Things, Trouble, Trust: On Building Trust in IoT Systems

Tigist Abera, N. Asokan, Lucas Davi, Farinaz Koushanfar, Andrew Paverd, Ahmad-Reza Sadeghi, Gene Tsudik

Aalto University

Outline

- What is **attestation** and why do we need it in IoT?
- Why is traditional attestation not a good fit for IoT?
- How is current research addressing these gaps?

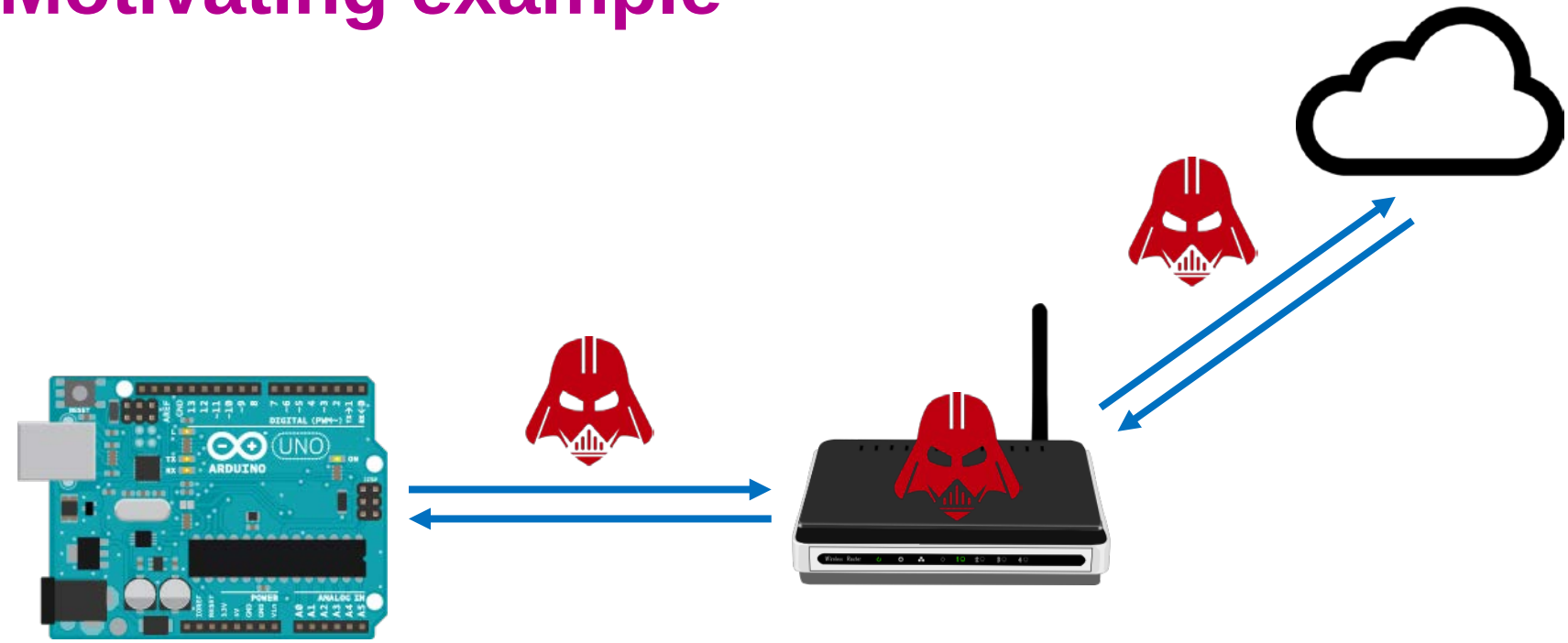
Motivating example

The following message is received (from an IoT device):

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

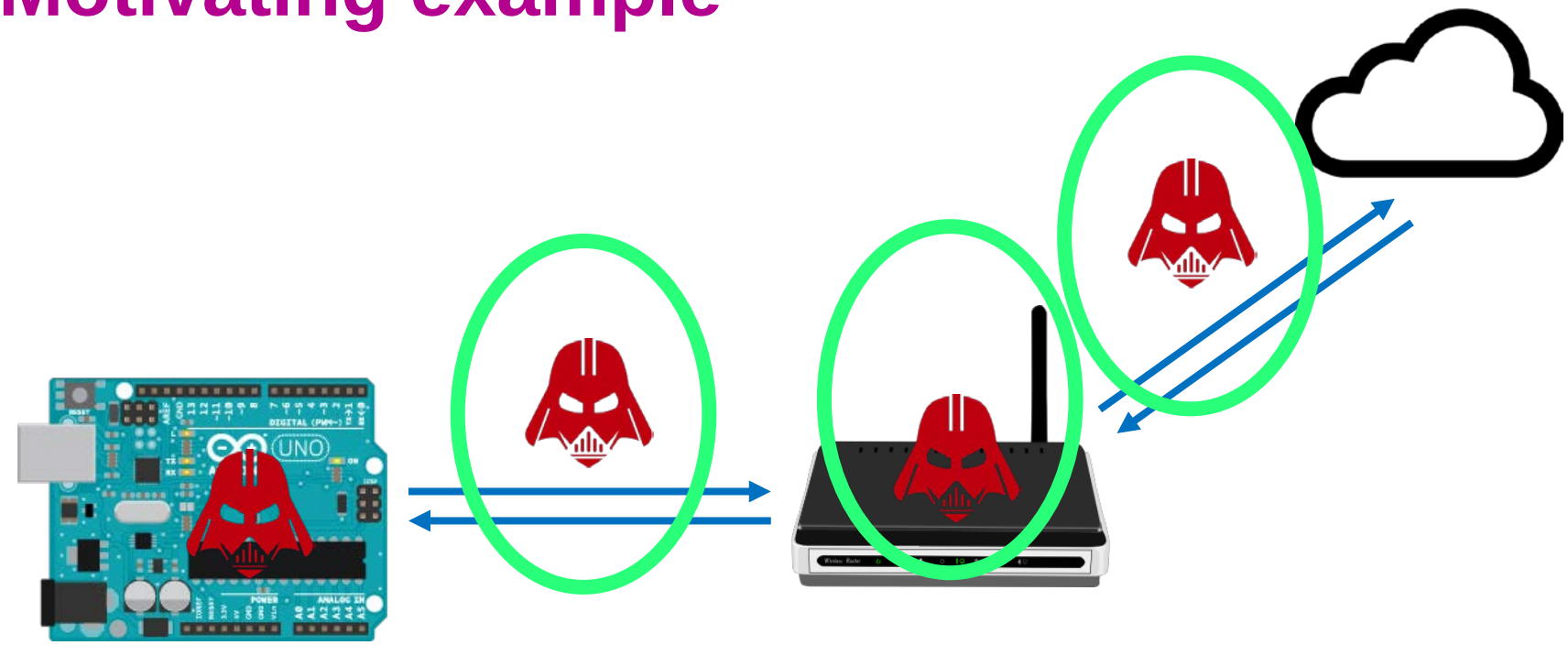
What does it mean?

Motivating example



Network adversary: read, modify, falsify communication

Motivating example



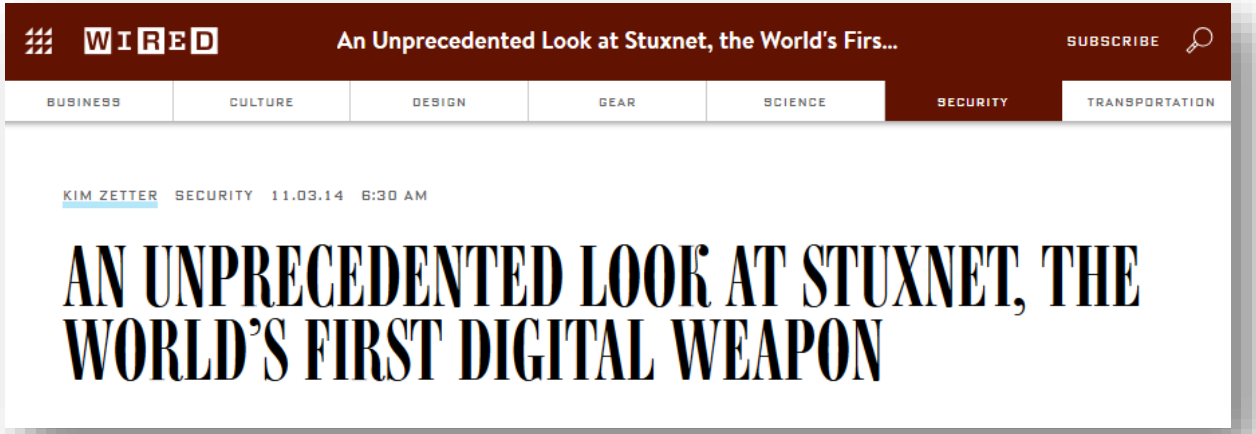
Network adversary: read, modify, falsify communication

✓ authenticated, integrity-protected communication

Malware: extract secrets, change state, modify behaviour

Physical adversary: has physical access to device

IoT malware



https://www.wired.com/2014/11/countdown-to-zero-day-stuxnet/

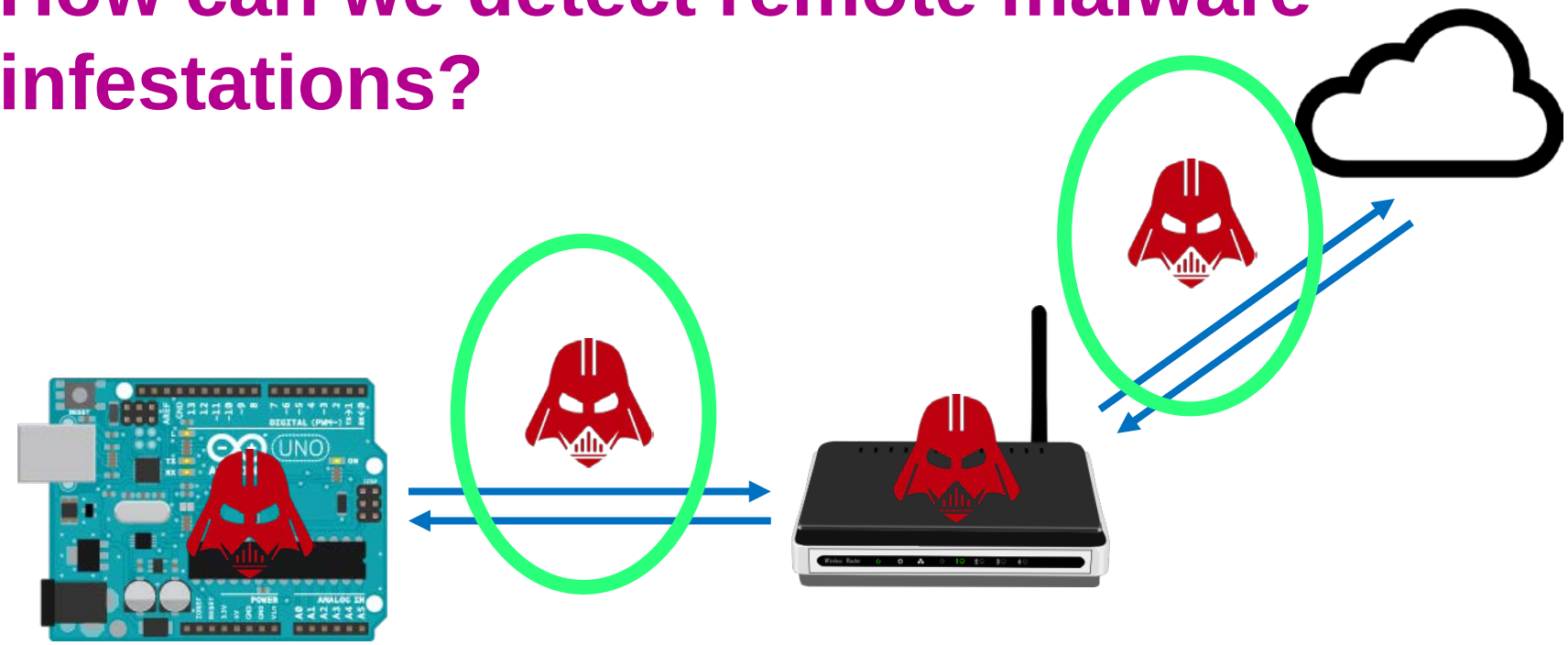
http://www.zdnet.com/article/iot-malware-and-ransomware-attacks-on-the-incline-intel-security/

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

How can we detect remote malware infestations?



Network adversary: read, modify, falsify communication

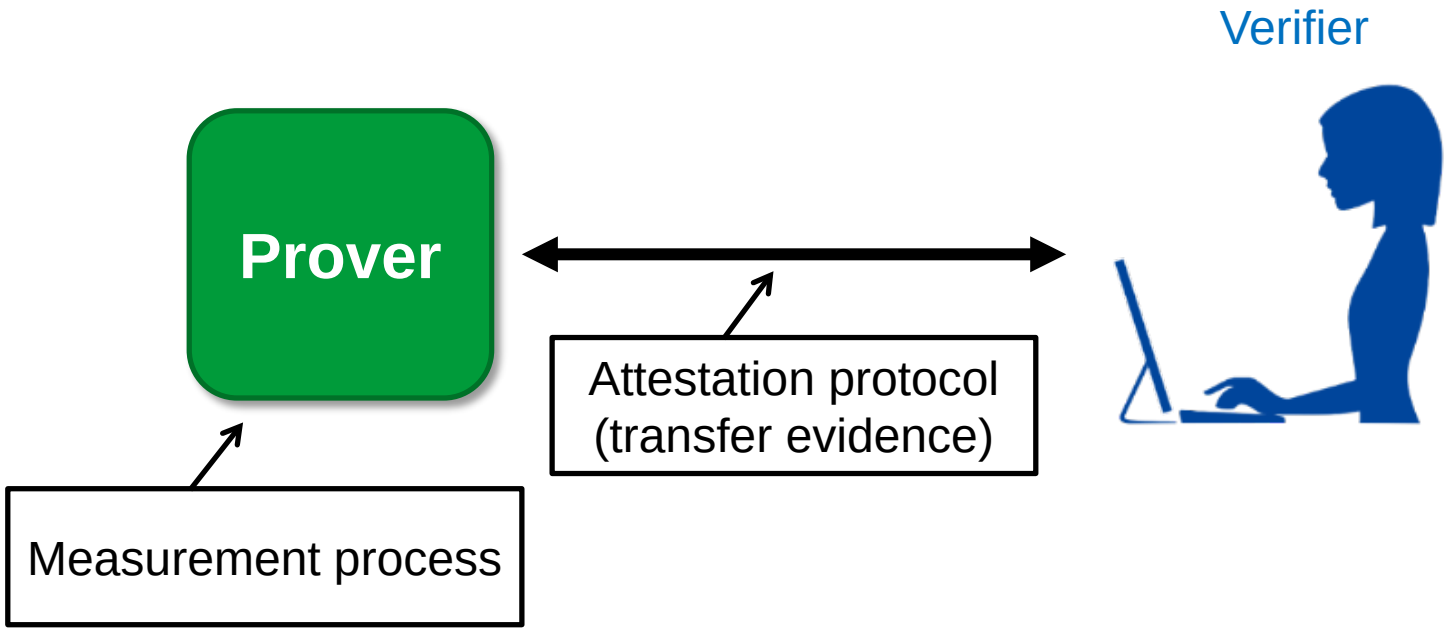
✓ authenticated, integrity-protected communication

Malware: extract secrets, change state, modify behaviour

Physical adversary: has physical access to device

Attestation

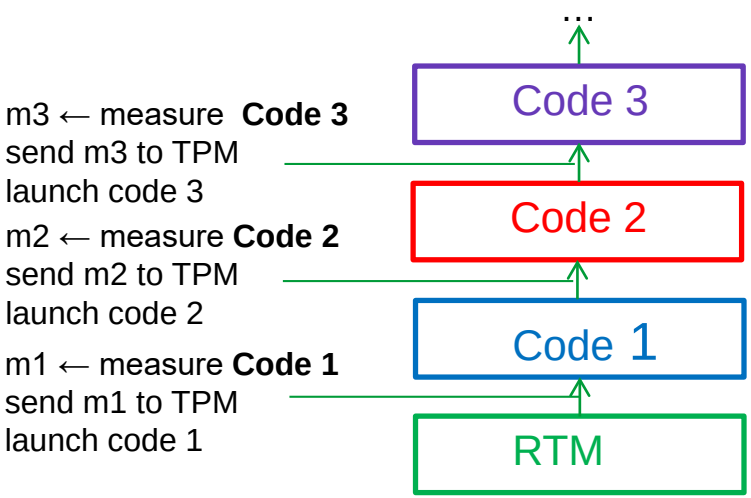
Verifier ascertains current state and/or behaviour of *prover*.



Measurement: aggregated hashing

Platform Configuration Registers (PCRs) store aggregated platform “state” measurement

- Requires a root of trust for measurement (RTM)
- A given state reached ONLY via correct extension sequence
  - “PCR extension rule”



$H_{new} = H(H_{old} \mid new)$ *[PCR extension rule]*

$H_0 = 0$

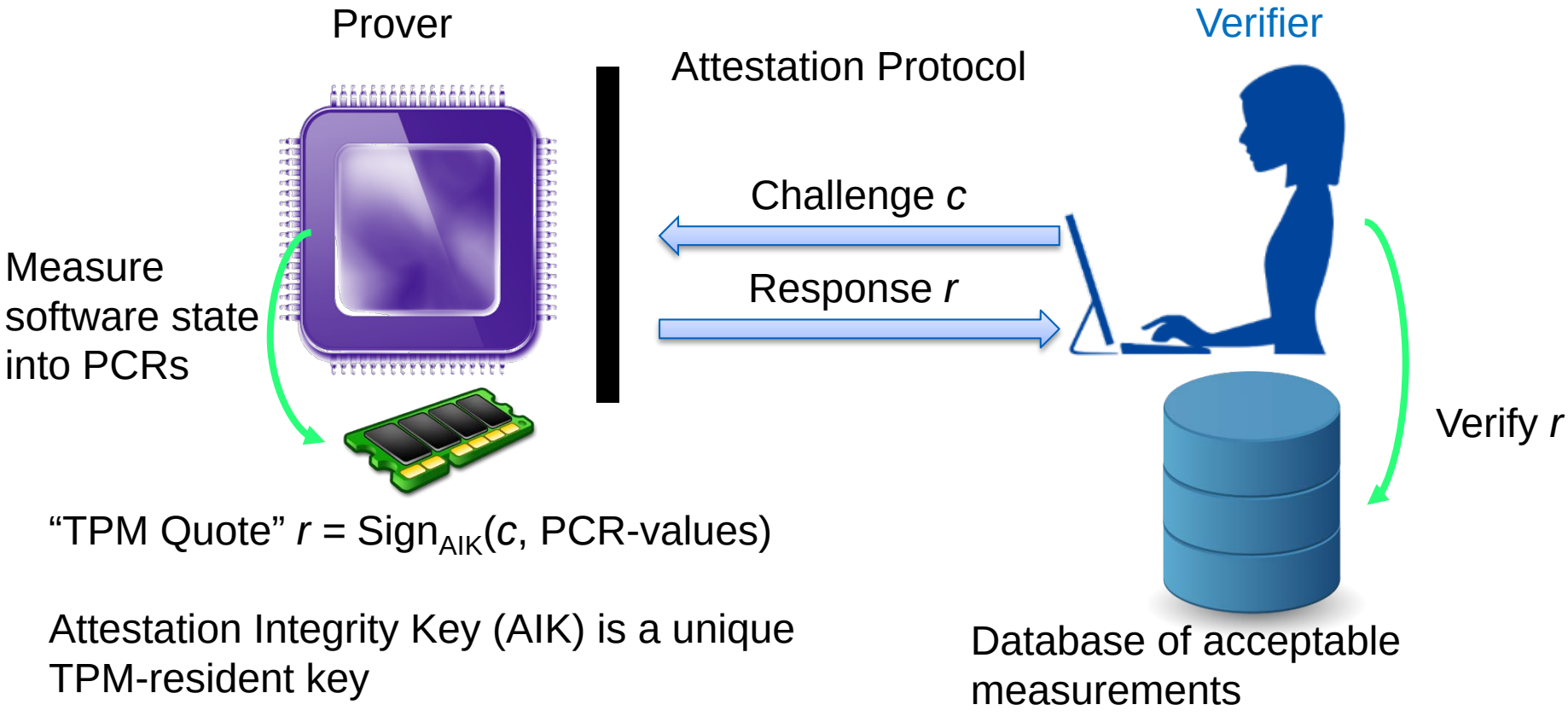
$H_1 = H(0|m1)$

$H_2 = H(H(0|m1)|m2)$

$H_3 = H(H(H(0|m1)|m2)|m3)$

Attestation Protocol

- **Goal**: Check whether the prover is in a trustworthy state



Drawbacks of Traditional Attestation

- Needs additional hardware and software
- Not suitable for “anaemic” provers
- Covers only the initial loading of software
- Deals with only one prover and one verifier
- Database of acceptable measurements does not scale

Attestation of Things (AoT)

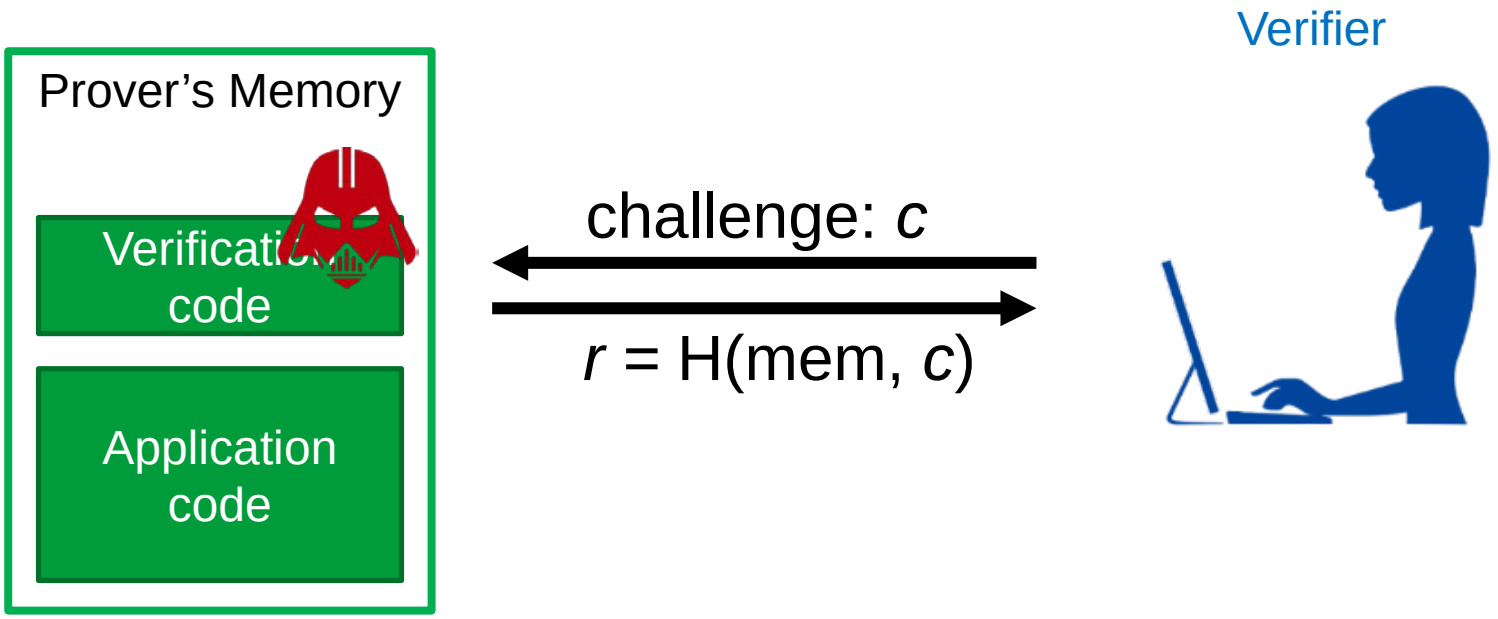
- Software-based attestation
- Hybrid attestation
- Scalability of attestation
- Run-time attestation

Attestation of Things (AoT)

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

Software-based Attestation

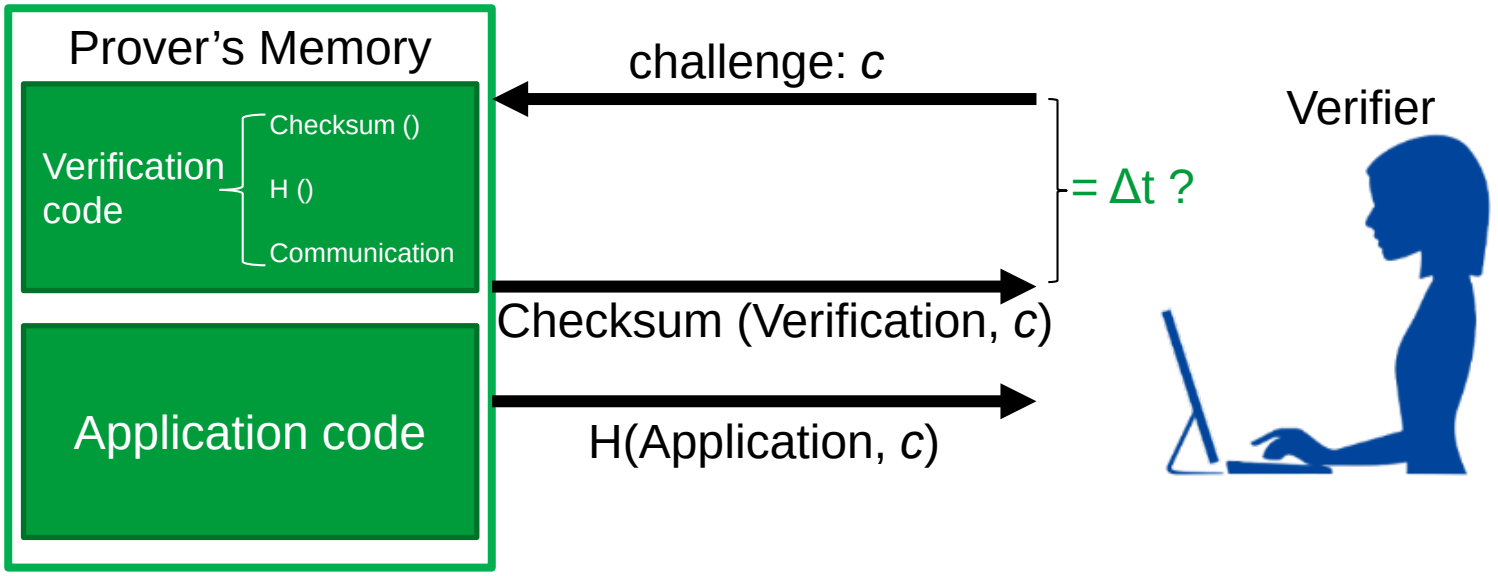
- Assumes no hardware features to support attestation
  - No secrets on prover (e.g. no AIK)



Software-based Attestation

- Pioneer system
  - compute time-optimal checksum of verifier

Authenticity?



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

Software-based Attestation: summary

Limitations of timing side channels

- verifier must know exact hardware configuration
- difficult to prove time-optimality
- assumes “adversarial silence” during attestation
- limited to “one-hop” networks
  - requires authenticated channel (e.g. physical connection)

Attestation of Things (AoT)

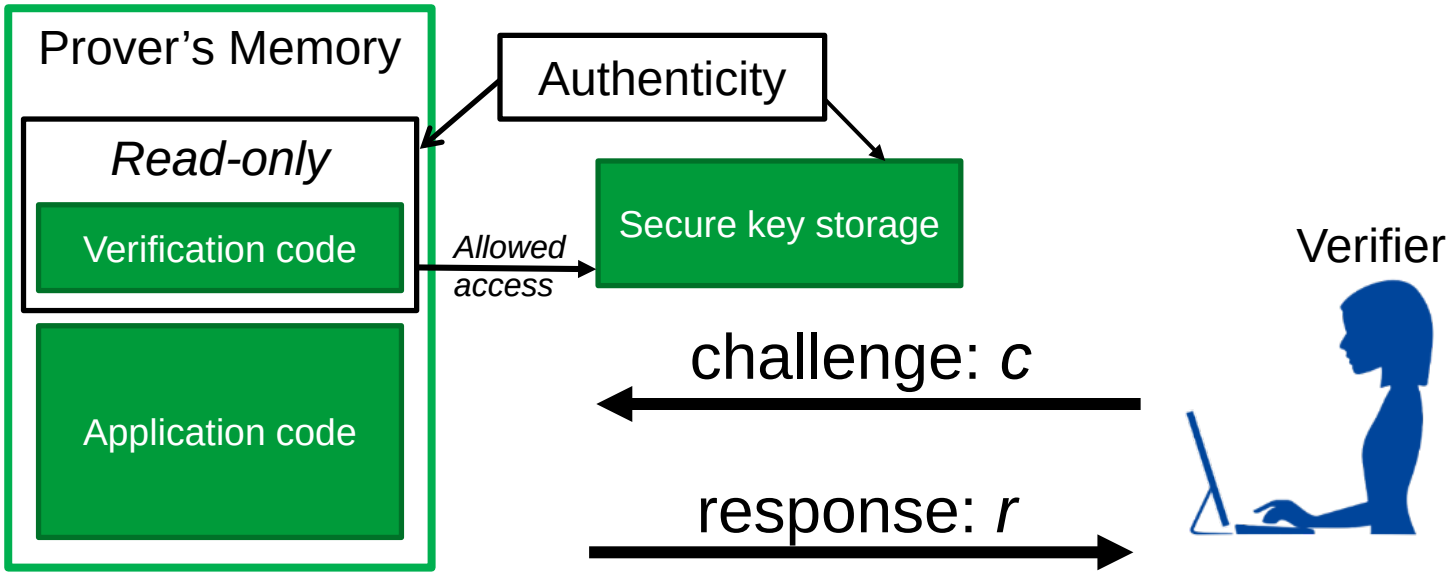
- Software-based attestation
- Hybrid attestation
- Scalability of attestation
- Run-time attestation

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

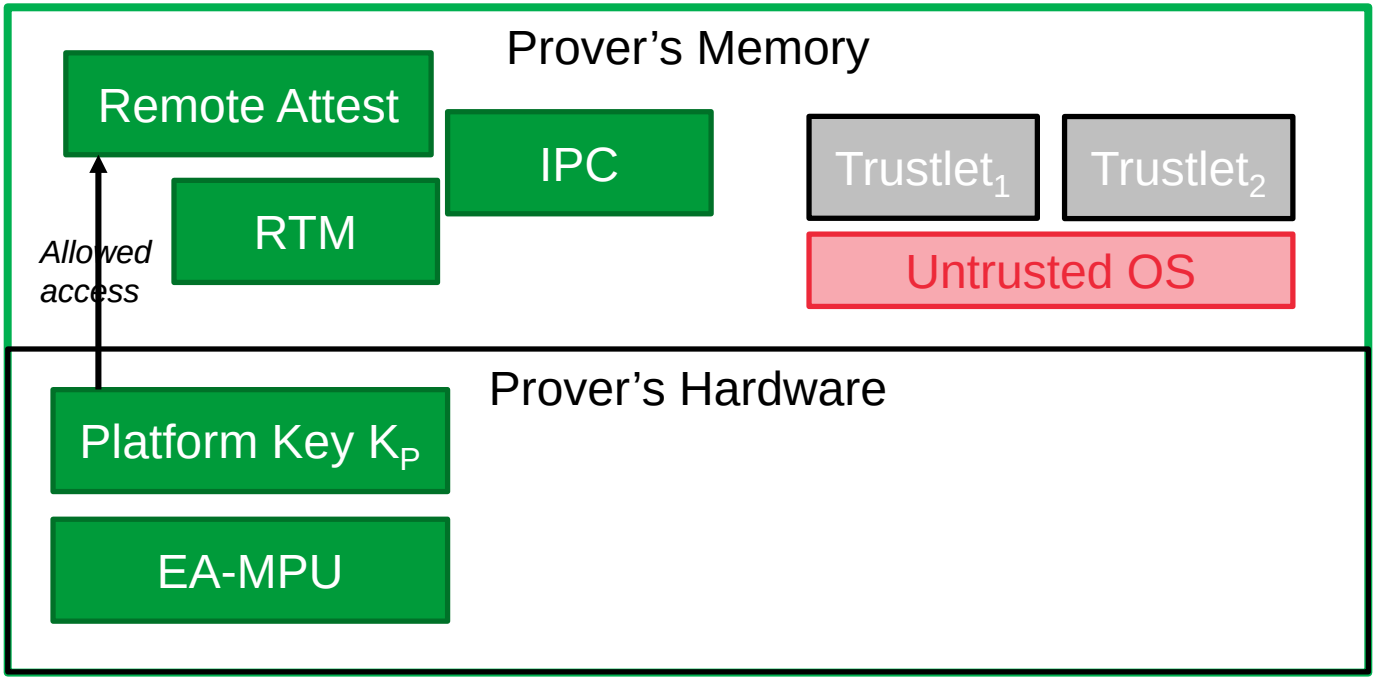


K El Defrawy, A. Francillon, D. Perito, and G. Tsudik. **SMART: Secure and Minimal Architecture for (Establishing a Dynamic) Root of Trust**. NDSS '12

A. Francillon, Q. Nguyen, K. Rasmussen and G. Tsudik, **A Minimalist Approach to Remote Attestation**, DATE 2014

Hybrid Attestation: TrustLite & TyTAN

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



P. Koeberl, S. Schulz, A.-R. Sadeghi, and V. Varadharajan. **TrustLite: A Security Architecture for Tiny Embedded Devices**. EuroSys '14

F. Brasser, B. El Mahjoub, A.-R. Sadeghi, P. Koeberl, **TyTAN: Tiny Trust Anchor for Tiny Devices,** DAC '15

Hybrid Attestation: summary

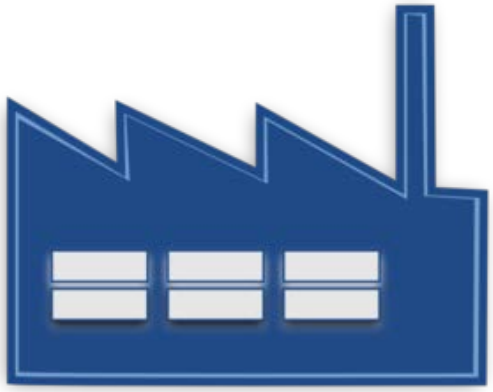
- Advantages of hybrid approaches
  - can be used across a network / over an untrusted channel
  - Verifier need not know prover's exact hardware configuration
- Drawbacks
  - Needs additional hardware support
  - But minimal MCU trust anchors soon available commercially
    - TrustZone-M (ARM v9), …

Attestation of Things (AoT)

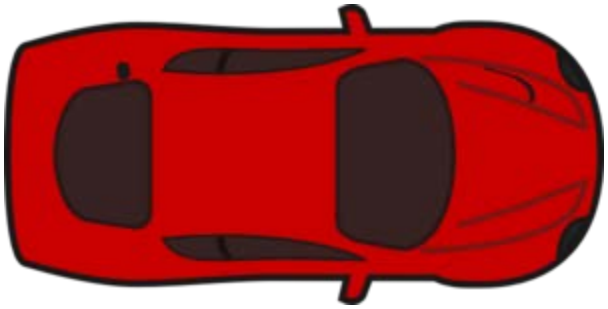
- Software-based attestation
- Hybrid attestation
- Scalability of attestation
- Run-time attestation

Scalability of Attestation

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



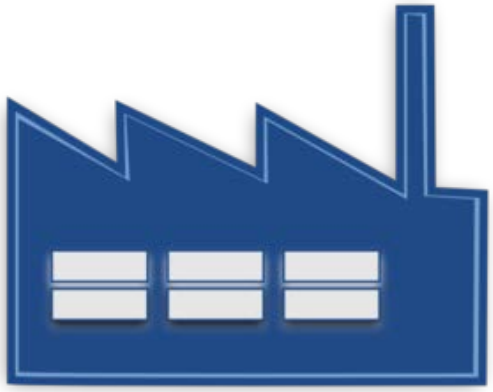
Smart factories



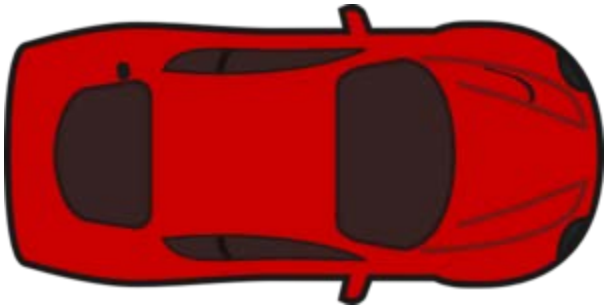
Smart vehicles

Scalability of Attestation

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



Smart factories

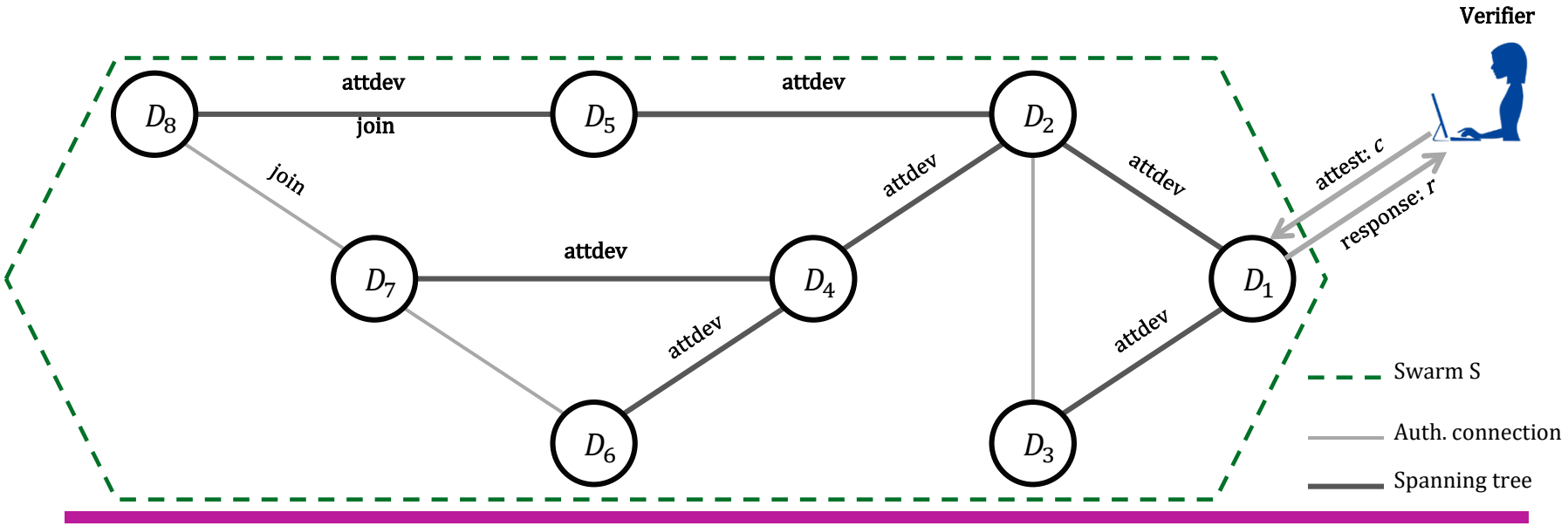


Smart vehicles

Scalability of Attestation

SEDA: Scalable Embedded Device Attestation

- more efficient than attesting each node individually
- can use any type of measurement process



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

Moreno Ambrosin, Mauro Conti, Ahmad Ibrahim, Gregory Neven, Ahmad-Reza Sadeghi, Matthias Schunter. **SANA: Secure and Scalable Aggregate Network Attestation**. CCS '16

Scalability of Attestation: summary

How to extend SEDA to

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

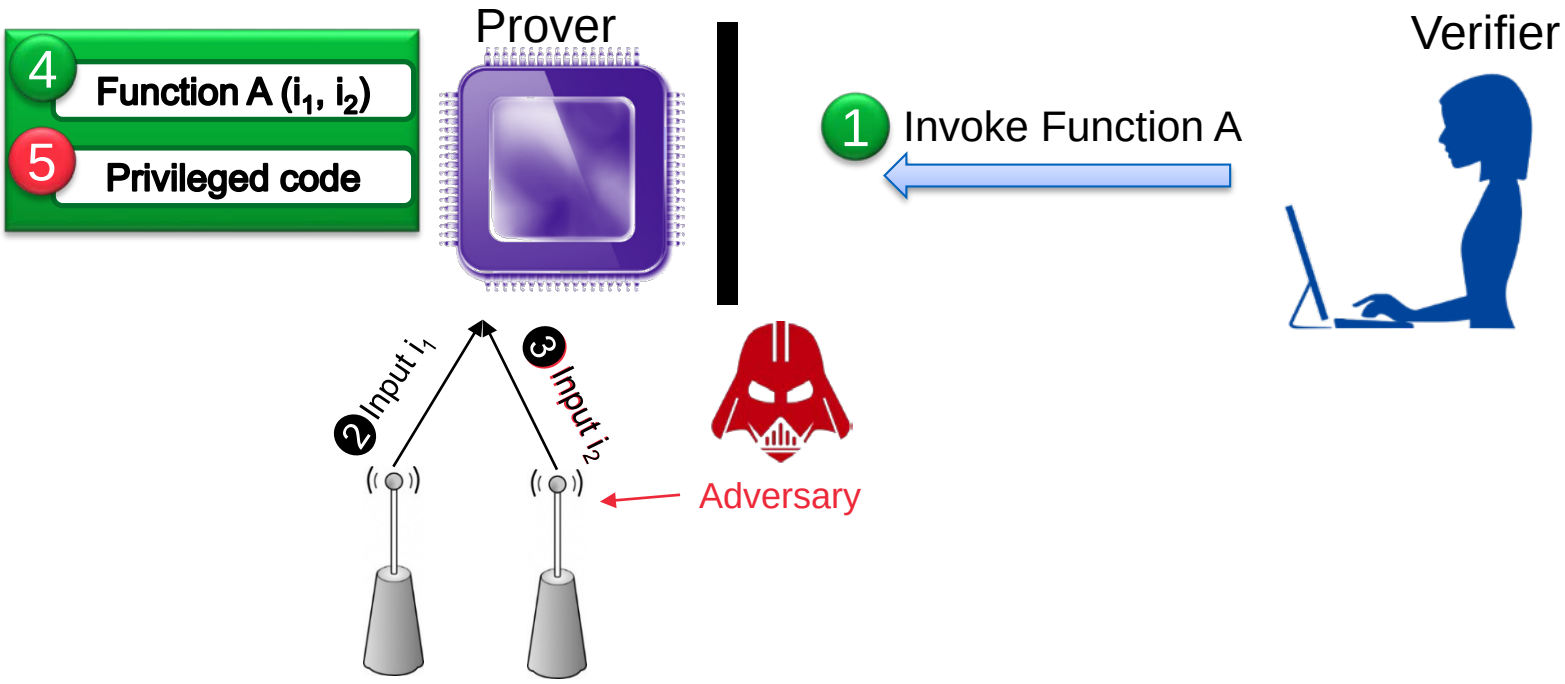
Attestation of Things (AoT)

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

Why run-time attestation?

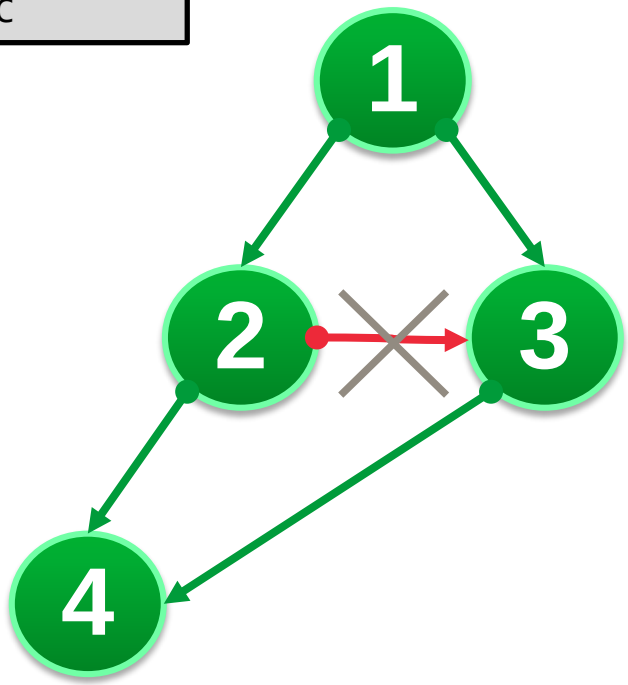
- Traditional attestation measures binaries at load time
- Cannot capture run-time attacks
  - return-oriented programming
  - control data attacks

Run-time attacks



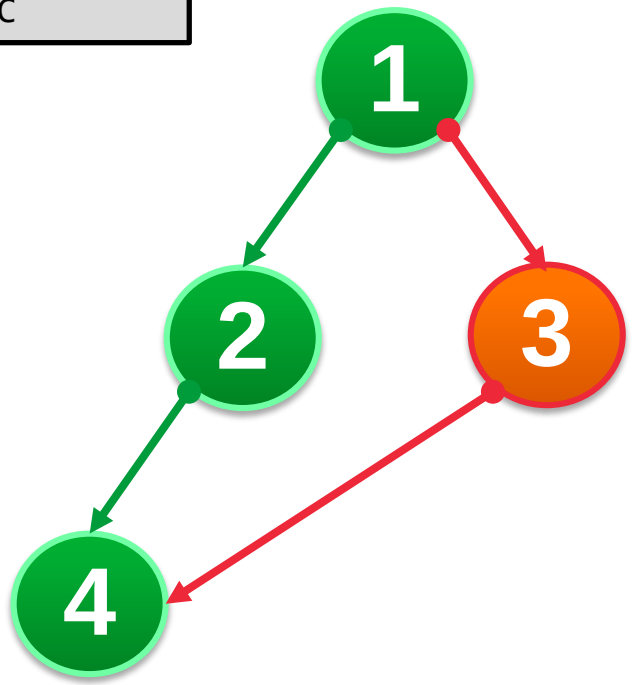
Control flow integrity (CFI)

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



Run-time attacks without violating CFI

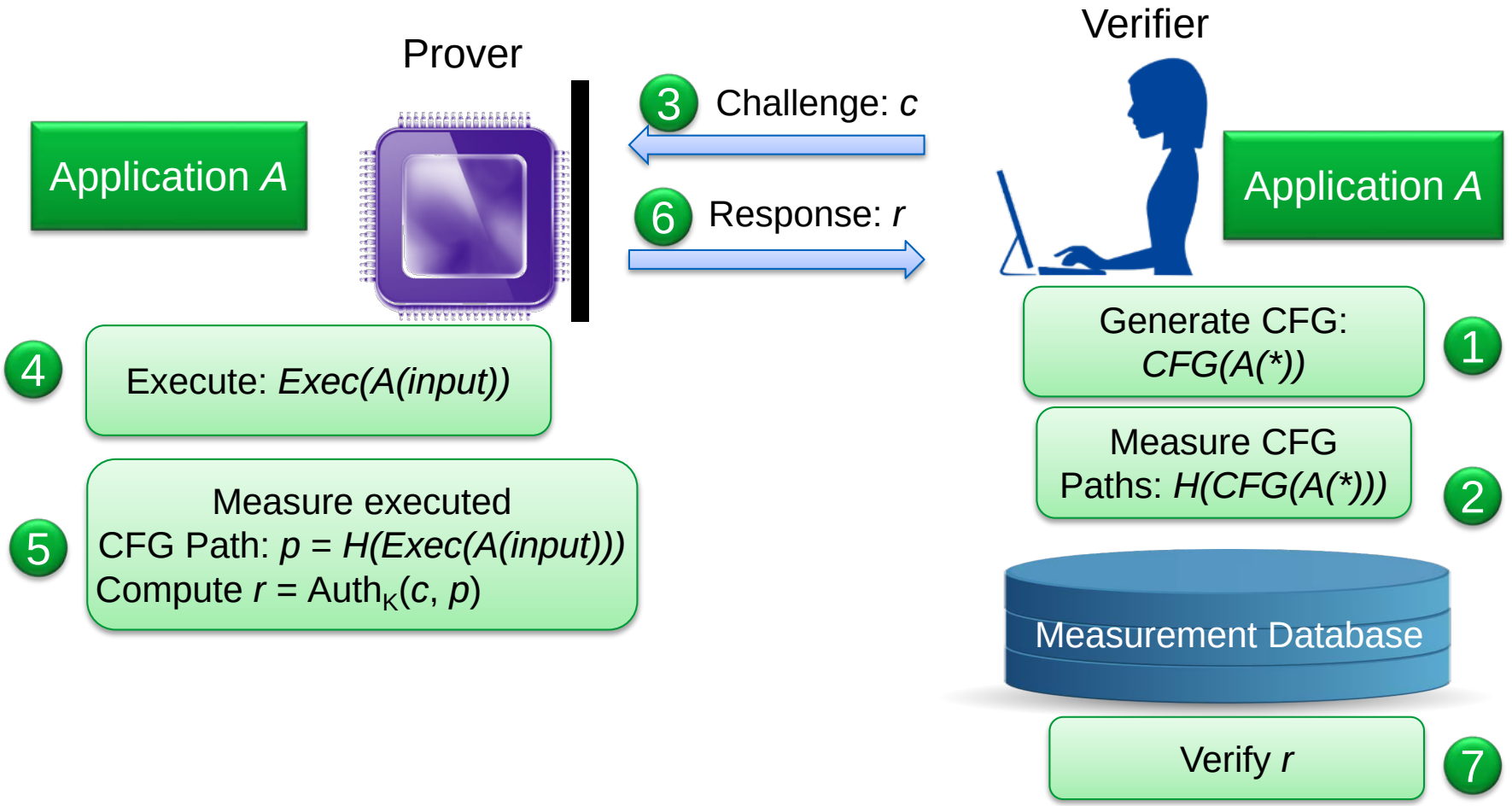
```
① if (cond)
②  then: block_A
③  else: block_B
④ block_C
```



Normal operation

Privileged operation

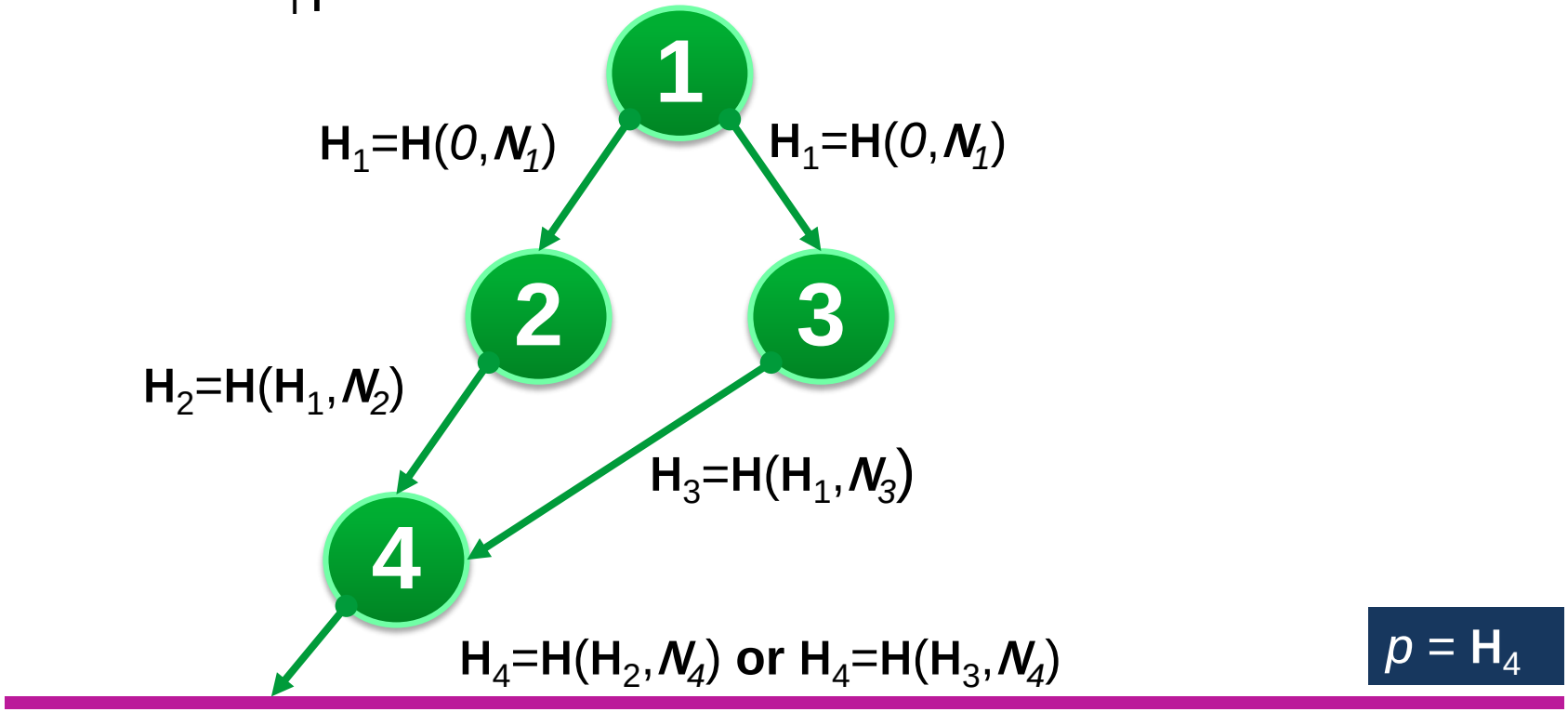
Control-Flow Attestation (C-FLAT)



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**, http://arxiv.org/abs/1605.07763, CCS '16.

C-FLAT: High-Level Idea

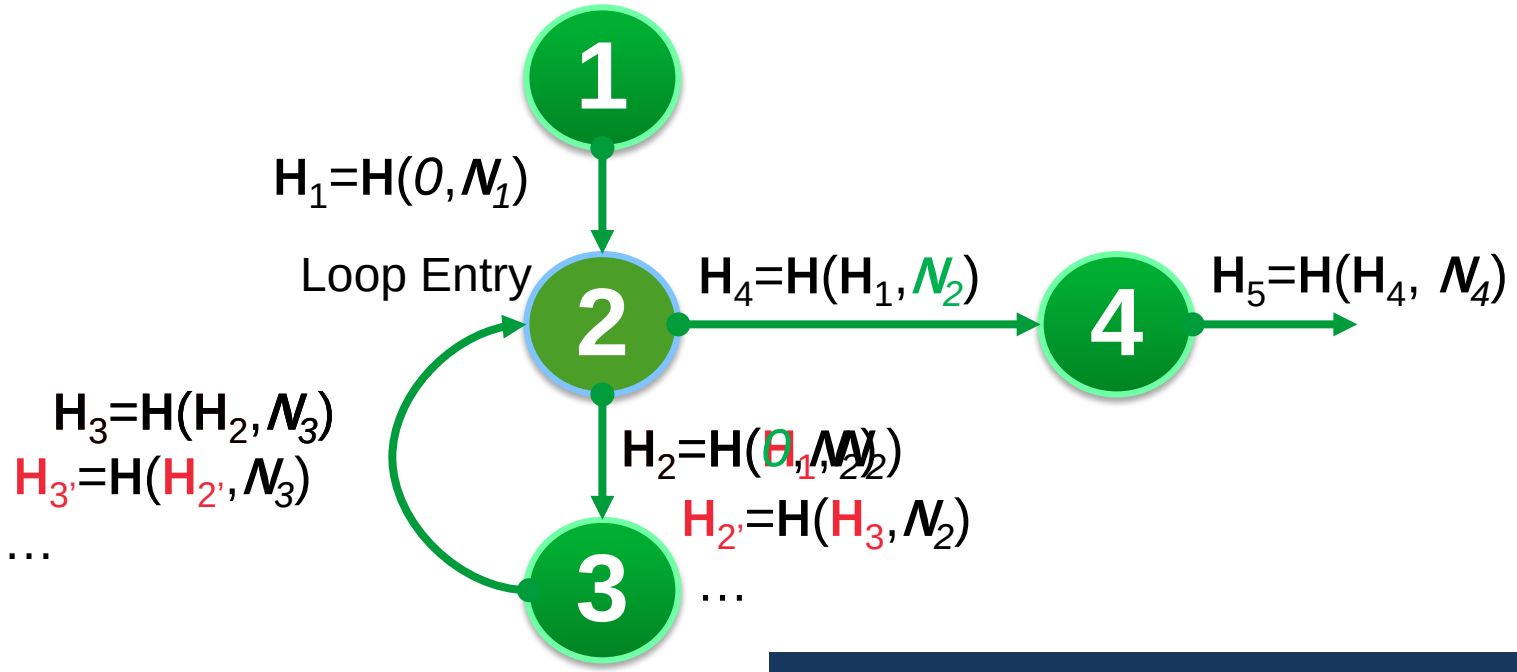
- Cumulative Hash Value: $\mathbf{H}_j=\mathbf{H}(\mathbf{H}_i,N)$, where $\mathbf{H}_i$ previous hash result and N is the current node



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**, http://arxiv.org/abs/1605.07763, CCS '16.

Handling Loops

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



H_x different for each loop iteration

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**, http://arxiv.org/abs/1605.07763, CCS '16.

Proof-of-Concept Implementation

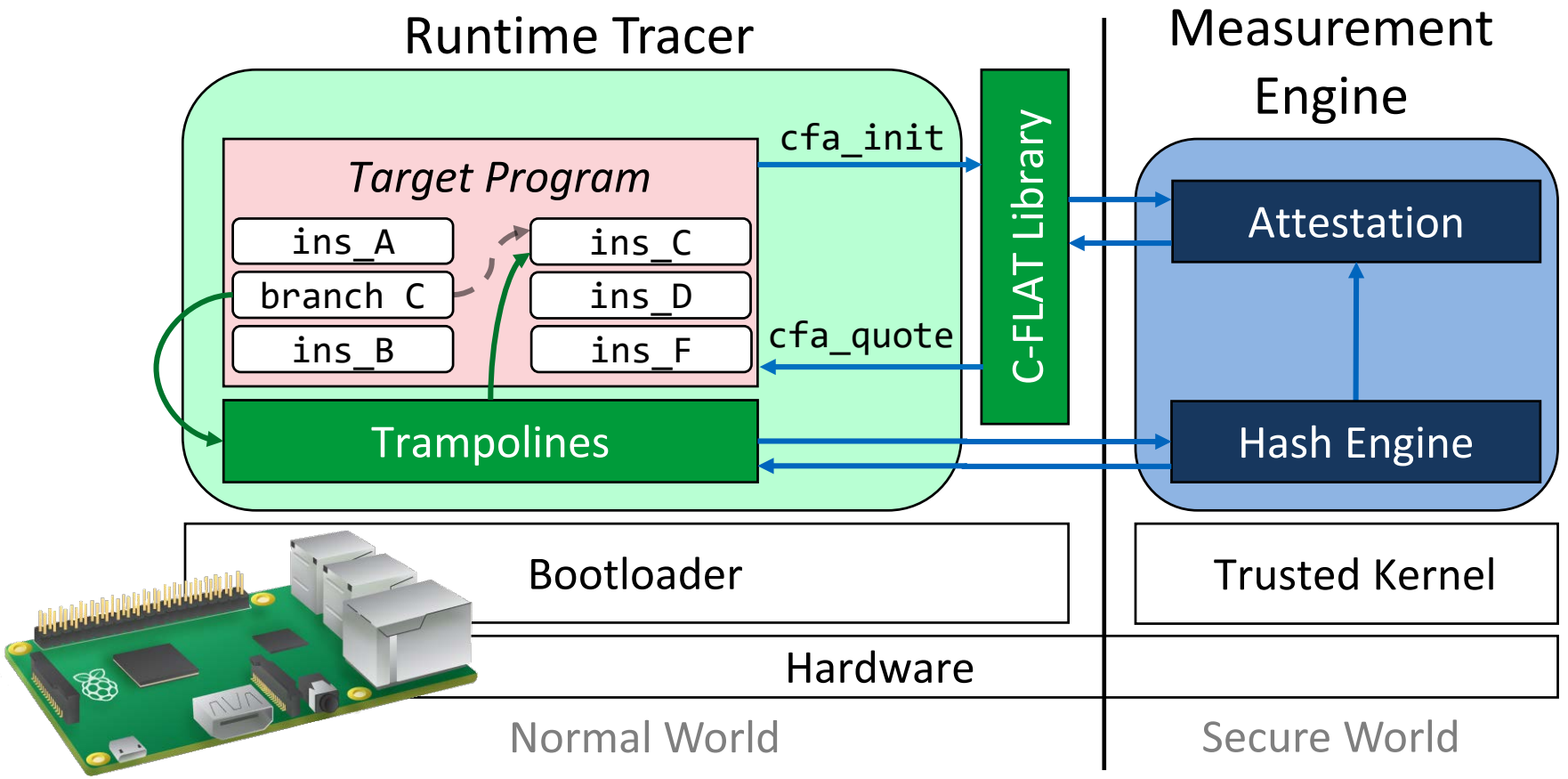
- Bare-metal prototype on Raspberry Pi 2
  - Single-purpose program instrumented using binary-rewriting
  - Runtime Monitor written in ARM assembler
  - Measurement Engine isolated in TrustZone-A Secure World

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

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**, http://arxiv.org/abs/1605.07763, CCS '16.

Proof-of-Concept Implementation



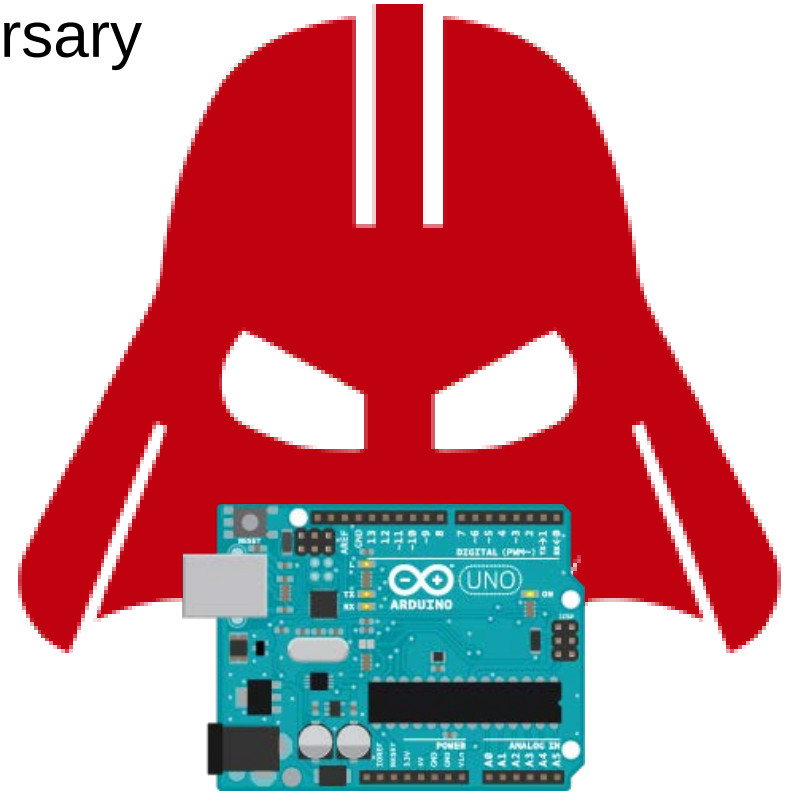
T. Abera, N. Asokan, L. Davi, J-E. Ekberg, A. Paverd, T. Nyman, A-R. Sadeghi, G. Tsudik, **C-FLAT: Control Flow Attestation for Embedded Systems Software**, http://arxiv.org/abs/1605.07763, CCS '16.

Run-time attestation: summary

- How can we scale control flow attestation?
  - Better ways to aggregate measurements?
  - Faster/simpler purpose-built hash functions?
  - Attestation of properties rather than measurements?
    - from attestation to checking compliance with a (dynamic) policy?

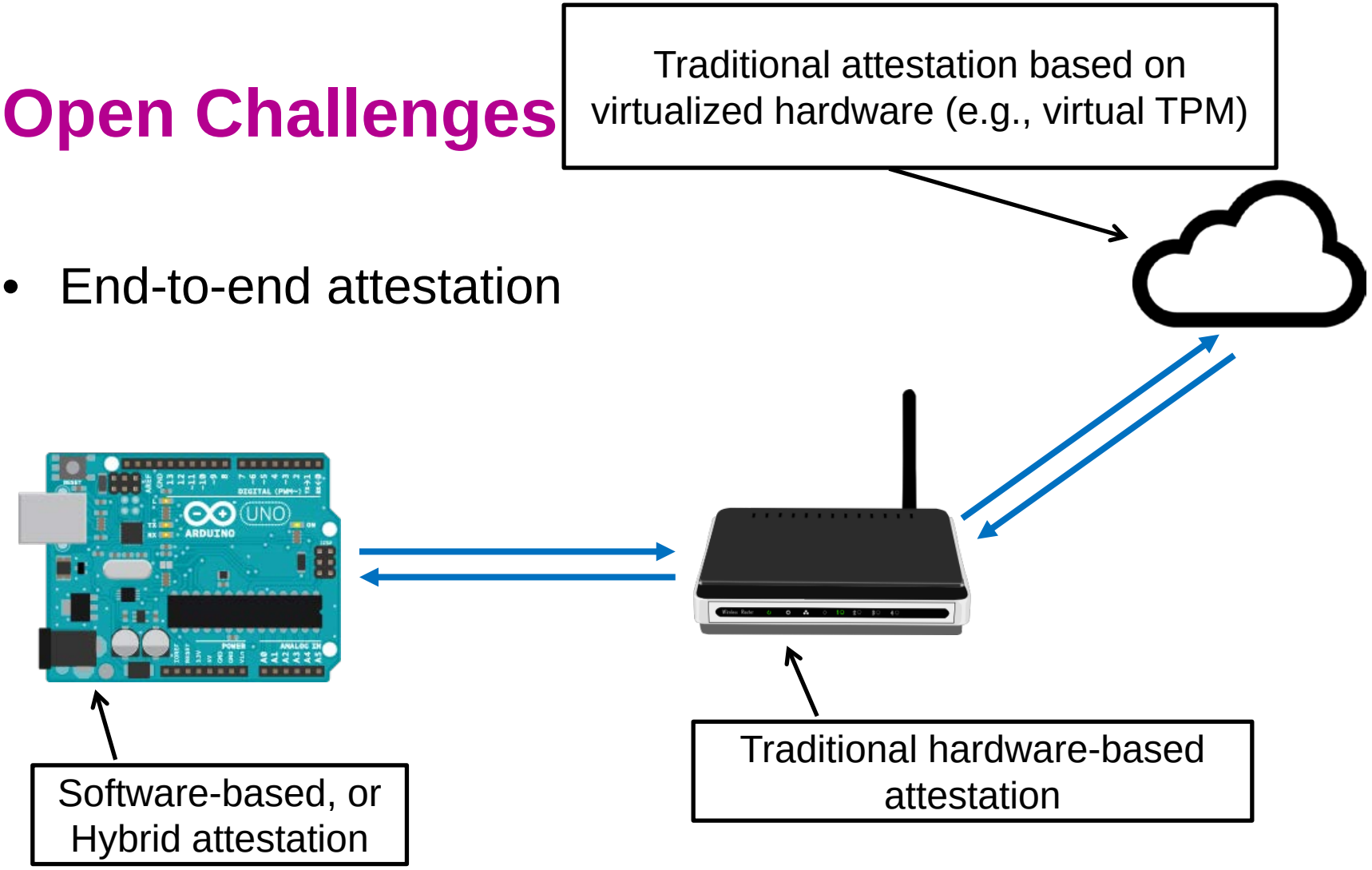
Open Challenges

- Physical adversary



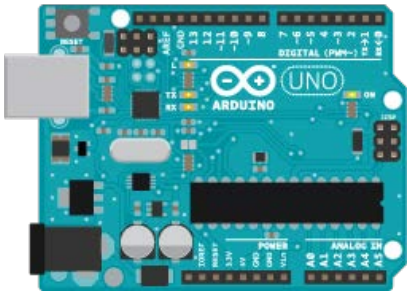
Open Challenges

- End-to-end attestation



Open Challenges

- Device heterogeneity



Summary

- Attestation is useful to establish trust in a remote device
- Traditional attestation not applicable to IoT settings
  - Too heavy: Resource-constrained devices
  - Too limiting: need run-time attestation, more scalability
- Recent/current research in addressing these gaps
  - Open issues remain