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

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

IoT malware and ransomware attacks on the incline: Intel Security

Intel Security has released a five-year retrospective report on industry threats, finding people have become dependent on devices at the cost to their security and privacy, allowing malware and ransomware attacks to rapidly grow.

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
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”

\[ H_{\text{new}} = H(H_{\text{old}} | \text{new}) \]

\[ H_0 = 0 \]
\[ H_1 = H(0|m_1) \]
\[ H_2 = H(H(0|m_1) |m_2) \]
\[ H_3 = H(H(H(0|m_1) |m_2)|m_3) \]
Attestation Protocol

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

Measure software state into PCRs

"TPM Quote" \( r = \text{Sign}_{\text{AIK}}(c, \text{PCR-values}) \)

Attestation Integrity Key (AIK) is a unique TPM-resident key
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)

\[
\text{Prover's Memory} \\
\text{Verification code} \\
\text{Application code} \\
\text{Verifier}
\]

\[
\text{challenge: } c \\
\text{r = H(mem, c)}
\]
Software-based Attestation

• Pioneer system
  – compute time-optimal checksum of verifier

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

Hybrid Attestation: TrustLite & TyTAN

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

---


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
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

---


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

1. Invoke Function A

2. Input $i_1$

3. Input $i_2$

4. Function A ($i_1$, $i_2$)

5. Privileged code

Prover

Verifier

Adversary
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

Diagram:

- Normal operation
- Privileged operation
Control-Flow Attestation (C-FLAT)

33

Prover

Application A

Execute: $\text{Exec}(A(\text{input}))$

Measure executed
CFG Path: $p = H(\text{Exec}(A(\text{input})))$

Compute $r = \text{Auth}_K(c, p)$

Verifier

Application A

Generate CFG: $\text{CFG}(A(\ast))$

Measure CFG Paths: $H(\text{CFG}(A(\ast)))$

Measurement Database

Verify $r$

1

2

3 Challenge: $c$

4

5

6 Response: $r$

7

C-FLAT: High-Level Idea

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

$$
H_1 = H(0, N_1) \quad H_1 = H(0, N_1) \\
H_2 = H(H_1, N_2) \\
H_3 = H(H_1, N_3) \\
H_4 = H(H_2, N_4) \text{ or } H_4 = H(H_3, N_4)
$$

$p = H_4$
Handling Loops

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

\[ H_1 = H(0, N_1) \]
\[ H_2 = H(H_1, N_2) \]
\[ H_3 = H(H_2, N_3) \]
\[ H_4 = H(H_3, N_3') \]
\[ H_5 = H(H_4, N_4) \]

\[ p = H_5, <H_1, \{H_3, \#H_3}> \]

\( H_x \) different for each loop iteration

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

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

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

Software-based, or Hybrid attestation

Traditional attestation based on virtualized hardware (e.g., virtual TPM)

Traditional hardware-based 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