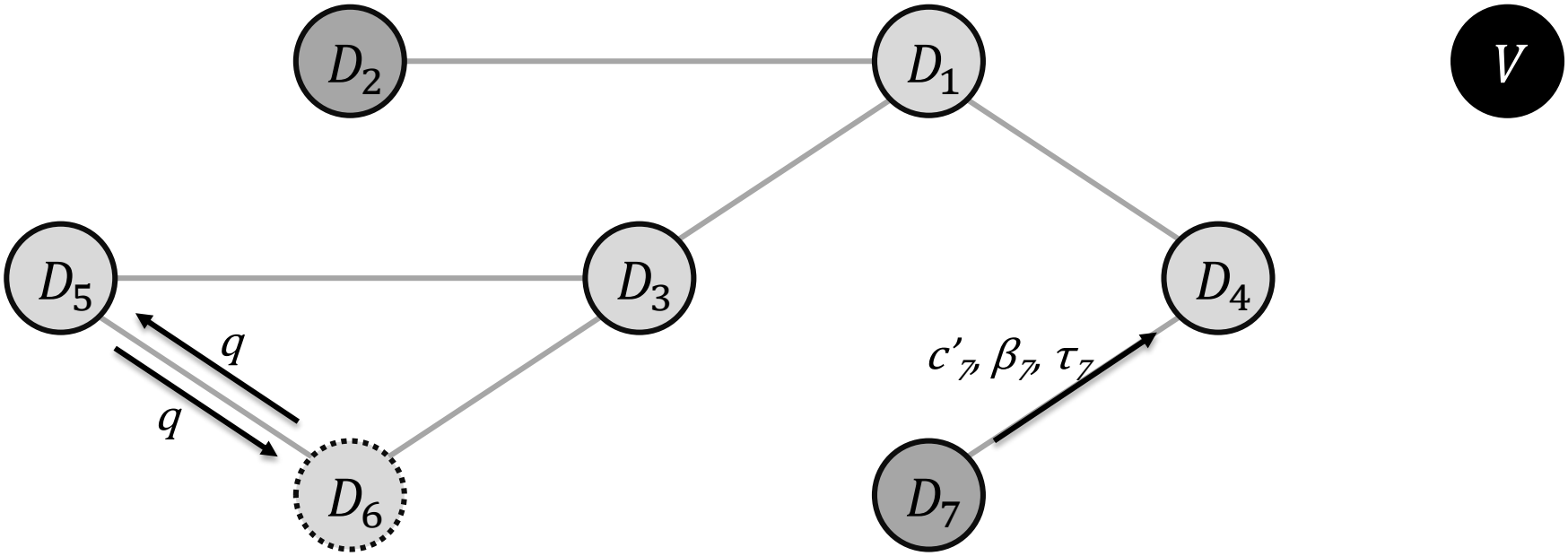
SEDA (in detail)



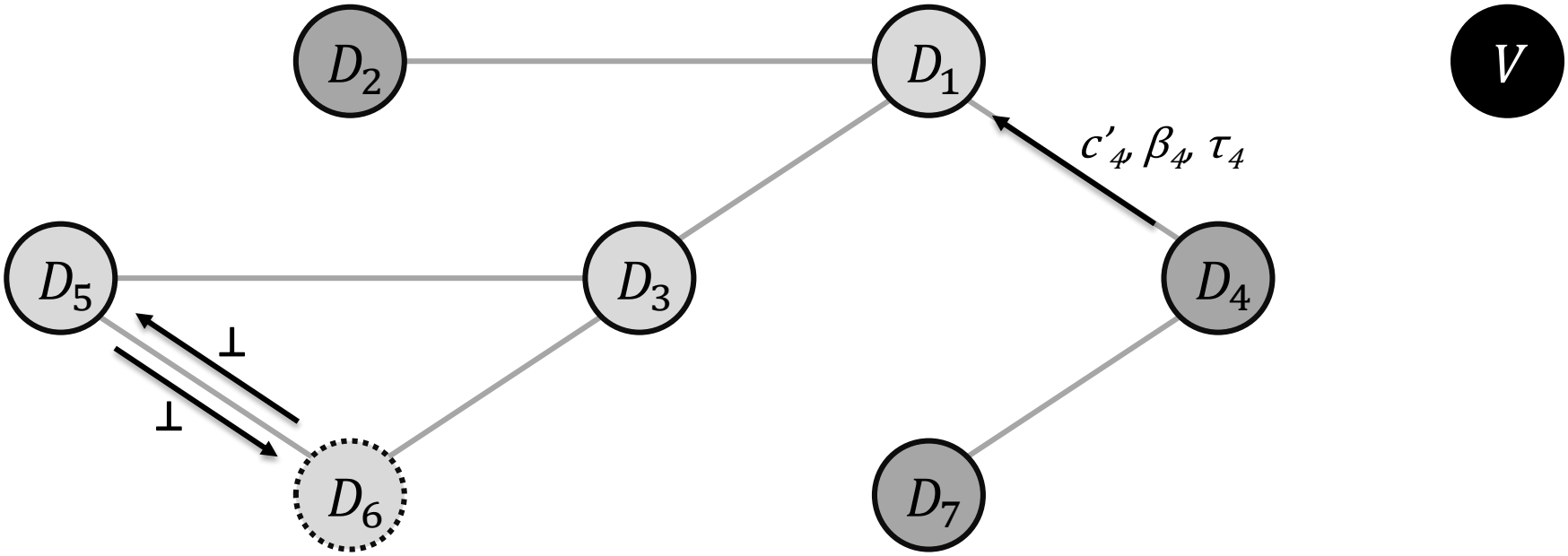
SEDA (in detail)



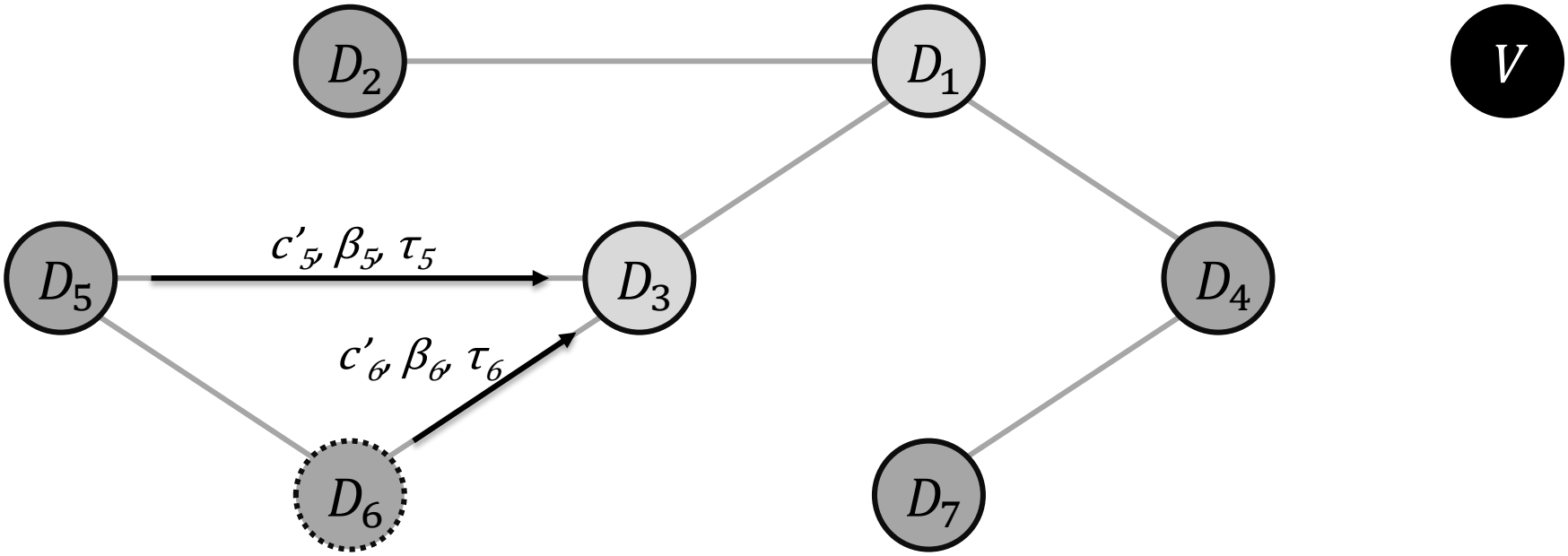
SEDA (in detail)



SEDA (in detail)



SEDA (in detail)

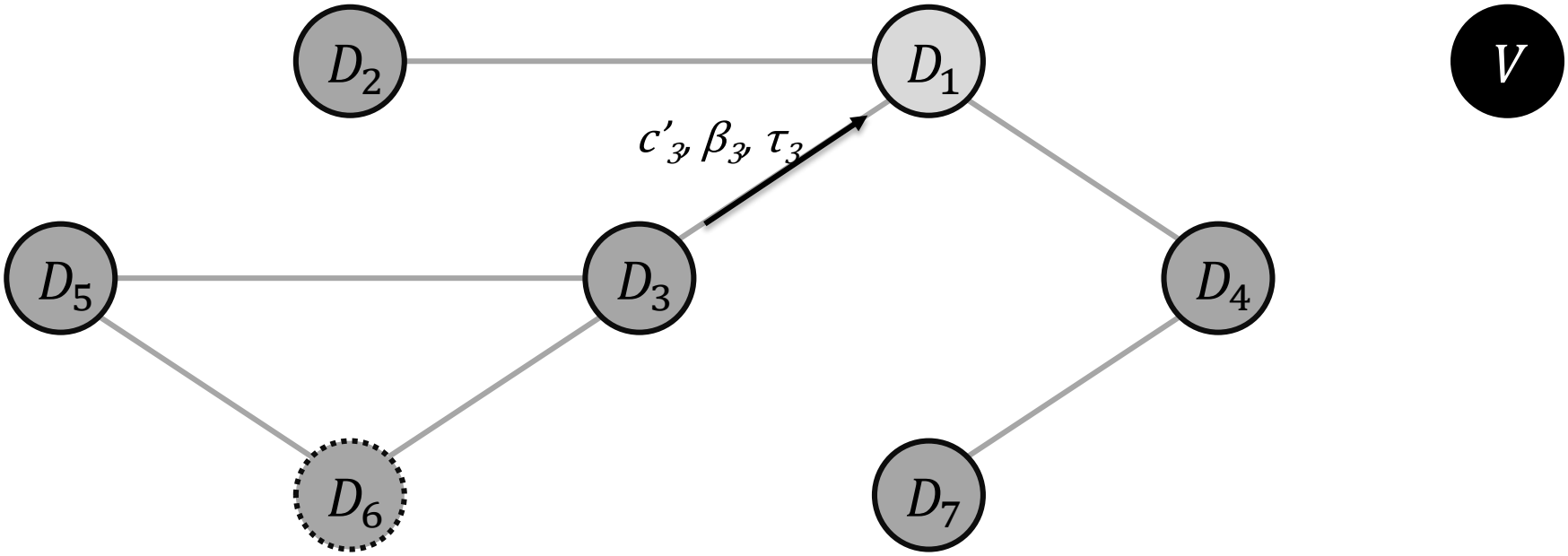


N. Asokan, F. Brasser, A. Ibrahim, A.-R. Sadeghi, M. Schunter, G. Tsudik, and C. Wachsmann. **SEDA: Scalable Embedded Device Attestation**. CCS '15

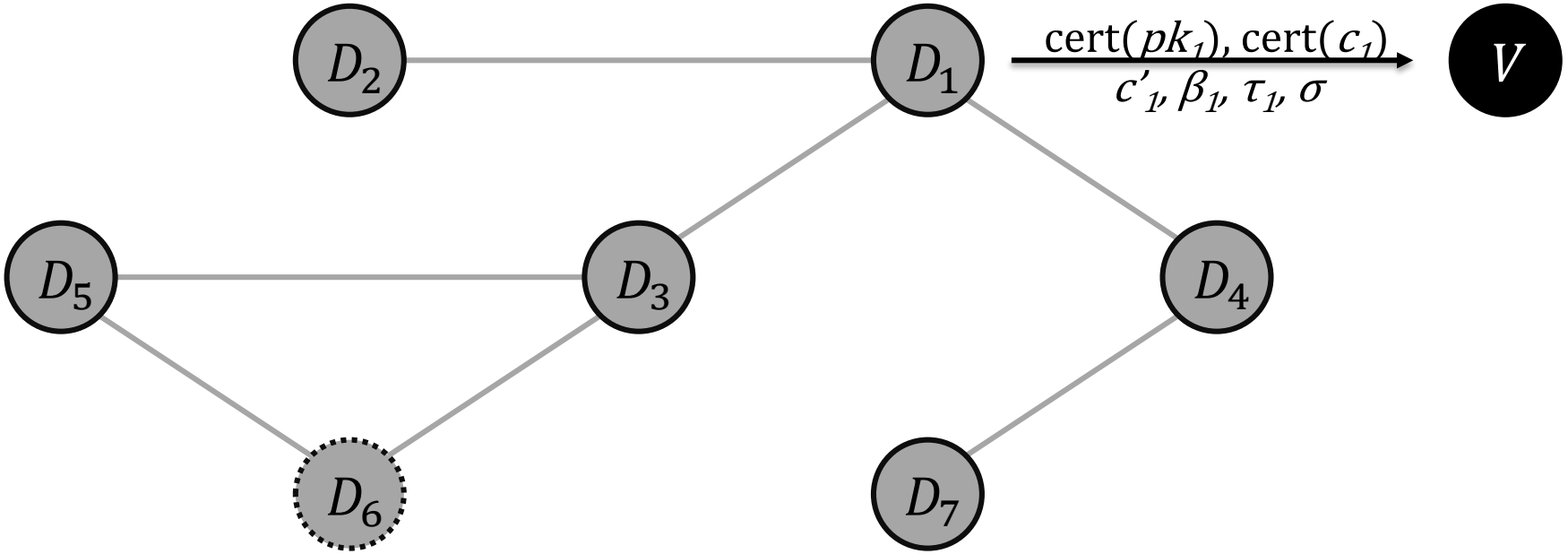
Aalto University

UNIVERSITY OF HELSINKI

SEDA (in detail)



SEDA (in detail)



Abuse of Attestation

- Attestation protocols usually consider a benign verifier
- But adversary could impersonate verifier to abuse attestation
- E.g., computing a MAC over all memory in a typical microcontroller could take ~ 750 ms
  - could be used for denial of service (DoS) attack

Abuse of Attestation

- Verifier must be authenticated to prover
  - but asymmetric crypto is computationally expensive
- Prover must detect replays of previous requests
  - can use nonces
  - can use counters
  - can use timestamps

Abuse of Attestation

- Nonces
  - require integrity-protected storage for previous nonces
- Counters
  - require minimal integrity-protected storage for counter
- Timestamps
  - require trusted synchronized clock at prover side

Use Execution-Aware Memory Access Control (EA-MAC) to protect counters and timers.