Caveat (IoT) Emptor: Towards Transparency of IoT Device Presence

Sashidhar Jakkamsetti, Youngil Kim, and Gene Tsudik
CS Department
University of California, Irvine
gene.tsudik@uci.edu

To appear at ACM CCS 2023

1

sprout.ics.uci.edu

SPROUT - Security and Privacy Research OUTfit

- 7 PhD students, a few visitors
- ca. 30 PhD Alumni

Current Topics:

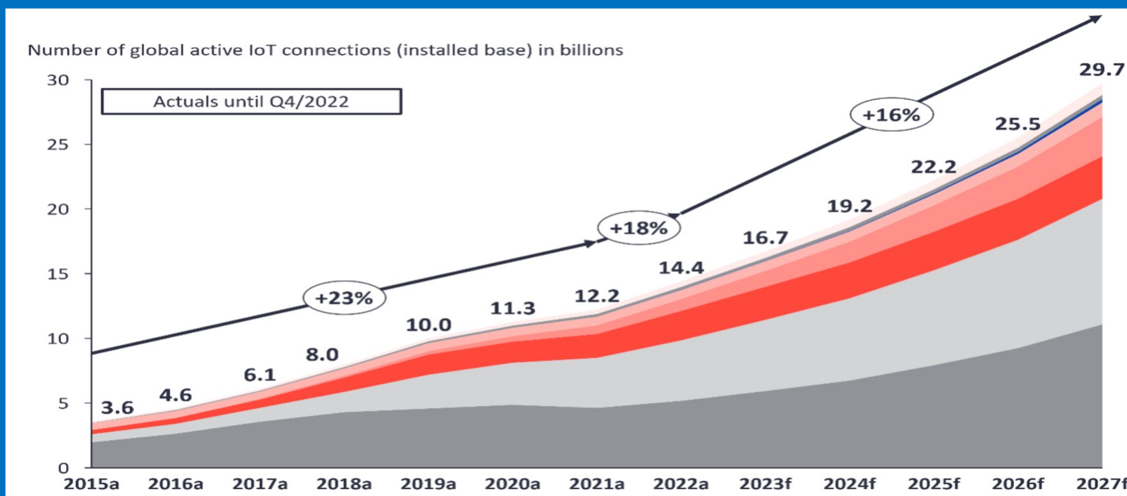
- **Security and Privacy for Embedded (ES/CPS/IoT/smart) Devices**
- Large-Scale Anonymity
- Privacy-Agile Cryptographic Techniques
- **Fun things with secure hardware components (TEE-s)**
- Usable security in different contexts (e.g., CAPTCHAs)
- Biometrics + De-authentication + Side-Channels + Attacks

2

2

Popularity of IoT Devices

Over 14 billion deployed and growing

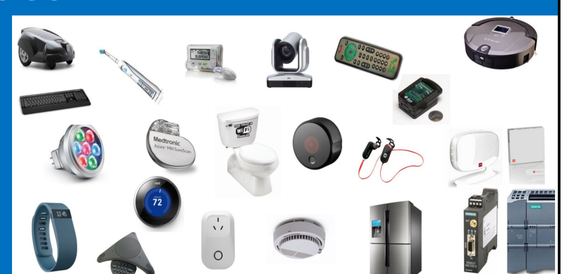


3

3

The IoT Ecosystem

- Ubiquitous, diverse and growing
- Widely used in
  - Smart home/office applications
  - Smart cities and industrial automation
- Functionality: sensors, actuators, control units
- Range from high-end (almost smartphone-like) to very low-end (amoebas)
- Interconnected and/or Internet-connected
- Attractive targets for attacks and malware (e.g., Stuxnet, Mirai Botnet)
- Attacks targets: privacy, security/safety, zombification
- This will get a lot worse...



4

Attacks

METRO

Staten Island creep hacks into Ring security camera to spy on teenager

By Larry Celona, Reuven Fenton and Natalie Musumeci
Published Dec. 13, 2019 | Updated Dec. 13, 2019, 3:48 p.m. ET

Is your TV, smart speaker and video doorbell spying on you? Worst offenders exposed

If you own a smart device then it might be gathering more data than you think.

By Dave Snelling, Technology Editor
11:42, 7 Sep 2023

CYBERSECURITY

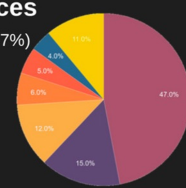
Bugs in This Brand of 'Smart' Garage Doors Could Allow a Hacker to Open Them at Will

A host of software flaws in products sold by the smart home company Nexx makes them ripe for manipulation by unscrupulous cyber punks.

By Lucas Ropek Published April 5, 2023

Most Hacked Devices

Security Camera Systems (47%)
Smart Hubs (15%)
NAS (12%)
Printers (6%)
Smart TVs (5%)
IP Phones (4.3%)



Vulnerabilities/Threats | 4 MIN READ | NEWS

Hey, Siri: Hackers Can Control Smart Devices Using Inaudible Sounds

A technique, dubbed the "Near-Ultrasound Inaudible Trojan" (NUIT), allows an attacker to exploit smartphones and smart speakers over the Internet, using sounds undetectable by humans.

Robert Lemos
Contributing Writer, Dark Reading
March 28, 2023

5

5

Why?

- Device-makers don't prioritize security or privacy
- Budget constraints: $$$ cost, size, performance, bandwidth, etc.
- Rush-to-market syndrome
- Malleable software/firmware
- Lack of assurance (no verification) of hw or sw
- Consumer tendency towards monocultures, e.g., Echo VA, Ring DB, Nest,…
    - Compromise one → compromise all
- **Bottomline: The "IoT Armageddon" is coming…**

6

Motivation

IoT Security guidelines

- Do not consider user privacy in the general sense
- Aimed at device owners or operators

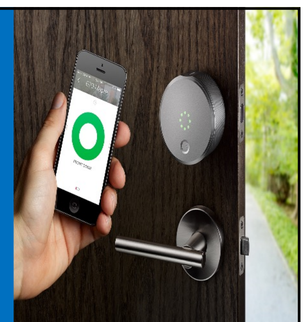
Data protection regulations

- Service providers must ask for user consent before collecting, processing, storing, and sharing user data
- Aimed at web sites that collect information
- IoT devices don't just sense and/or actuate alone → sensed data and actuation commands are propagated (over the Internet) to digital twins on the web/cloud

7

7

Motivation



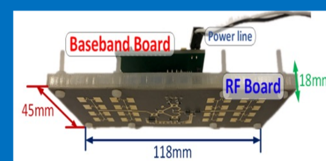
- Our scope/goal:
  - Consider all nearby (potentially impacted) users, not just owners
  - Include both actuation and sensing capabilities
  - Use compliance-based device-architectural approach
    - Preferably, with no hardware modifications
- Examples:
  - AirBnB renters (or hotel guests) can be locked out if smart locks are hacked. Could disable them if presence known in advance
  - Meeting participants can be (even accidentally) recorded without their knowledge by smart cams. Could ask to unplug if informed in advance.
- **Question**: How to make nearby users aware of IoT device presence and capabilities
- **End-Goal**: help users make informed privacy/security/safety decisions

8

8

An Obvious Alternative: Detection

- Specialized Equipment
  - Users must have special/custom hardware
  - Not guaranteed to detect all devices
  - Many communication media types
  - Communication can be stealthy
- Network Traffic Analysis
  - Need time to perform network traffic analysis
  - Probabilistic detection → error-prone
  - Not useful for devices that communicate seldom



9

9

Other Alternatives

- Scanning QR Codes (affixed to devices or elsewhere)
  - Need line of sight
  - Manual process, driven by users
  - Can miss some devices
- Registration-Based Approach, e.g., Personalized Privacy Assistant (CMU)
  - Infrastructure assists discovery of nearby IoT devices
  - E.g., a benevolent cloud repository of registered IoT devices (capabilities, location, etc.)
  - Informs users about the data practices associated with devices
  - Supports discovery of device settings (e.g., opt-in, opt-out, data erasure)

10

10

PAISA: Privacy-Agile IoT Sensing and Actuation

Announcements: IoT devices announce themselves to all nearby users

Requirements:

- ***Availability***: announcements must be generated at fixed time intervals
- ***Unforgeability***: announcements must have integrity and authenticity
- ***Freshness***: announcements must be fresh and reflect current/recent status of device software
- ***Casualness***: announcements must be receivable by all nearby users, with no prior associations and without establishing any secure channels

11

11

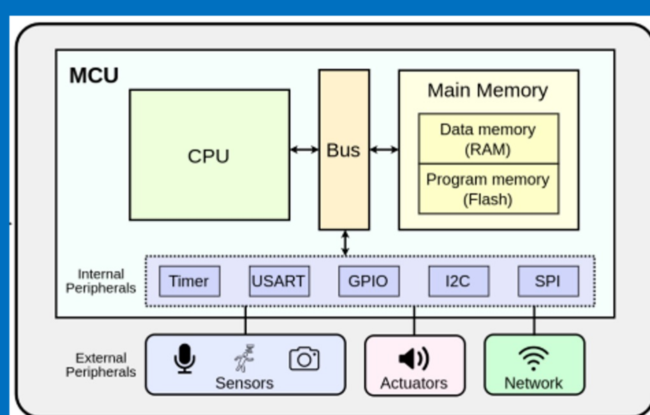
Background: IoT Devices

Sensors & Actuators

- Microphones, GPS units, cameras, smoke/CO2, and motion detection
- Speakers, light switches, door locks, alarms, sprinklers

Network Interfaces / Media:

- WiFi, Bluetooth, Zigbee, Cellular



12

12

Background: Trusted Execution Environments (TEEs)

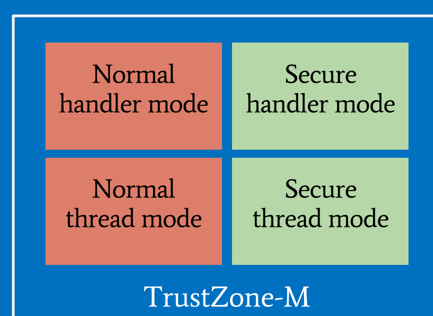
- Isolated execution (one or more "Enclaves"): Small piece of security-sensitive code that can be run in isolation from all other software on the same platform (including hypervisor/OS etc.)
- Persistent secure storage: Integrity-protected and roll-back protected storage, typically realized using "sealing" and a limited number of hardware-based monotonic counters
- Remote Attestation: Ability to prove to a remote party precisely what code is running in a **genuine** TEE
- Prominent Examples: TPM, Intel SGX, ARM TrustZone, etc.

13

13

Background: Trusted Execution Environments (TEEs)

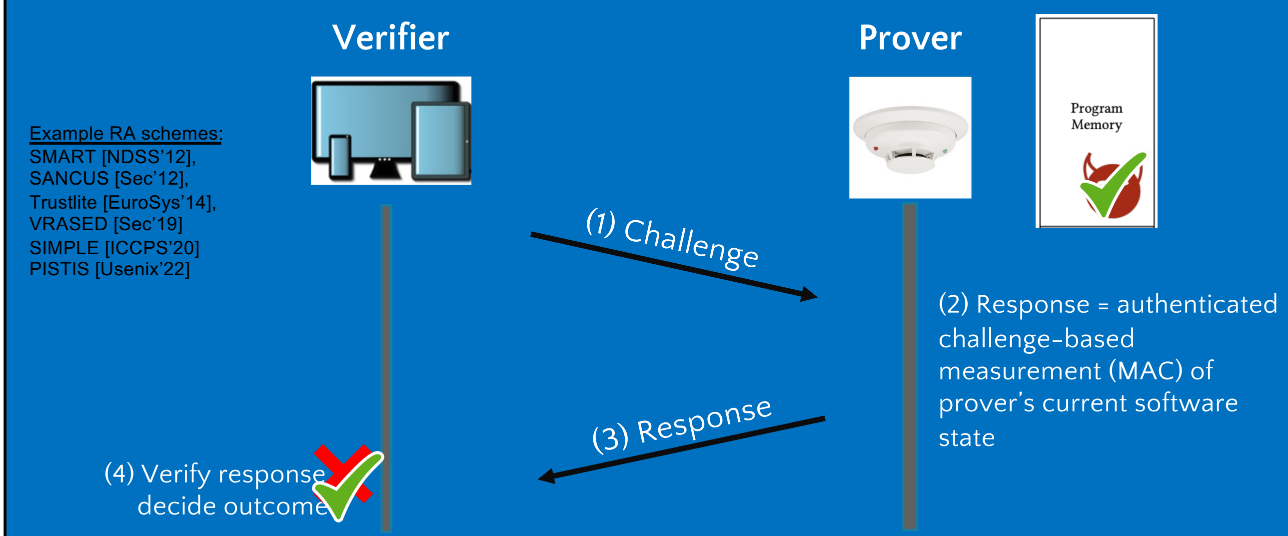
- ARM TrustZone-M (and TrustZone-A)
  - Available on ARM Cortex-M23/M33/M55 MCUs
  - Two isolated regions: **Secure World** and Normal World
  - Supports secure boot for code integrity
  - Can assign peripherals to **Secure World**
  - Raise *SecureFault* exception if a violation is detected
  - Non-Secure Callable (NSC) functions residing in Secure World can be called by Normal World application



14

14

Remote Attestation (RA)



15

15

RA use in PAISA

- IoT device (Prover) generates a random challenge and attests its own program memory with it, signs result
- Sends result to user device (Verifier) for checking

16

16

PAISA Entities and Phases

IoT Device (I_{dev})

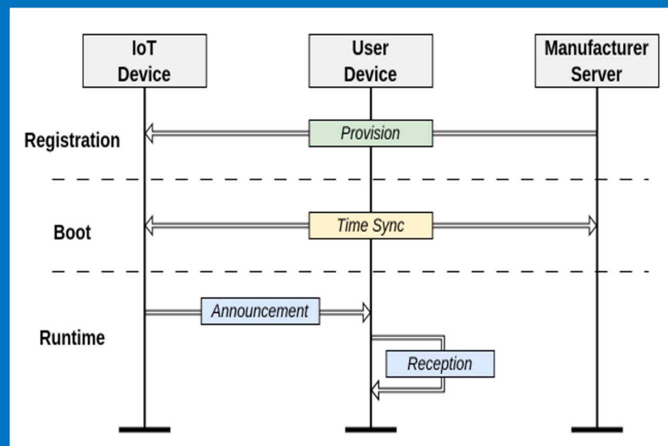
- TEE-equipped IoT device

User Device (U_{dev})

- User's personal trusted device

Manufacturer Server (M_{srv})

- Back-end trusted server
  hosted by device manufacturer
- Third-party servers can be used to ease burden of M_{srv}



17

17

Adversary Model

- IoT Device (I_{dev})
  - Adversary has full control over device memory/storage except:

    TCB = PAISA software + peripheral configuration in **Secure World**
  - Out-of-Scope:
    - jamming, wormhole attacks (partially mitigated), invasive physical, and runtime (e.g., control-flow and data-only) attacks
    - non-compliant devices
- U_{dev} and M_{srv} : Assumed trusted
- Dolev-Yao adversarial model applicable to all communication

18

18

Design Challenges

- DoS attacks from within I_{dev} : Malware in Normal World can simply "squat" on the network interface
  - Configure timer and network peripherals as secure with high(er) priority
- Announcement Size: Device information can be long
  - Minimize message size by placing all information into a (shortened) URL
  - Announcement carries only: (i) URL, (ii) timestamp, and (iii) signature

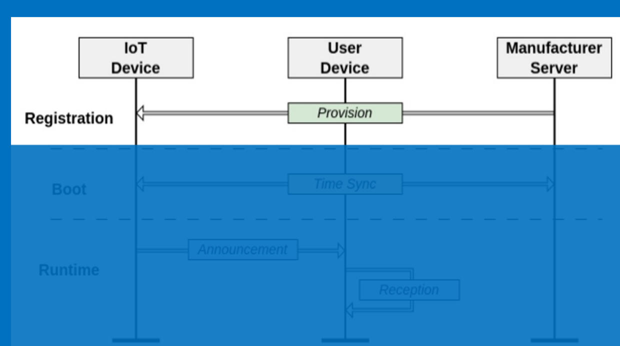
19

19

Registration/Provisioning Phase

Provisioning

- Takes place when I_{dev} is manufactured
- Assumed to be secure
- M_{srv} creates *Device Manifest* of I_{dev} and stores it in URL_{Man}
- M_{srv} installs:
  - Normal World: normal software
  - Secure World: PAISA software, M_{srv} public key, URL_{Man}, and metadata



20

20

Registration Phase

Device Manifest may include

- Device ID
- Current status
- Device type/model
- Sensors/actuators
- Manufacturer
- Provisioning date/location
- Network interfaces
- Location of deployment
- Certificates
- ...


```
{
  "id": "1892",
  "status": "active",
  "type": "thermostat",
  "manufacturer": "Google",
  "sensor": [ "temperature", "humidity",
    "proximity", "motion", "light" ],
  "actuator": [ "air conditioning" ],
  "purpose": [
    "control temperature and
    humidity indoors",
    "actuate light on motion"
  ],
  "network": "WiFi",
  "location": "Chichago, IL, USA",
  "certificates": [
    "device_cert",
    "manufacturer_cert"
  ]
}
```

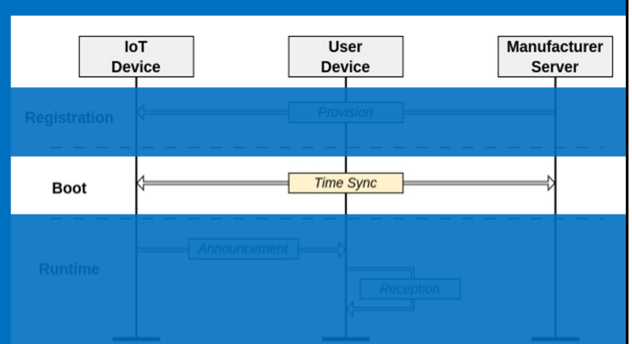
```
{
  "id": "237834",
  "status": "active",
  "type": "surveillance camera",
  "manufacturer": "Blink",
  "sensor": [ "microphone", "motion",
          "camera" ],
  "actuator": [ "speaker" ],
  "purpose": [
    "detect objects and humans",
    "observe human behavior and report
    unexpected behavior"
  ],
  "network": "WiFi",
  "location": "Los Angeles, CA, USA",
  "certificates": [
    "device_cert",
    "manufacturer_cert"
  ]
}
```


21

21

Boot Phase

- Time Sync
  - Synched timestamp needed to ensure *Timeliness* property
  - At every boot, I_{dev} synchs time with M_{srv} using 3-way signature-based protocol (a DoS potential)
  - Afterwards, I_{dev} maintains current time using its local timer (controlled by TCB)



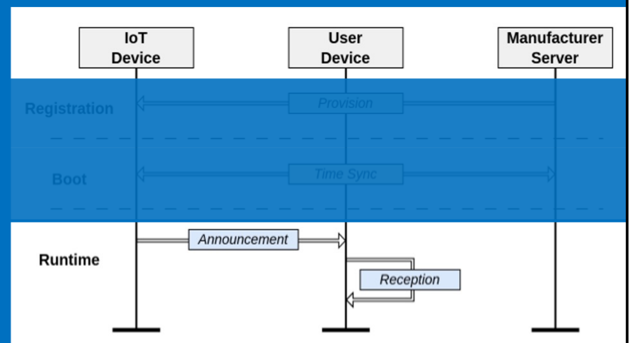
22

22

Runtime Phase

Announcements:

- At fixed intervals, I_{dev} generates and broadcasts an announcement containing: nonce, current time, URL_{Man}, attestation report, and signature
- All program memory in Normal World is attested and compared with stored hash value
- Attestation interval can differ from Announcement interval



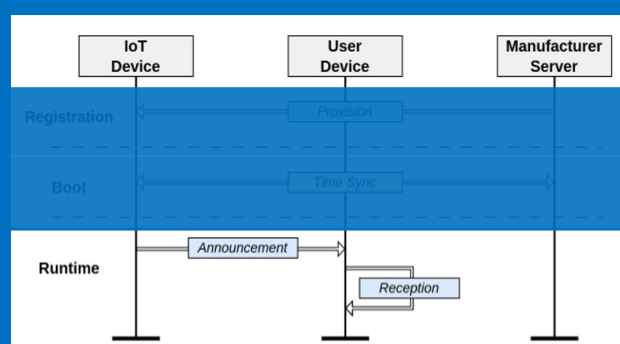
23

23

Runtime Phase

Reception:

- U_{dev} checks if I_{dev} time is acceptable
- Fetches Device Manifest from URL_{Man} and verifies signatures
- Checks attestation report to determine if I_{dev} is trustworthy; if not, alerts the user



24

24

Implementation: Setup

IoT Device

- NXP LPC55S69-EVK → main board running PAISA software
- ESP32-C3-DevKitC-02 → network interface connected to main board via UART

User Device

- Google Pixel 6

Manufacturer Server

- Desktop with Intel i5-11400 processor (Ubuntu 20.04 LTS)

25

25

Implementation: Challenges

Broadcast without Established Channels

- IEEE 802.11 WiFi Beacon frames, typically used by routers to advertise presence
- Announcement can be populated with *vendor-specific* elements in Beacon frames

Crypto Overhead

- Use cryptographic accelerator on the main NXP board (CASPER) for Elliptic Curve (EC-DSA) crypto operations

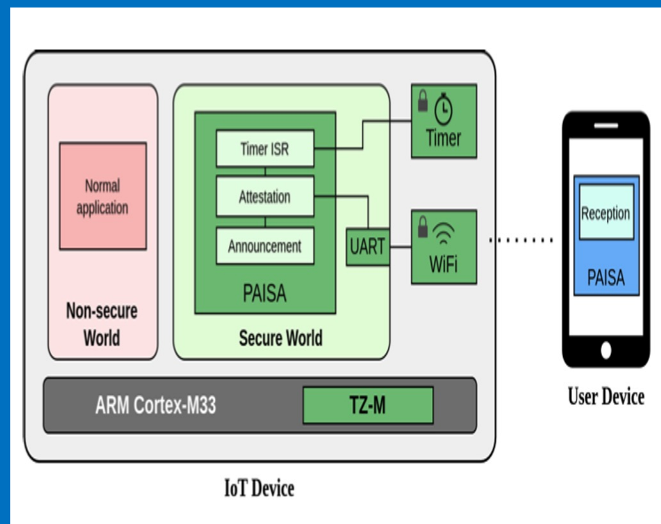
26

26

Implementation: IoT Device

PAISA Software on Main board

- *Timer Interrupt Service Routine* (ISR): Given highest priority, triggered at fixed intervals. Executes *Attestation* and *Announcement*
- *Attestation:* Computes SHA256 over program memory in Normal World
- *Announcement:* Generates announcement and hands over to network interface via UART



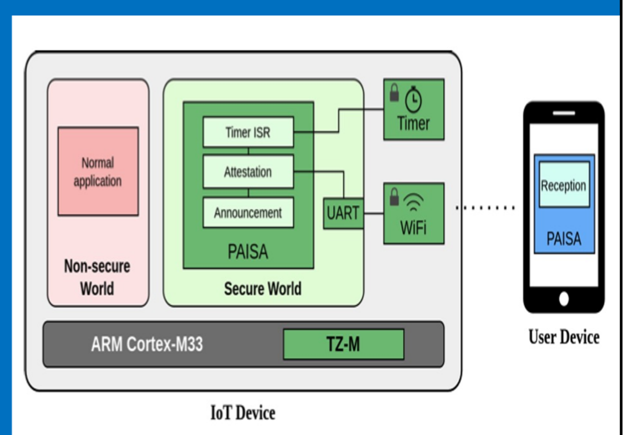
27

27

Implementation: IoT Device

Normal World software on main board

- Thermal sensor application: reads temperature values from sensor and sends to server via *UART*
- Non-Secure Callable function (NSC): since *UART* is exclusive to Secure World, we implemented *UART* NSC function in Secure World, which can be invoked by thermal sensor application



28

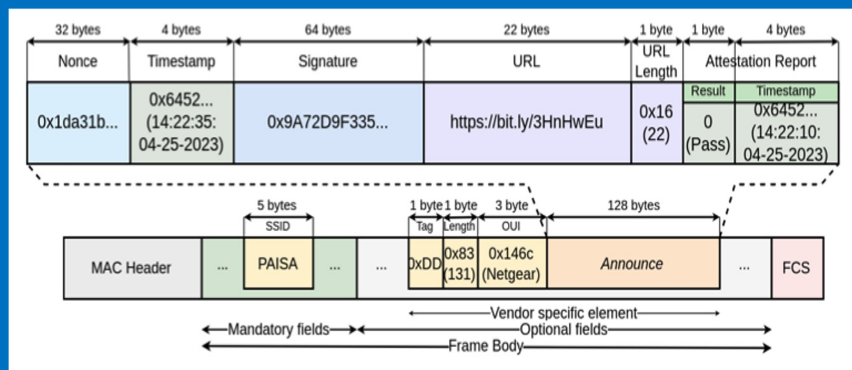
28

Implementation: IoT Device

Network Stack in Network Interface

- Boot time with M_{srv}: uses Station mode for *TimeSync* using UDP
- Runtime with U_{dev}: uses SoftAP mode for *Announcement*

Announce Message:



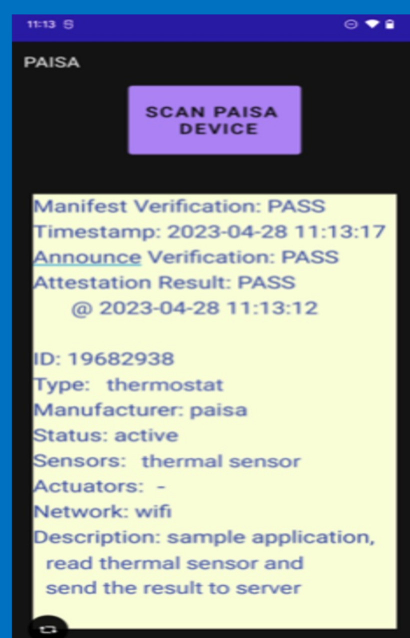
29

29

Implementation: User Device

Reception App

- Scans for, and identifies, packets with WiFi beacon frame containing *SSID="PAISA"*
- Parse the message and retrieve Device Manifest from URL_{Man}: uses *AsyncTask* for threading
- Verifies signatures and informs user



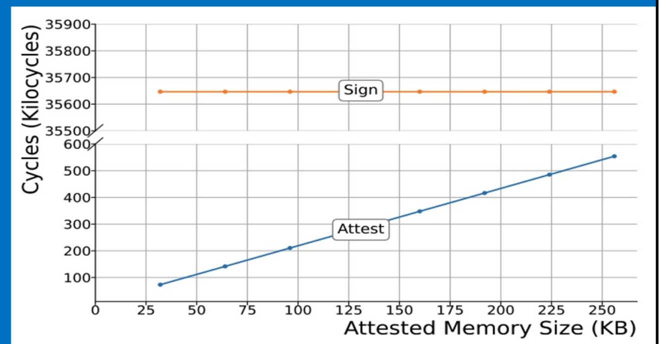
30

30

Evaluation

Overhead on I_{dev}

- Signing (EC-DSA): ≈230ms @ 150MHz, while RA – ≈4ms for 250KB @ 150MHz
- Initializing device driver (at boot): 9.66ms
- *TimeSync* (at boot): ca. 1sec
  Reasonable since device boots infrequently
- Announcement (at runtime): 236.21ms, of which 230ms due to signing latency



PAISA Procedure	Cycles		Time @ 150MHz (ms)	
	Mean	Standard Deviation	Mean	Standard Deviation
InitDevice	1,449,461	121	9.66	0.02
TimeSync	161,386,850	28,129,473	1,075.91	187.53
Announcement	35,431,478	87,119	236.21	0.58

Table 2: PAISA **Overhead on** I_{dev} **at** *BootTime.*

31

Evaluation

Overhead on M_{srv} and U_{dev}

- Signing/Verification takes 1ms each @ 2.6GHz in M_{srv}
- Reasonable → multiple requests can be served in parallel
- *Reception* takes 1s @ [1.8-2.8]GHz in U_{dev}
- Reasonable → latency is mainly due to network delay

Device	PAISA Procedure	Time (ms)	
		Mean	Standard Deviation
M_{svr} @ 2.6GHz	**TimeSync**	5.60	2.77
U_{dev} @ [1.8-2.8]GHz	**Reception**	1070.34	247.00

Table 3: PAISA **Overhead on** M_{svr} **and** U_{dev}.

32

32

Discussion and Future work

Compatibility with other platforms

- Bare-metal devices with minimal or no security guarantees. No miracles without hardware modifications!
  - *Secure storage* for keys and certificates
  - *Prioritized tasks* must be preemptively executable
  - *Prevention of* security violations

  - GAROTA (Usenix'22) and AWDT (S&P'21) are good examples

33

33

Discussion and Future work

Compatibility with other platforms

- Mid-tier IoT devices
  - ARM Cortex-M23/M35/M55/M85 based MCU: TrustZone-M is available
  - RISC-V based MCU: Physical Memory Protection (PMP) satisfies PAISA security properties

- Higher-end IoT devices
  - ARM Cortex-A series: TrustZone-A is a superset of TrustZone-M

34

34

Discussion and Future work

- Other Network Interfaces
  - Bluetooth: 5.0 has a feature for extended advertising with up to 255 bytes of arbitrary data
    - Already implemented in PAISA!
  - Cellular: illegal to tinker with cellular packets in many countries/jurisdiction. Plus cellular is not really "nearby"
  - Zigbee: usually, there is a hub/controller that receives Zigbee traffic and has a WiFI or Bluetooth interface. Assuming (some) trust, it can resend individual devices' announcements

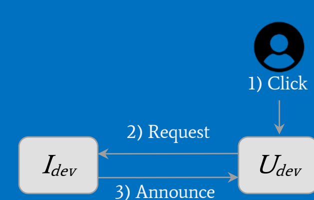
35

35

Discussion and Future work

How to Initiate Announcements

- Push Model: initiated by I_{dev} (PAISA now): broadcasts announcements at regular intervals
  - Power: efficient (switch to sleep mode in idle state)
  - Network: inefficient (announcements clog bw)
  - Needs secure timer interrupt



- Pull Model: initiated by U_{dev}: requests announcements from all nearby I_{dev}-s
  - Power: inefficient (network interface always on)
  - Network: efficient (announcements on-demand)
  - No secure timer interrupt needed
  - BUT… what about attestation?

36

36

Discussion and Future work

Localization: PAISA doesn't localize devices!

- ✓ RSSI: Received Signal Strength Indicator
  - ✓ Available on many network interface including WiFi and Bluetooth
  - ✓ Noise can worsen quality worse, due to, e.g., walls, doors, glass, etc
- ✓ Additional features
  - ✓ WiFi (5m~15m): WiFi fingerprinting, Angle of Arrival (AoA), Time of Flight (ToF)
  - ✓ Bluetooth (1m~3m): AoA, Angle of Departure (AoD) for finding direction (Bluetooth 5.1~)
  - ✓ Ultra Wideband (UWB) (10cm~30cm): AoA, ToF

37

37

Summary

- Devices are increasingly surrounding us in many spheres of everyday life
- Many of them are not easily perceivable by us
- Need a way to make users aware of their presence and capabilities
  - This is CLEARLY not applicable to all devices!
- Detection- and registration-based approaches work, partially
- We propose a compliance-based device-architectural approach
  - Complementary
- PAISA is not a panaceas, just the first step – a feasibility demo
- Much work remains to be done

38

38

Thank you!

questions?

39

39