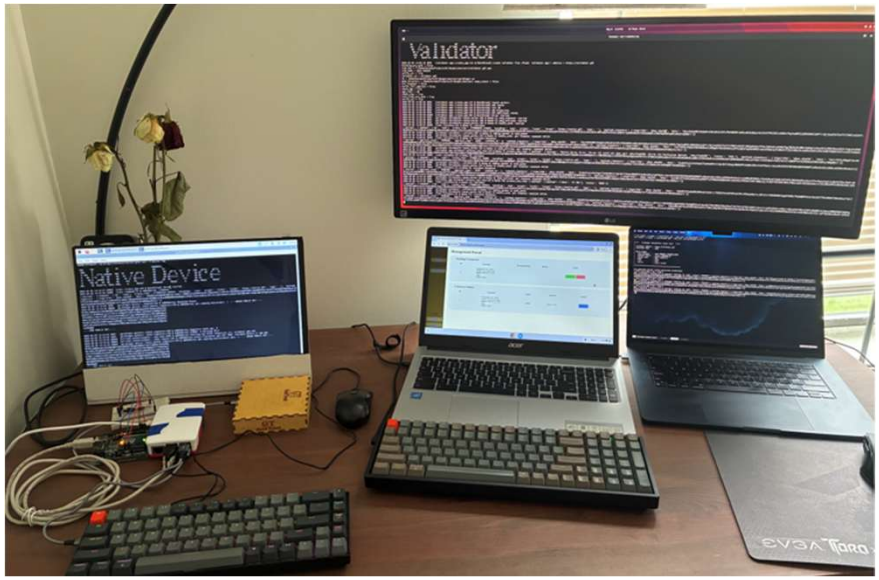
Securing Two-Party Authentication of SCADA Commands in the Field using PUF-Encrypted Public-Key Signing

Kareem Ahmad∗, Arman Allahverdi∗, **Vincent John Mooney III∗†‡** and Santiago Grijalva∗‡
∗School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia, USA
†School of Computer Science, Georgia Institute of Technology, Atlanta, Georgia, USA
‡School of Cyber-Security and Privacy, Georgia Institute of Technology, Atlanta, Georgia, USA

Georgia Tech

Acknowledgement

This work has been partially supported by the U.S. Department of Energy (DoE) Office of Cybersecurity, Energy Security, and Emergency Response (CESER) under Cybersecurity for Energy Delivery Systems (CEDS) Agreement Number #DE-CR0000055 to the Georgia Tech Research Corporation: GridLogic: Hardware/Software Codesign for Deep Grid Visibility and Security



Georgia Tech

Outline

1. Problem Statement
2. Background
    1. Definitions
    2. System Architecture
    3. Attack Surface
    4. TPA Protocol
3. Methodology
    1. Signing Flow
    2. Device Enrollment
    3. Experimental System
4. Results
5. Discussion
6. References

Outline

1. **Problem Statement**
2. Background
    1. Definitions
    2. System Architecture
    3. Attack Surface
    4. TPA Protocol
3. Methodology
    1. Signing Flow
    2. Device Enrollment
    3. Experimental System
4. Results
5. Discussion
6. References

Problem Statement

- Cybersecurity in modern power utilities is becoming increasingly important as
  - April 7, 2026: U.S.A. CISA Advisory regarding active exploitation of PLCs in Critical Infrastructure [1]
  - July 22, 2026: Advisory updated to include PLCs made by three separate manufacturers under attack. Attackers disabled critical warnings, manipulated SCADA displays, and in at least one case acquired a remote ssh shell. [1]
- Utilities often leave communication networks unencrypted
- Man-in-the-middle packet manipulation can result in the execution of malicious commands

[1] CISA AA26-097A

Problem Statement

- GridLogic Two-Party Authentication (TPA) Protocol enables validating commands before execution

Problem Statement

- GridLogic Two-Party Authentication (TPA) Protocol enables validating commands before execution
  - One party may be an entry-level employee

Problem Statement

- GridLogic Two-Party Authentication (TPA) Protocol enables validating commands before execution
  - One party may be an entry-level employee
  - The second party is a senior executive

Problem Statement

- GridLogic Two-Party Authentication (TPA) Protocol enables validating commands before execution
  - One party may be an entry-level employee
  - The second party is a senior executive
- GridLogic TPA previously used symmetric signing keys
- Compromise of a device and leakage of symmetric keys gives attacker the ability to sign their own packets

- **Goal #1:** Use a public-private signing algorithm in GridLogic TPA to limit key exposure
- **Goal #2:** Encrypt private signing keys on edge devices using a Physically Unclonable Function (PUF) [2] to protect against device compromise

Outline

1. Problem Statement
2. **Background**
    1. Definitions
    2. System Architecture
    3. Attack Surface
    4. TPA Protocol
3. Methodology
    1. Signing Flow
    2. Device Enrollment
    3. Experimental System
4. Results
5. Discussion
6. References

Definitions

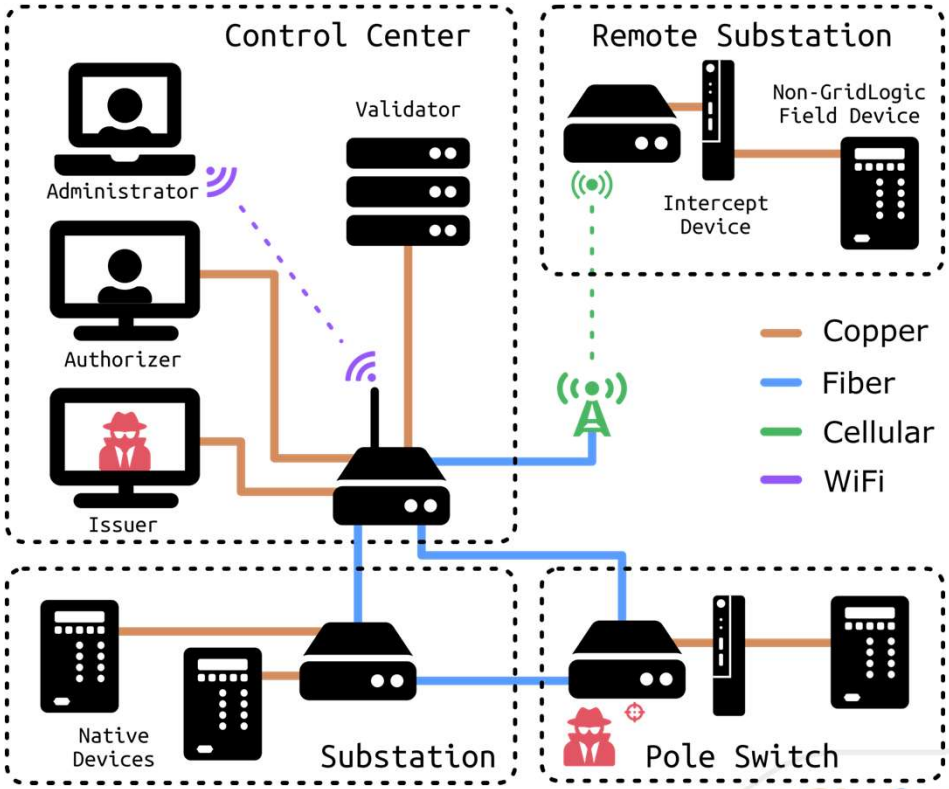
- **Two Party Authentication (TPA):** A variant of two-factor authentication that requires two parties, typically two separate individual humans, to authenticate an action for it to occur
- **Certificate Authority (CA):** A trusted entity that signs public keys and affirms the entity the public key belongs to.
- **Physically Unclonable Function (PUF):** Hardware that expresses “an inherent and unclonable instance-specific feature of a physical object” [2]. These features can be used generate a reproducible secret key that exists outside Non-Volatile Memory and only after powering up a microchip (for a silicon-based PUF)
- **Initialization Vector (IV):** Non-secret value used to prevent codebook attacks in block ciphers.

Definitions

- **Privacy-Enhanced Mail (PEM):** A file format defined in RFC 7468 of the IETF commonly used for storing certificates and public/private keys
- **Rivest-Shamir-Adelmen (RSA):** A family of public-key cryptosystems that use modular exponentiation and large prime numbers
- **Elliptic Curve Digital Signature Algorithm (ECDSA):** A family of public-key digital signature algorithms that use elliptic curve cryptography

System Architecture

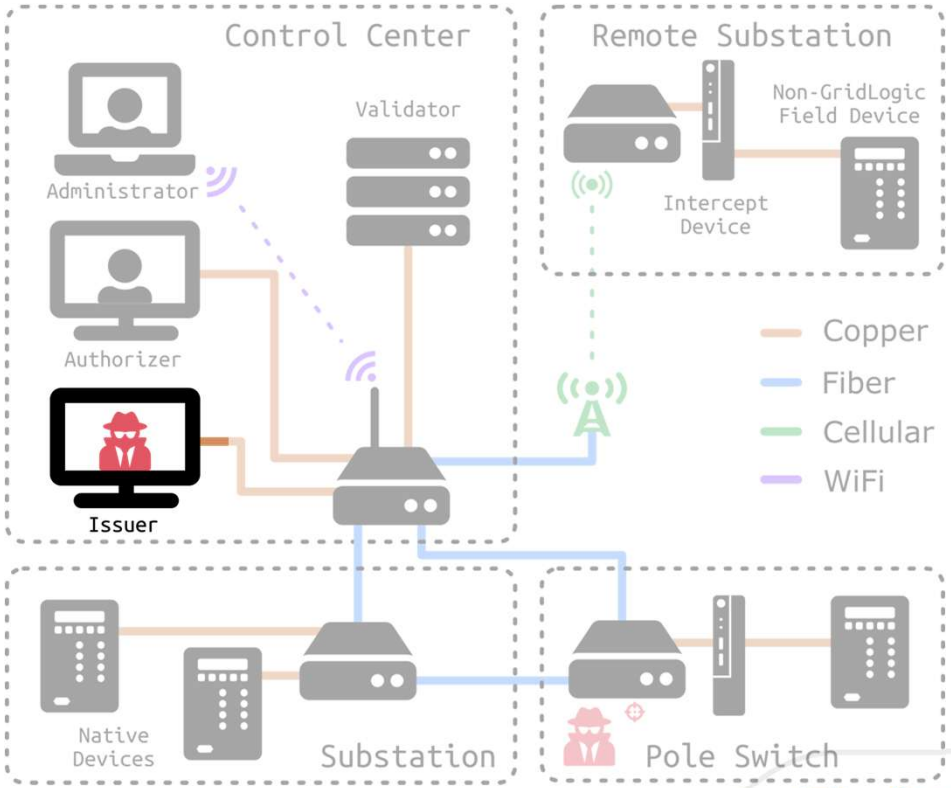
- **Validator:** Computational entity that signs packets and coordinates TPA between Issuers, Authorizers and GridLogic Devices
- **Issuer:** A potentially untrusted entity that creates and sends commands
- **Authorizer:** A trusted entity that reviews TPA requests and approves or rejects them
- **Administrator:** The highest level of trusted entity that enrolls or unenrolls Authorizers, Issuers or Devices
- **GridLogic Native Device:** A field device that receives SCADA commands using the GridLogic TPA protocol and performs actions based on the command received



Georgia Tech

Attack Surface

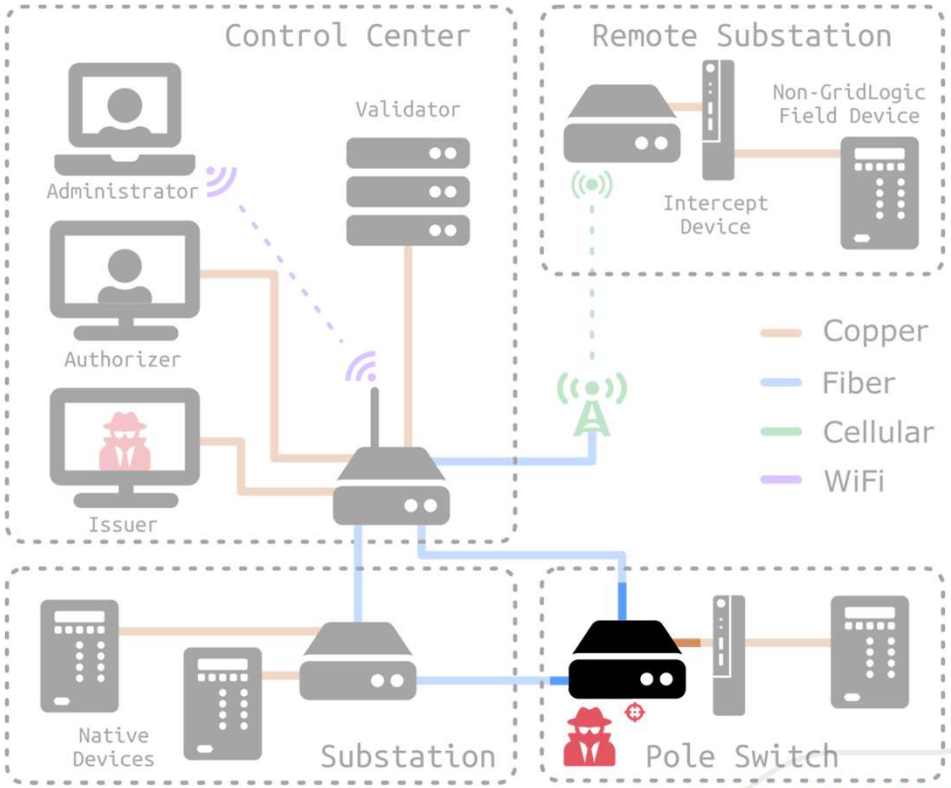
- **Control Center Attacker**: Low-level engineer with access to Issuer-level signing keys attacking the system from inside the control center.
- **Field Station Attacker**: Low-level engineer with access to Issuer-level signing keys attacking the system from network access in the field



Georgia Tech

Attack Surface

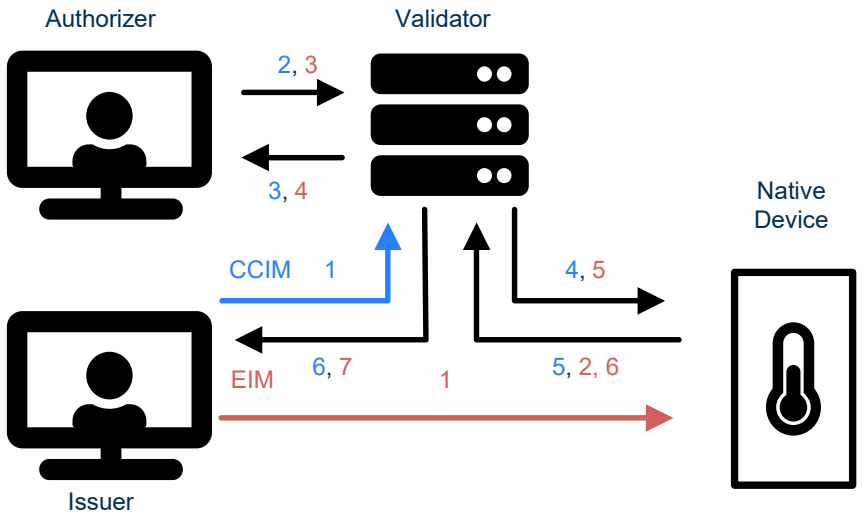
- **Control Center Attacker**: Low-level engineer with access to Issuer-level signing keys attacking the system from inside the control center.
- **Field Station Attacker**: Low-level engineer with access to Issuer-level signing keys attacking the system from network access in the field



Georgia Tech

GridLogic Two-Party Authentication (TPA) Protocol

- Protocol is built on top of HTTP(S) using Representational State Transfer (REST) [3] APIs served on the Validator and each Native Device.
- Control Center Intercept Mode (CCIM)
  - Command Packet is sent by Issuer to Validator first. Validator does the TPA flow before sending a signed Packet to the GridLogic Device.
- Edge Intercept Mode (EIM)
  - Packet is sent by Issuer to GridLogic Device directly. Device reaches out to Validator which does the TPA flow before sending a response back to the GridLogic Device.

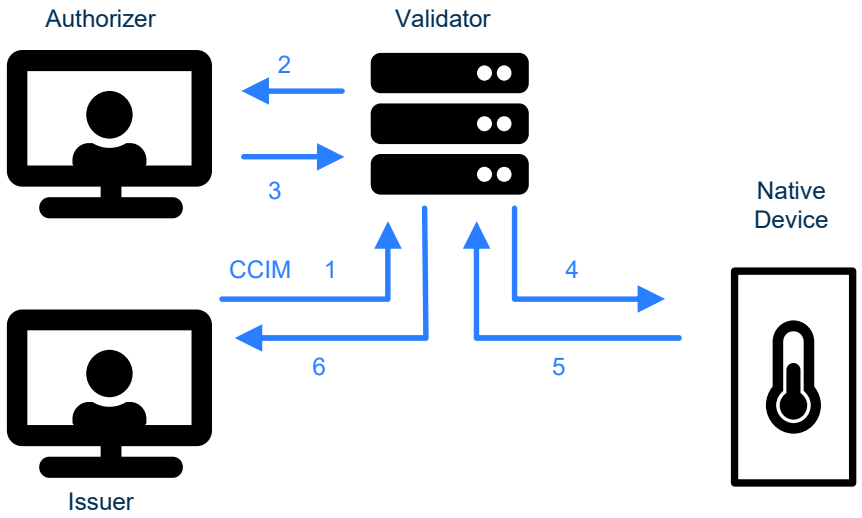


*All packets are signed by their sender.

Georgia Tech

GridLogic Two-Party Authentication (TPA) Flow

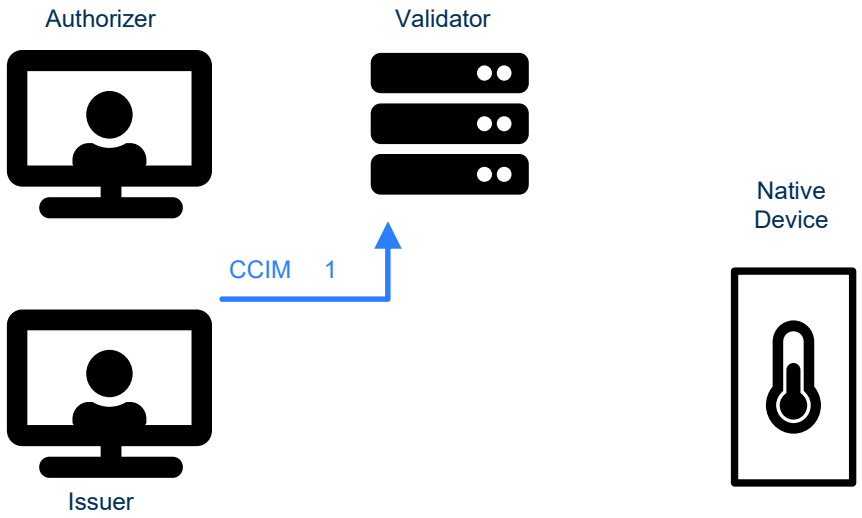
- Protocol is built on top of HTTP(S) using Representational State Transfer (REST) [3] APIs served on the Validator and each Native Device.
- Control Center Intercept Mode (CCIM)
  - Command Packet is sent by Issuer to Validator first. Validator does the TPA flow before sending a signed Packet to the GridLogic Device.
- Edge Intercept Mode (EIM)
  - Packet is sent by Issuer to GridLogic Device directly. Device reaches out to Validator which does the TPA flow before sending a response back to the GridLogic Device.



Georgia Tech

GridLogic TPA Flow: CCIM

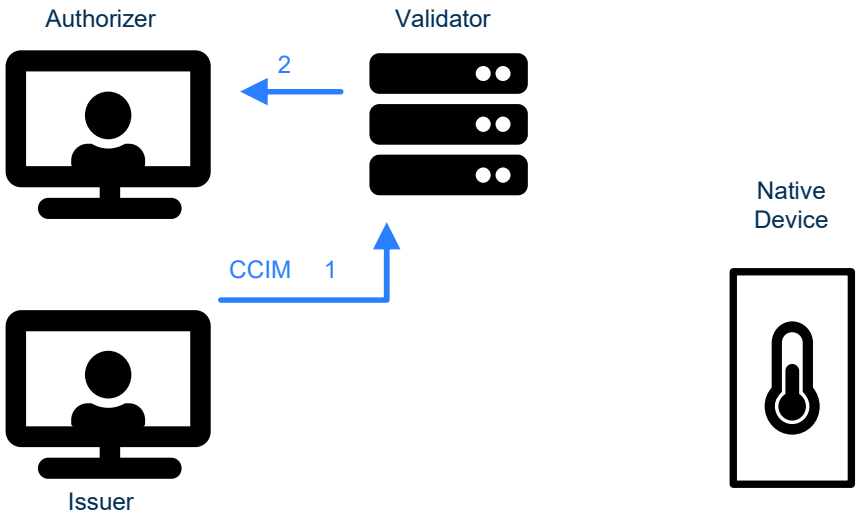
- Protocol is built on top of HTTP(S) using Representational State Transfer (REST) [3] APIs served on the Validator and each Native Device.
- Control Center Intercept Mode (CCIM)
  - Command Packet is sent by Issuer to Validator first. Validator does the TPA flow before sending a signed Packet to the GridLogic Device.
- Edge Intercept Mode (EIM)
  - Packet is sent by Issuer to GridLogic Device directly. Device reaches out to Validator which does the TPA flow before sending a response back to the GridLogic Device.



Georgia Tech

GridLogic TPA Flow: CCIM

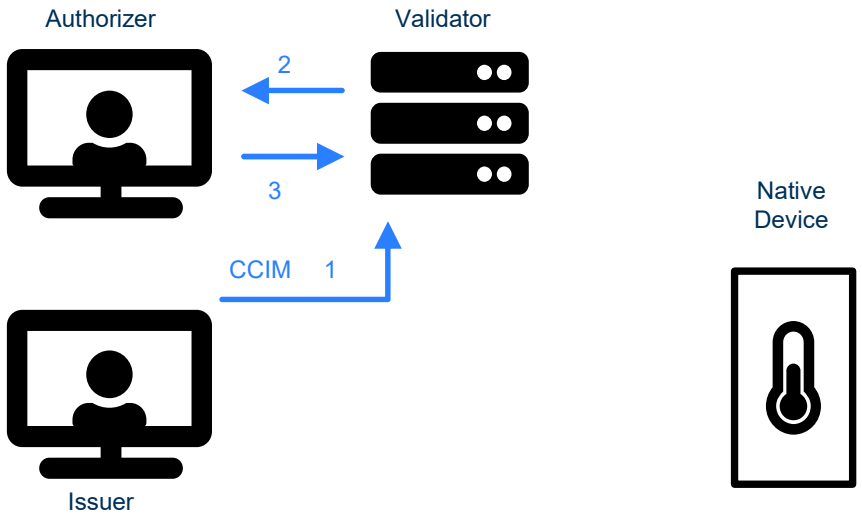
- Protocol is built on top of HTTP(S) using Representational State Transfer (REST) [3] APIs served on the Validator and each Native Device.
- Control Center Intercept Mode (CCIM)
  - Command Packet is sent by Issuer to Validator first. Validator does the TPA flow before sending a signed Packet to the GridLogic Device.
- Edge Intercept Mode (EIM)
  - Packet is sent by Issuer to GridLogic Device directly. Device reaches out to Validator which does the TPA flow before sending a response back to the GridLogic Device.



Georgia Tech

GridLogic TPA Flow: CCIM

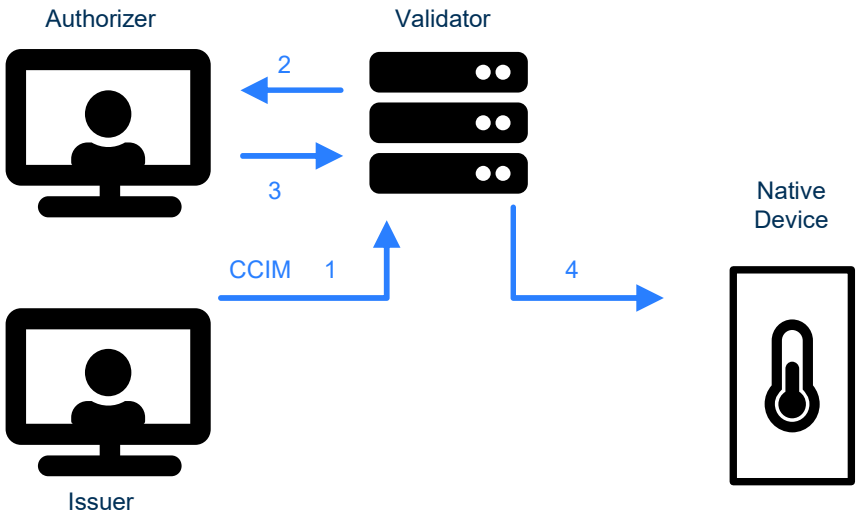
- Protocol is built on top of HTTP(S) using Representational State Transfer (REST) [3] APIs served on the Validator and each Native Device.
- Control Center Intercept Mode (CCIM)
  - Command Packet is sent by Issuer to Validator first. Validator does the TPA flow before sending a signed Packet to the GridLogic Device.
- Edge Intercept Mode (EIM)
  - Packet is sent by Issuer to GridLogic Device directly. Device reaches out to Validator which does the TPA flow before sending a response back to the GridLogic Device.



Georgia Tech

GridLogic TPA Flow: CCIM

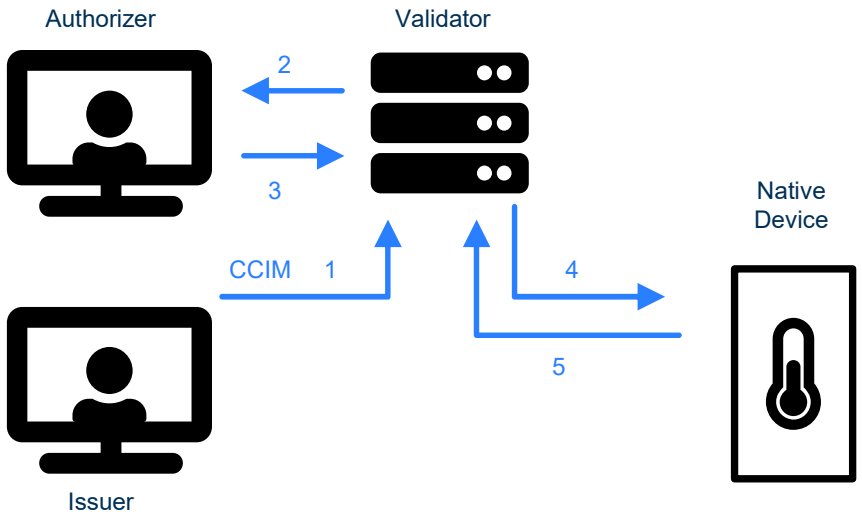
- Protocol is built on top of HTTP(S) using Representational State Transfer (REST) [3] APIs served on the Validator and each Native Device.
- Control Center Intercept Mode (CCIM)
  - Command Packet is sent by Issuer to Validator first. Validator does the TPA flow before sending a signed Packet to the GridLogic Device.
- Edge Intercept Mode (EIM)
  - Packet is sent by Issuer to GridLogic Device directly. Device reaches out to Validator which does the TPA flow before sending a response back to the GridLogic Device.



Georgia Tech

GridLogic TPA Flow: CCIM

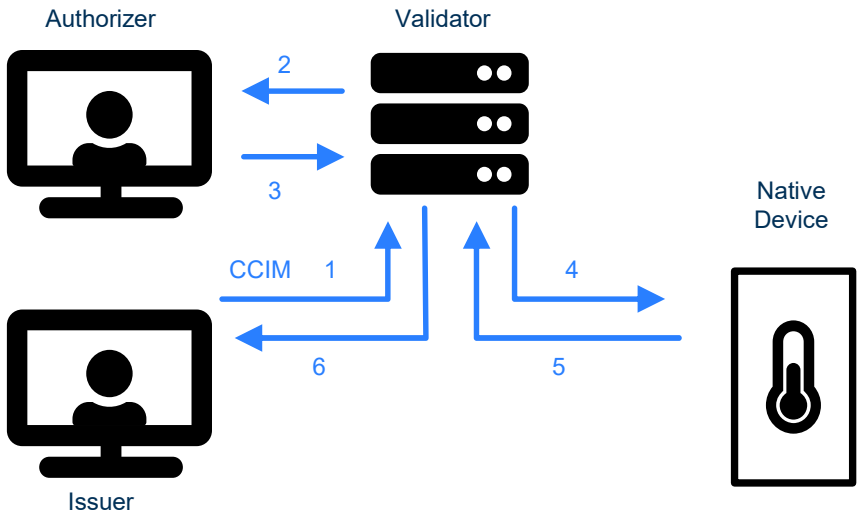
- Protocol is built on top of HTTP(S) using Representational State Transfer (REST) [3] APIs served on the Validator and each Native Device.
- Control Center Intercept Mode (CCIM)
  - Command Packet is sent by Issuer to Validator first. Validator does the TPA flow before sending a signed Packet to the GridLogic Device.
- Edge Intercept Mode (EIM)
  - Packet is sent by Issuer to GridLogic Device directly. Device reaches out to Validator which does the TPA flow before sending a response back to the GridLogic Device.



Georgia Tech

GridLogic TPA Flow: CCIM

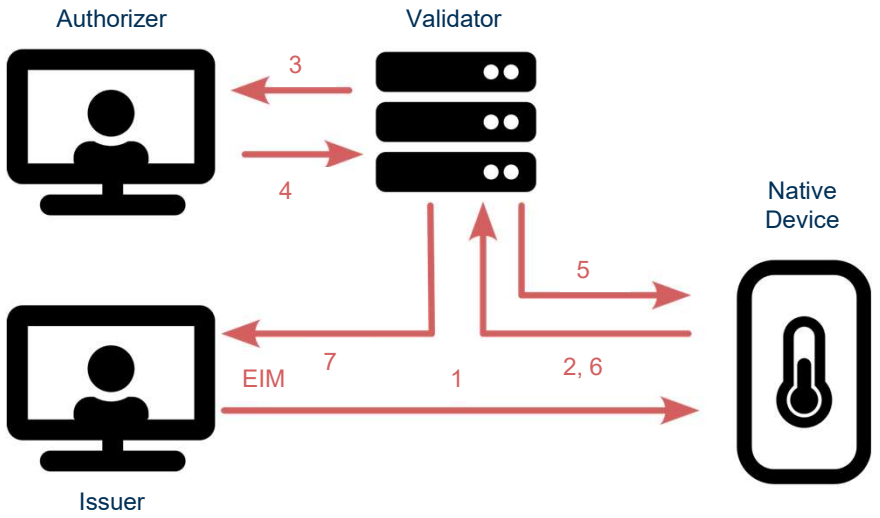
- Protocol is built on top of HTTP(S) using Representational State Transfer (REST) [3] APIs served on the Validator and each Native Device.
- Control Center Intercept Mode (CCIM)
  - Command Packet is sent by Issuer to Validator first. Validator does the TPA flow before sending a signed Packet to the GridLogic Device.
- Edge Intercept Mode (EIM)
  - Packet is sent by Issuer to GridLogic Device directly. Device reaches out to Validator which does the TPA flow before sending a response back to the GridLogic Device.



Georgia Tech

GridLogic Two-Party Authentication (TPA) Flow: EIM

- Protocol is built on top of HTTP(S) using Representational State Transfer (REST) [3] APIs served on the Validator and each Native Device.
- Control Center Intercept Mode (CCIM)
  - Command Packet is sent by Issuer to Validator first. Validator does the TPA flow before sending a signed Packet to the GridLogic Device.
- Edge Intercept Mode (EIM)
  - Packet is sent by Issuer to GridLogic Device directly. Device reaches out to Validator which does the TPA flow before sending a response back to the GridLogic Device.

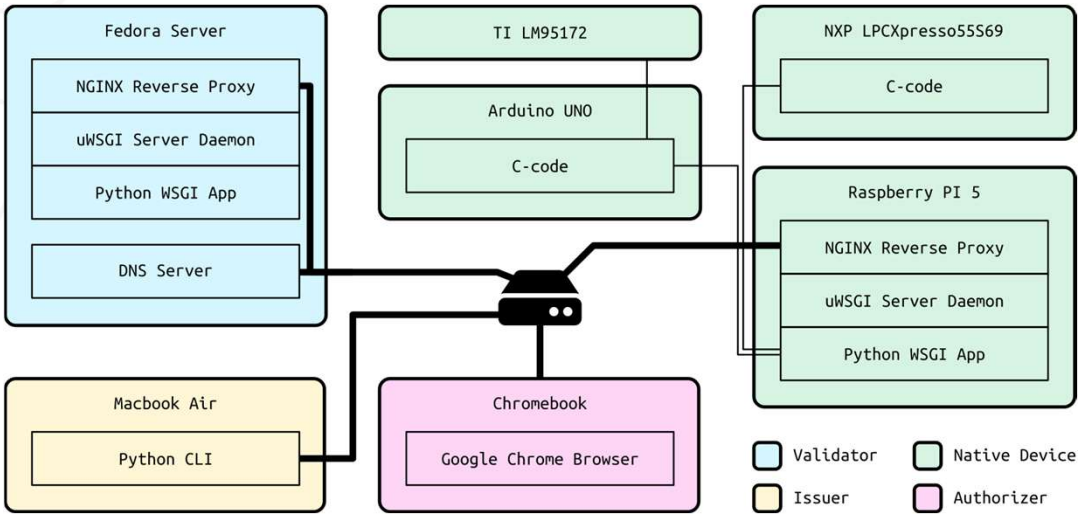


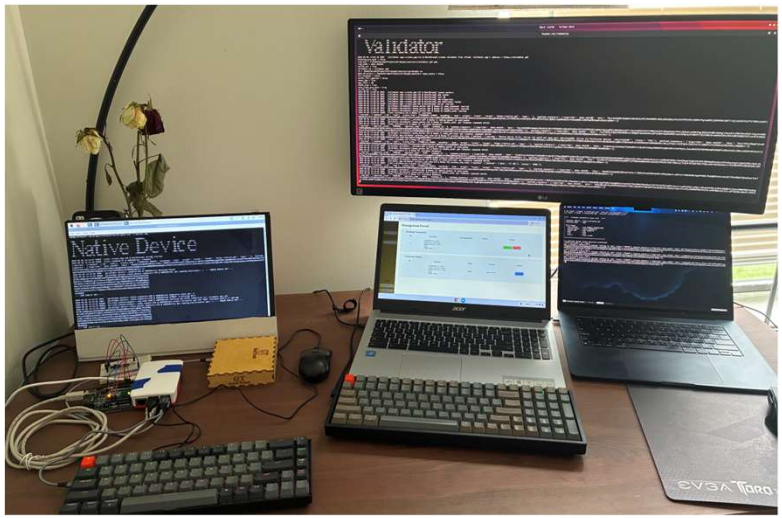
Georgia Tech

Outline

1. Problem Statement
2. Background
    1. Definitions
    2. System Architecture
    3. Attack Surface
    4. TPA Protocol
3. **Methodology**
    1. Experimental System
    2. Signing Flow
    3. Device Enrollment
4. Results
5. Discussion
6. References

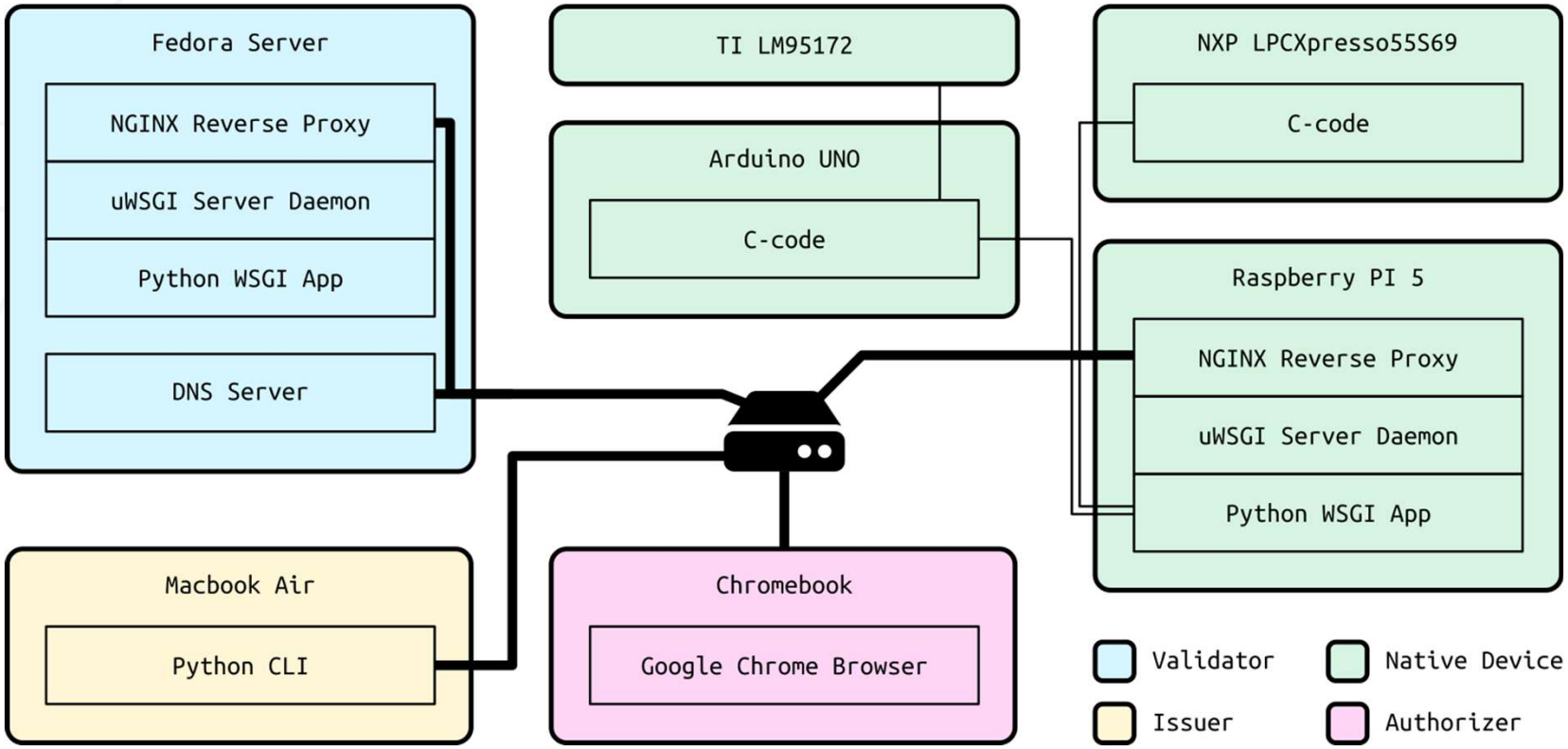
Experimental System





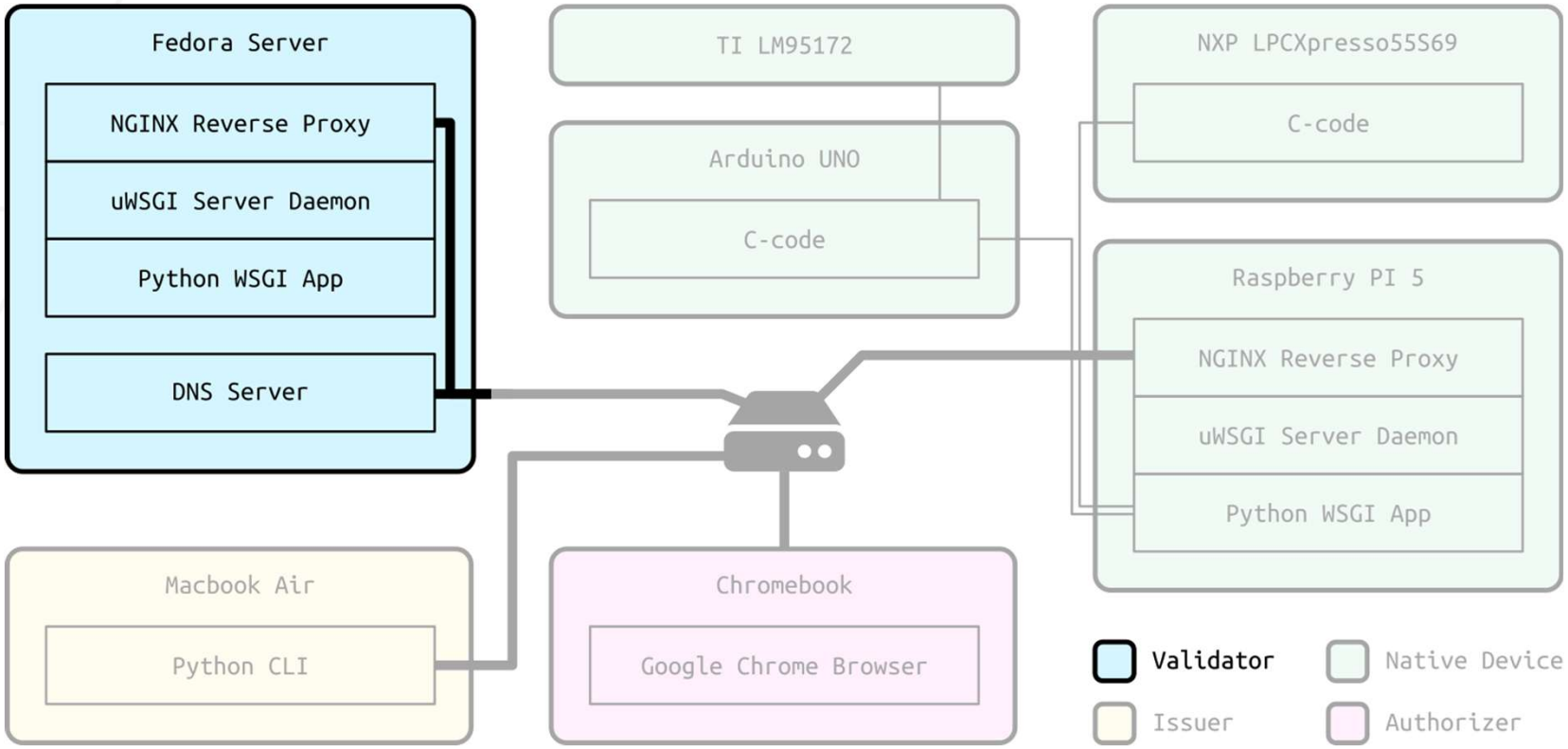
Georgia Tech

Experimental System



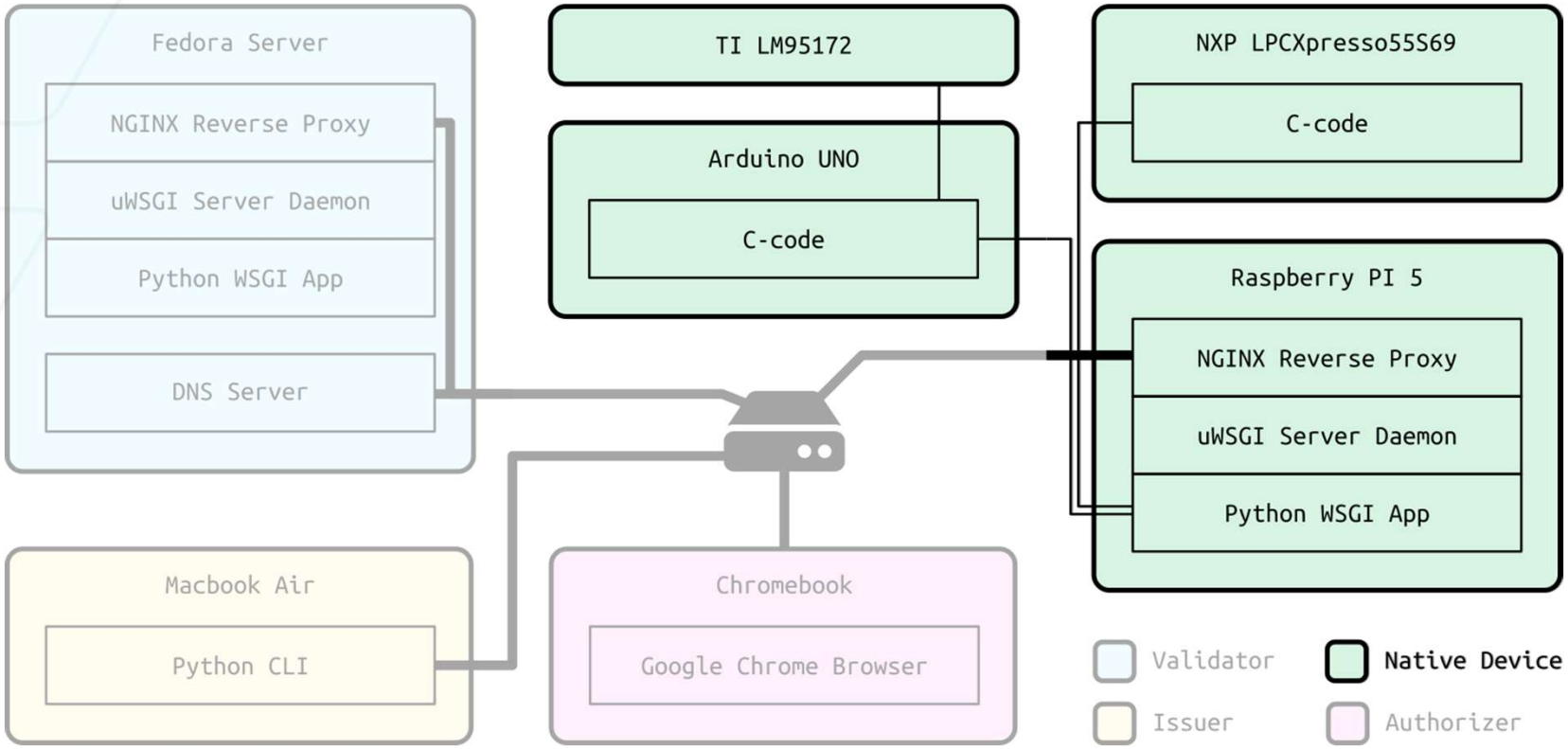
Georgia Tech

Experimental System



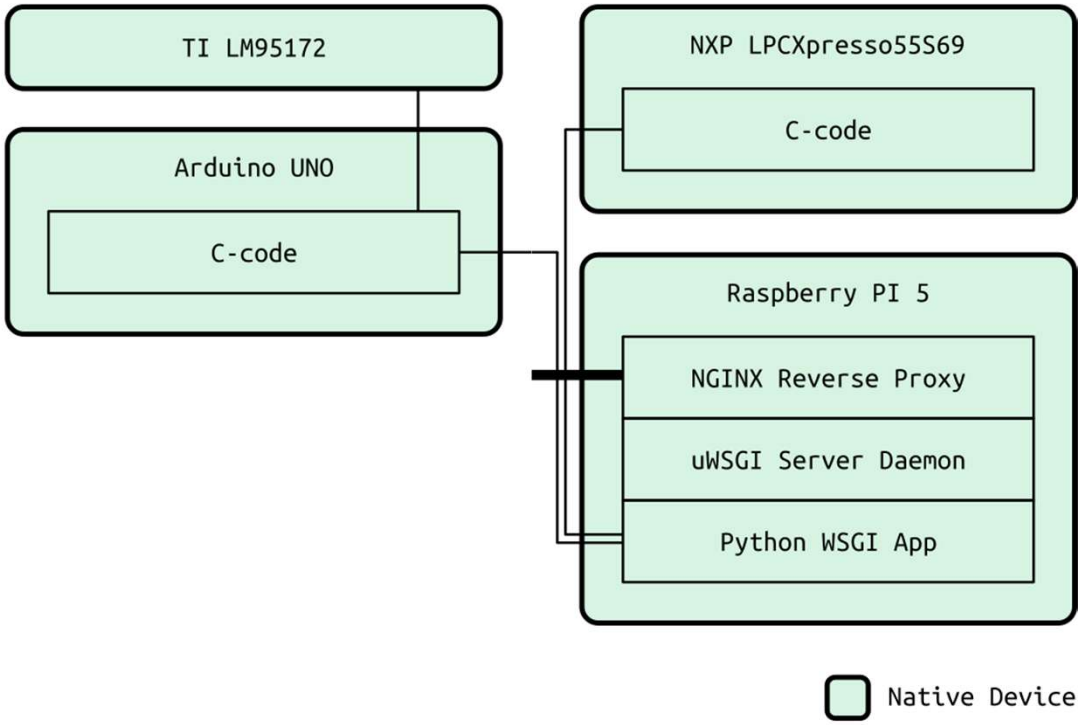
Georgia Tech

Experimental System



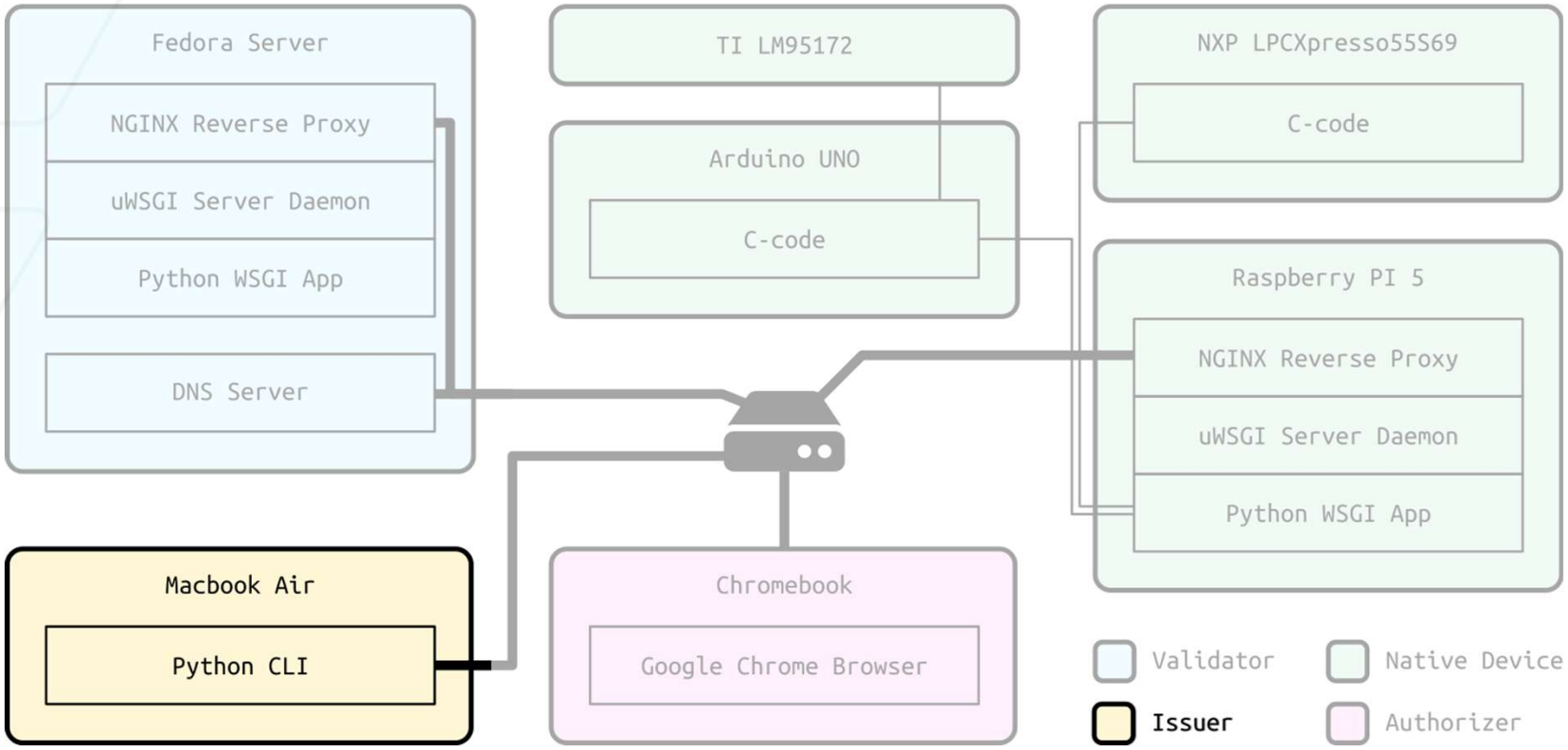
Georgia Tech

Experimental System: Native Device



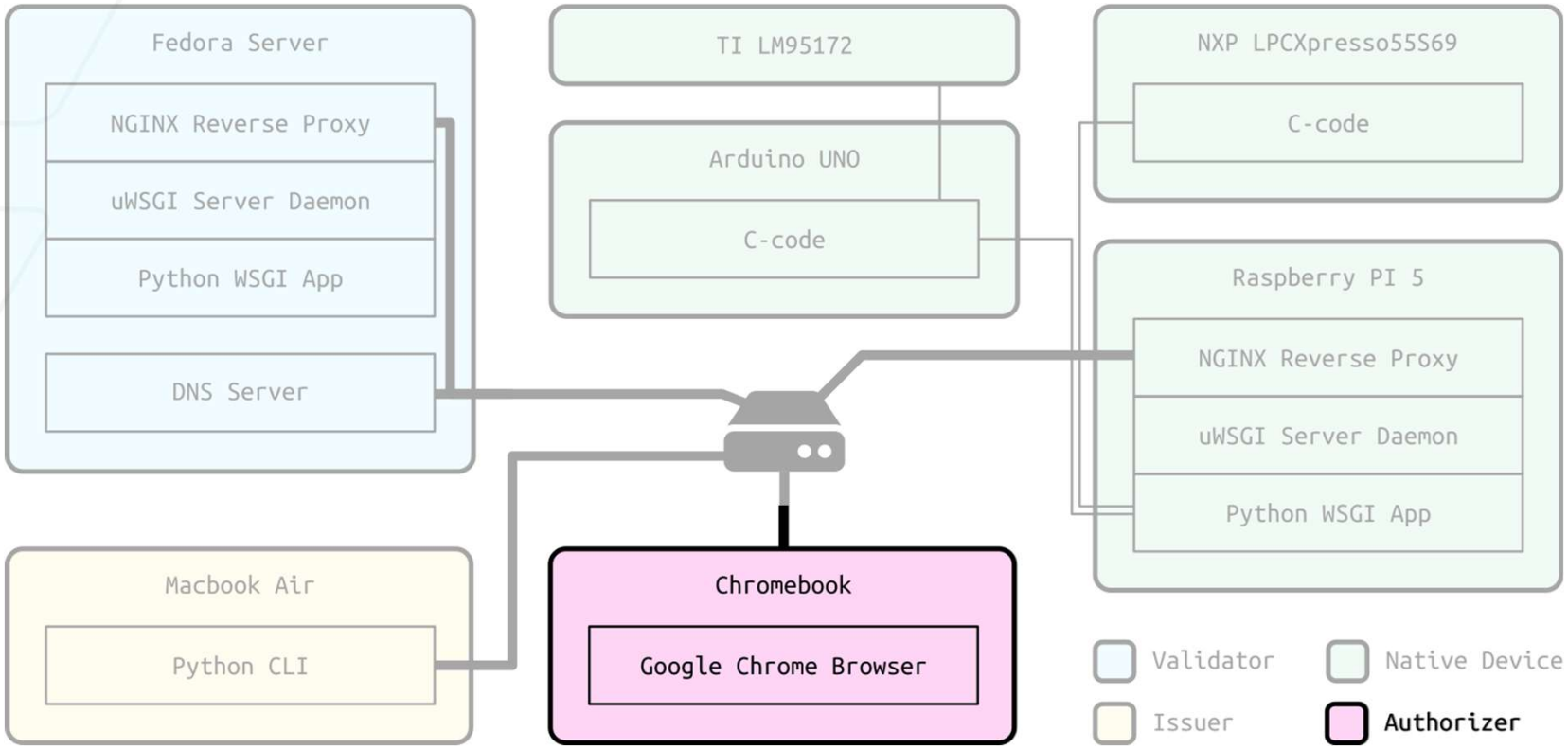
Georgia Tech

Experimental System



Georgia Tech

Experimental System



Georgia Tech

Native Device Enrollment

1. Device is assigned a Hostname and/or IP address
2. Validator Root Certificate is installed in Device

Native Device Enrollment

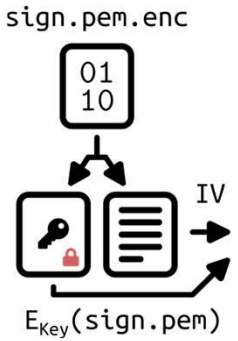
1. Device is assigned a Hostname and/or IP address
2. Validator Root Certificate is installed in Device
3. PUF Enrollment
    a. SRAM PUF fingerprint is extracted to produce an Activation Code and generate a root AES key
    b. An AES key for encrypting the private signing key is generated and stored encrypted using the root AES key in nonvolatile memory

Native Device Enrollment

1. Device is assigned a Hostname and/or IP address
2. Validator Root Certificate is installed in Device
3. PUF Enrollment
    a. SRAM PUF fingerprint is extracted to produce an Activation Code and generate a root AES key
    b. An AES key for encrypting the private signing key is generated and stored encrypted using the root AES key in nonvolatile memory
4. Public-Private Signing keypair (RSA or ECDSA) is generated
5. Private Signing Key is stored encrypted using the second AES key from the PUF
6. SSL Keypair (for securing TLS) and Certificate request is generated
7. Administrator signs Certificate request using Validator's CA

Native Device Signing Flow

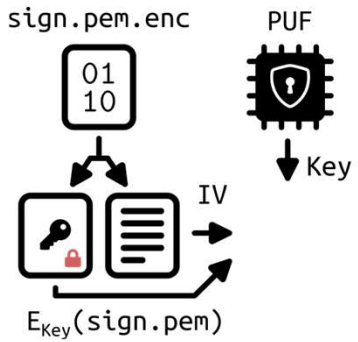
1. The encrypted private signing key and IV are read from memory storage as sign.pem.enc



Georgia Tech

Native Device Signing Flow

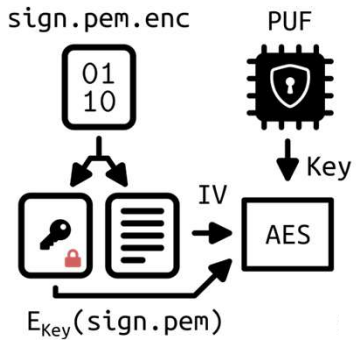
1. The encrypted private signing key and IV are read from memory storage as sign.pem.enc
2. The PUF-derived root AES key



Georgia Tech

Native Device Signing Flow

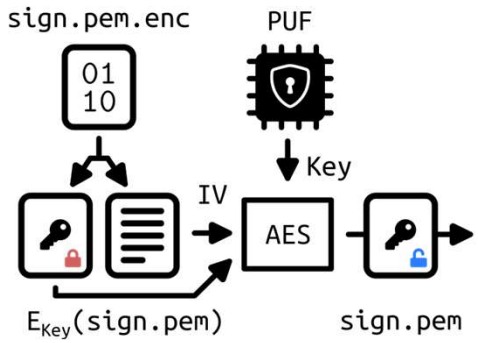
1. The encrypted private signing key and IV are read from memory storage as sign.pem.enc
2. The PUF-derived root AES key is used to decrypt the AES key stored in non-volatile memory on demand



Georgia Tech

Native Device Signing Flow

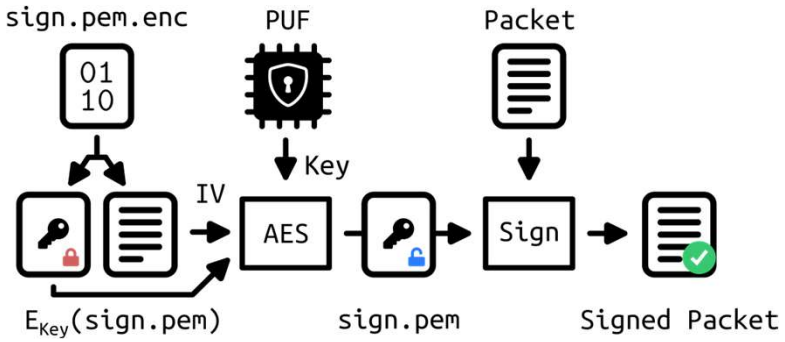
1. The encrypted private signing key and IV are read from memory storage as sign.pem.enc
2. The PUF-derived root AES key is used to decrypt the AES key stored in non-volatile memory on demand
3. The decrypted AES key is used to decrypt the private signing key into volatile memory
4. The decrypted private signing key



Georgia Tech

Native Device Signing Flow

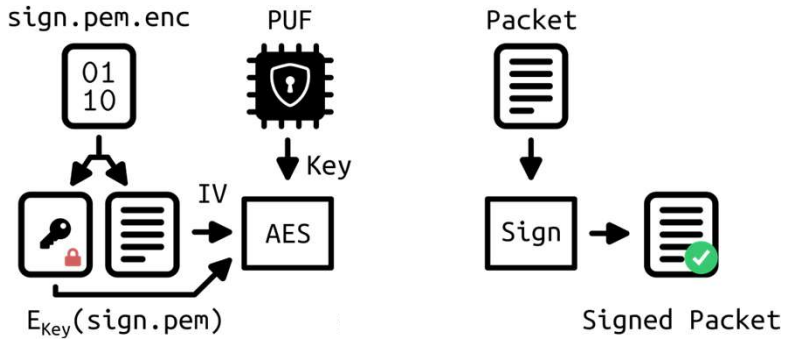
1. The encrypted private signing key and IV are read from memory storage as sign.pem.enc
2. The PUF-derived root AES key is used to decrypt the AES key stored in non-volatile memory on demand
3. The decrypted AES key is used to decrypt the private signing key into volatile memory
4. The decrypted private signing key is used to sign a packet



Georgia Tech

Native Device Signing Flow

1. The encrypted private signing key and IV are read from memory storage as sign.pem.enc
2. The PUF-derived root AES key is used to decrypt the AES key stored in non-volatile memory on demand
3. The decrypted AES key is used to decrypt the private signing key into volatile memory
4. The decrypted private signing key is used to sign a packet
5. The decrypted private signing key is erased from volatile memory



Georgia Tech

Outline

1. Problem Statement
2. Background
    1. Definitions
    2. System Architecture
    3. Attack Surface
    4. TPA Protocol
3. Methodology
    1. Signing Flow
    2. Device Enrollment
    3. Experimental System
4. **Results**
5. Discussion
6. References

Experiment

- For each scenario 50 commands are sent spaced 2 seconds apart
- Roundtrip time is measured from when the issuer creates the command packet to when the issuer receives the final response
- Scenarios are a cross of all the following
  - CCIM vs EIM
  - ECDSA vs RSA
  - Wired (1Gbps) vs WiFi (5GHz)
  - PUF-Encrypted Public Key Signing (PUF) vs Public Key Signing (Sign) vs No Signing (TPA) vs No TPA (Base)

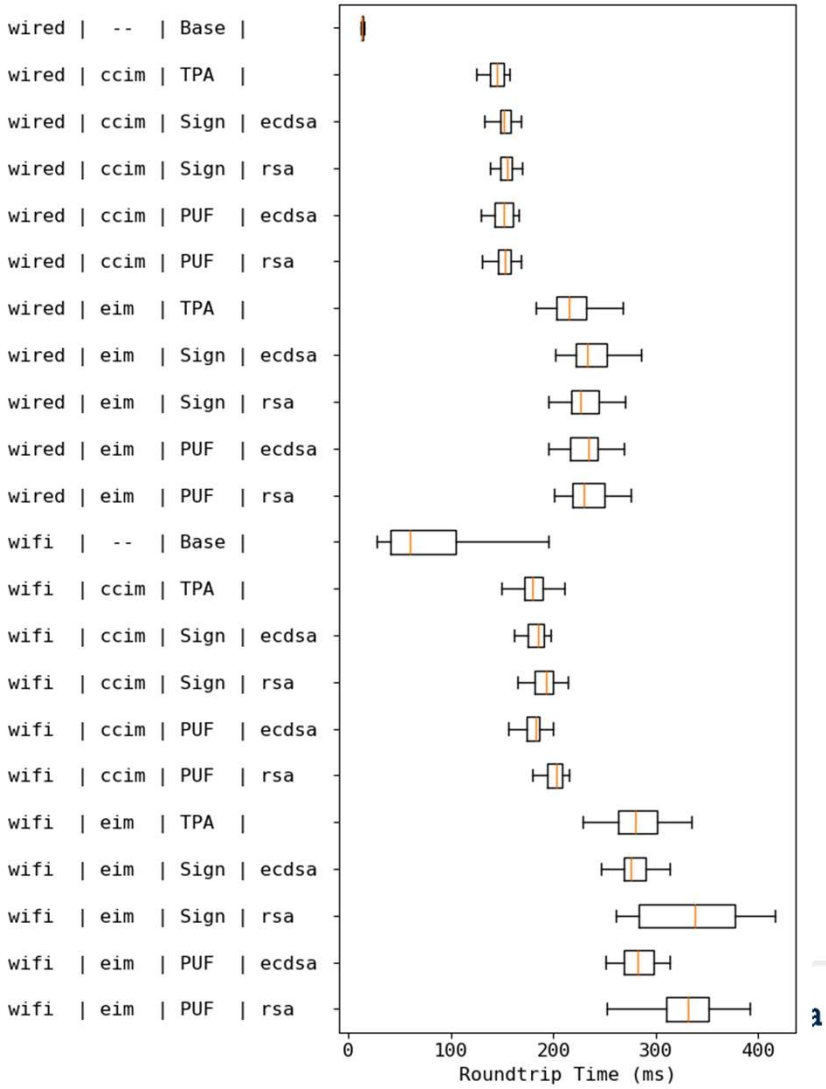
Experimental Results

Table 1

Network	Mode	Scenario	Algorithm	Roundtrip Time		
				ms	Δ	%
Wired	–	Base	–	13.0	–	–
Wired	CCIM	TPA	–	145.0	+132	1015%
Wired	CCIM	Sign	ECDSA	152.0	+139	1000%
Wired	CCIM	Sign	RSA	155.0	+142	1023%
Wired	CCIM	PUF	ECDSA	151.0	+138	992%
Wired	CCIM	PUF	RSA	153.0	+140	1008%
Wired	EIM	TPA	–	215.0	+202	1485%
Wired	EIM	Sign	ECDSA	234.0	+221	1631%
Wired	EIM	Sign	RSA	227.0	+214	1577%
Wired	EIM	PUF	ECDSA	234.0	+221	1631%
Wired	EIM	PUF	RSA	230.0	+217	1600%
WiFi	–	Base	–	60.0	–	–
WiFi	CCIM	TPA	–	179.0	+119	198%
WiFi	CCIM	Sign	ECDSA	186.0	+126	210%
WiFi	CCIM	Sign	RSA	193.0	+133	222%
WiFi	CCIM	PUF	ECDSA	182.0	+122	203%
WiFi	CCIM	PUF	RSA	203.0	+143	238%
WiFi	EIM	TPA	–	280.0	+220	367%
WiFi	EIM	Sign	ECDSA	275.0	+215	375%
WiFi	EIM	Sign	RSA	338.0	+278	463%
WiFi	EIM	PUF	ECDSA	282.0	+222	360%
WiFi	EIM	PUF	RSA	332.0	+272	453%

Experimental Results

- CCIM is on average 55% faster than EIM
- ECDSA is on average 7% faster than RSA
- PUF-based has an average a 0.4% overhead over PUF-less signing
- PUF-based signing has a maximum overhead of 50ms over not signing at all



Experimental Results: Speedup Calculation

Network	Mode	Scenario	Algorithm	Roundtrip Time	
				ms	Δ
Wired	–	Base	–	13.0	–
Wired	CCIM	TPA	–	145.0	+132
Wired	CCIM	Sign	ECDSA	152.0	+139
Wired	CCIM	Sign	RSA	155.0	+142
Wired	CCIM	PUF	ECDSA	151.0	+138
Wired	CCIM	PUF	RSA	153.0	+140
Wired	EIM	TPA	–	215.0	+202
Wired	EIM	Sign	ECDSA	234.0	+221
Wired	EIM	Sign	RSA	227.0	+214
Wired	EIM	PUF	ECDSA	234.0	+221
Wired	EIM	PUF	RSA	230.0	+217
WiFi	–	Base	–	60.0	–
WiFi	CCIM	TPA	–	179.0	+119
WiFi	CCIM	Sign	ECDSA	186.0	+126
WiFi	CCIM	Sign	RSA	193.0	+133
WiFi	CCIM	PUF	ECDSA	182.0	+122
WiFi	CCIM	PUF	RSA	203.0	+143
WiFi	EIM	TPA	–	280.0	+220
WiFi	EIM	Sign	ECDSA	275.0	+215
WiFi	EIM	Sign	RSA	338.0	+278
WiFi	EIM	PUF	ECDSA	282.0	+222
WiFi	EIM	PUF	RSA	332.0	+272

Speedup of CCIM over EIM

$$Speedup = \frac{Roundtrip_{EIM}}{Roundtrip_{CCIM}} - 1$$

$$\overline{Speedup} = \frac{\sum_{i=1}^{K} \frac{Roundtrip_{EIM,i}}{Roundtrip_{CCIM,i}}}{K} - 1$$

$$Speedup = \frac{\left(\frac{215}{145} + \frac{234}{152} + \frac{227}{155} + \frac{234}{151} + \frac{230}{153} + \frac{280}{179} + \frac{275}{186} + \frac{338}{193} + \frac{282}{182} + \frac{332}{203}\right)}{10} - 1 = 55\%$$

Georgia Tech

Outline

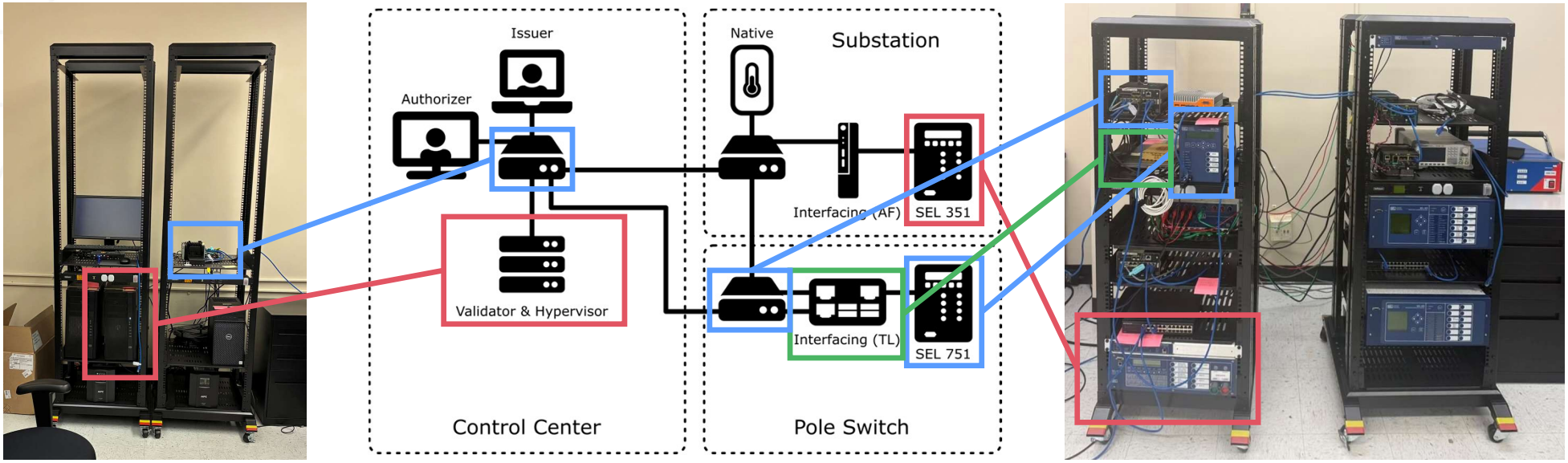
1. Problem Statement
2. Background
    1. Definitions
    2. System Architecture
    3. Attack Surface
    4. TPA Protocol
3. Methodology
    1. Signing Flow
    2. Device Enrollment
    3. Experimental System
4. Results
5. **Discussion**
6. References

Discussion

- The highest roundtrip time in our experiments was 338 ms (WiFi + EIM + RSA)
- The latency for a person to press a button based on the semantic understanding of a word ranges from 400 ms to 1500 ms with an average of 871 ms [4]
- The human factor in TPA dwarfs the latency of encryption, signing, and any other protocol overheads

[4] O. Hauk, et. al.

Future Work: Integrated Testbed System



Outline

1. Problem Statement
2. Background
    1. Definitions
    2. System Architecture
    3. Attack Surface
    4. TPA Protocol
3. Methodology
    1. Signing Flow
    2. Device Enrollment
    3. Experimental System
4. Results
5. Discussion
6. **References**

References

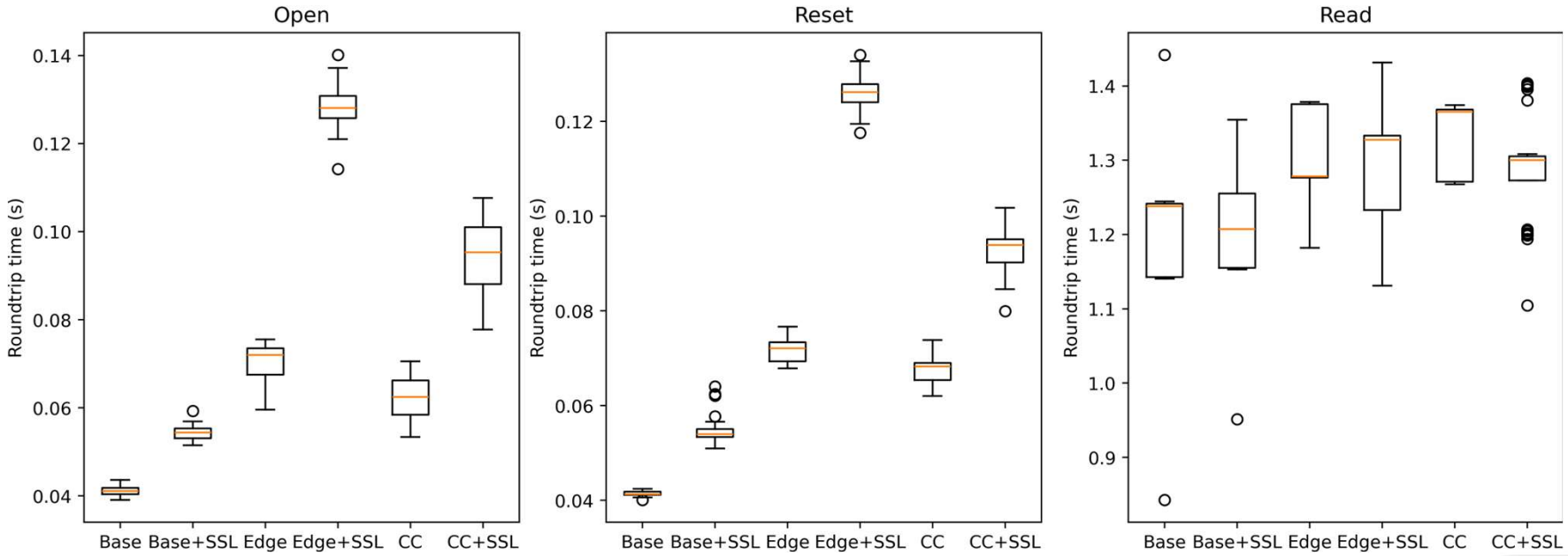
1. CISA, “Iranian-Affiliated Cyber Actors Exploit Programmable Logic Controllers Across US Critical Infrastructure,” U.S.A, AA26-097A, 2026, https://www.cisa.gov/sites/default/files/2026-07/aa26-097a-iranian-affiliated-cyber-actors-exploit-programmable-logic-controllers-across-us-critical-infrastructure_508c.pdf.
2. Roel Maes, *Physically Unclonable Functions: Constructions, Properties and Applications,* Springer, 2013.
3. M. Masse, REST API Design Rulebook. O’Reilly, 2011.
4. O. Hauk, C. Coutout, A. Holden, and Y. Chen, “The time-course of single-word reading: Evidence from fast behavioral and brain responses,” NeuroImage, vol. 60, no. 2, pp. 1462–1477, 2012.

Bonus Slides

SSL Overhead (Prior Work)

TABLE IV: SSL Overhead

Command	Base		EIM		CCIM	
	Δ	%	Δ	%	Δ	%
open	+13	32%	+59	84%	+32	52%
reset	+13	32%	+54	75%	+25	37%
read	+14	1%	-22	-2%	-32	-2%



[5] K. Ahmad, A. Allahverdi, V. Mooney, and S. Grijalva, “GridLogic 2FA: Two-Factor Authentication for Critical Infrastructure Commands in the Field,” in 2026 IEEE Texas Power and Energy Conference (TPEC), 2026.

Georgia Tech