Secure and Privacy-Aware Gateway for Home Automation Systems

Sinem Gür, Simge Demir, Şevval Şimşek and Albert Levi

Department of Computer Science and Engineering
Sabancı University, Istanbul

This work has been partially supported by TÜBİTAK (Scientific and Technological Research Council of Turkey) under grant 117E017

SINCONF 2020, November 4th, 2020

Sabancı Üniversitesi

Introduction

Security threats: connectivity to Internet, simplicity of IoT devices

Home Automation Systems includes a set of interconnected smart devices that are remotely accessible and integrated in the home environment.

A privacy-aware secure identification and authentication protocol for HAS as part of the Turkish-Polish bilateral FUSE (Full-Managed Secure Gateway for Home Automation Systems) project

Proposed system has been originated from the HAS architecture of Batalla and Gonciarz[1]

IHG (Innovative Home Gateway) is the gateway to manage the communication with IoT Devices and outside users through the HMS (Home Management Systems)

Sabancı Üniversitesi

[1] Batalla, J.M., Gonciarz, F. Deployment of smart home management system at the edge: mechanisms and protocols. 2019, *Neural Comput & Applic 31*, 1301–1315.

Cont’d

To ensure security and privacy-preservation → mutual authentication scheme based on the Idemix[2] credential system

Contributions:

- **Privacy**: failed communication by generating **fake Proof** → communication seem to be usual and hide the real identity of the device
- **Identification**: using the anonymous credential system
- **Authentication**: Double verification protocol to provide mutual authentication.

Sabancı Üniversitesi

[2]Camenisch, J., and Van Herreweghen, E., Design and Implementation of the idemix Anonymous Credential System, 2002, *ACM CCS 2002,* Washington, DC, USA.

Home Automation System Architecture

Overview of the Home Automation System architecture

Innovative Home Gateway (IHG)

- A device provided by network operators, similar to a set-top-box
- IoT Devices are registered to IHG at the first boot, don't connect any other device

HAS Management System (HMS)

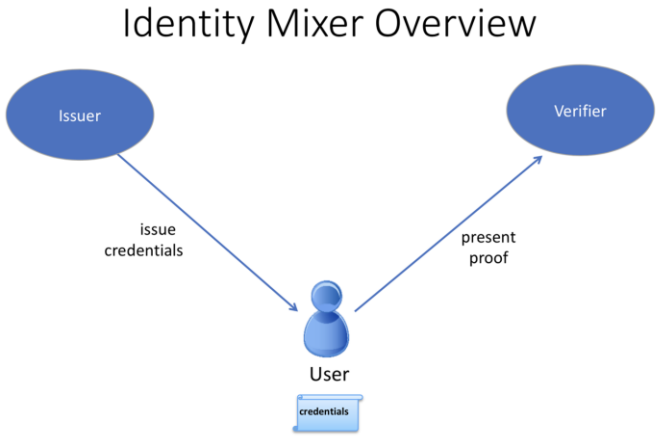
- Server belonging to the network operator, responsible for distribution of IHG set-top-boxes
- Communication between Vendor and IHG using MQTT[3] protocol.

Vendors and Homeowners

- Communicate with the IoT Devices for a particular reason in the system
- Cannot directly send messages to the IoT devices, communicate via IHG

[3]MQTT, "Documentation | MQTT", Accessed: 21 April 2019. http://mqtt.org/documentation

Background Information



1) Idemix[2] Credential System

One of the anonymous credential systems developed by IBM

Three parties in the Idemix system: 1) Issuer; 2) Verifier and 3) User

- Issuer: authority to issue the credentials to the users
- User: gets the credential of the set of attributes that need to be proven
- Verifier: verifies the user's specific attributes by checking its credential

HOW WE ADOPT?

Credentials of devices are issued by the Issuer to be used in the verification process

The communication protocol runs over the credential system

Sabancı Üniversitesi

[4]Camenisch, J., and Van Herreweghen, E., Design and Implementation of the idemix Anonymous Credential System, 2002, *ACM CCS 2002,* Washington, DC, USA.

Cont’d

2) MQTT[3]

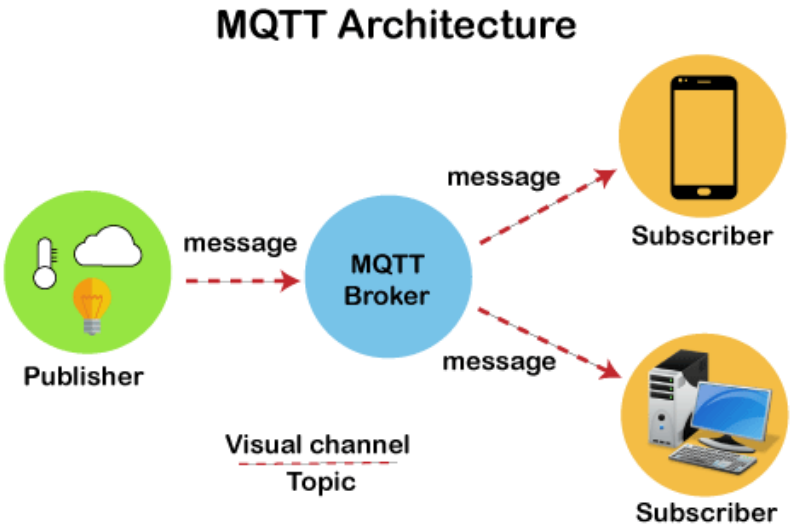
simple and lightweight machine-to-machine communication protocol

especially in the systems including low processing power devices such as IoT Devices

Publish/subscribe mechanism: parties register to the broker, messages are published over a topic, parties are subscribed to specific topic, get the published message

HOW WE ADOPT?

Communication protocol chosen to be MQTT



Sabancı Üniversitesi

[1]MQTT, "Documentation | MQTT", Accessed: 21 April 2019. http://mqtt.org/documentation

System Parts

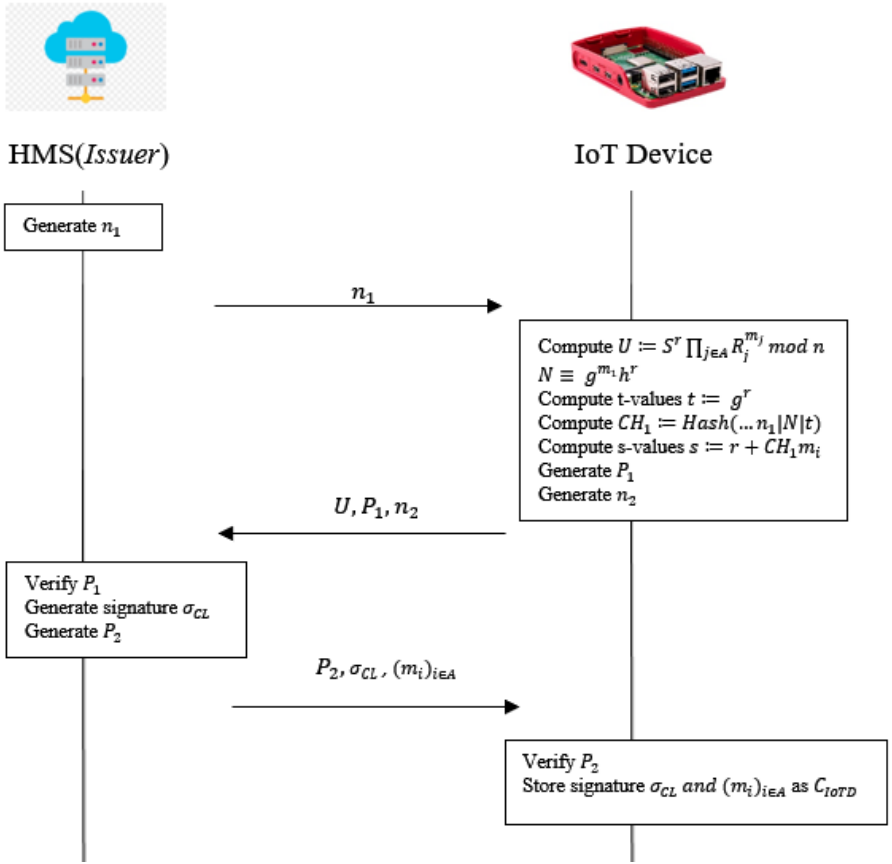
Credential → a document user receives from the Issuer, theoretically states the user has all or any subset of the attributes

Proof Specification Document (proofSpec) →contains the attribute list of the user that is to be proven, all the attributes in the proofSpec should also appear in the credential of the user

Challenge → a random number

Proof → a document that is created by challenge, proofSpec and credential, used for proving that the user has all the attributes specified in the proofSpec

Proposed Privacy-Aware Identification and Authentication Scheme



Initialization

- *Issuer* constructs Credential for each of the IoT devices in the system
- Credentials are specific to IoT devices
- Output: credentials are received and saved for the verification phase

Sabancı Üniversitesi

Proof Generation Algorithm

- Check whether **Credential** exist or not
- Load necessary inputs from **Credential**
- Check if IoT device has all the attributes in **proofSpec** (if proofSpec is a subset of of the Credential)
- If so; generate **Proof** using *buildProof* method from Idemix library[1]
- If not; generate a dummy, seemingly legitimate document as **fakeProof**

Algorithm 1 Proof Generation Algorithm

INPUT: C_{IoTD}, PS_{IoTD} and CH_1
OUTPUT: $Proof_{IoTD}$

1: **if** C_{IoTD} not exists **then**
2: Get C_{IoTD}
3: **end if**
4: Load C_{IoTD} inputs
5: **if** $PS_{IoTD} \subseteq C_{IoTD}$ **then**
6: Randomize $A' = AS^r \ mod \ n$
7: Compute t-values $t \coloneqq (A')^r (\prod R_i{}^{m_i}) (S^r)$
8: Add A' to common values list **common**
9: Compute $CH_2 \coloneqq Hash(\ldots | \boldsymbol{common} | t | CH_1)$
10: Compute s-values $s \coloneqq r + CH_2 m_i$
11: Generate $Proof_{IoTD} \coloneqq (\boldsymbol{common}, CH_2, s)$
12: **end if**
13: **if** not $PS_{IoTD} \subseteq C_{IoTD}$ **then**
14: Build a dummy document as $fakeProof_{IoTD}$
15: **end if**

Sabancı Üniversitesi

[1]IBM Research Zürich Security Team, Specification of the Identity Mixer cryptographic library,version 2.3.40, 2013, IBM Research, Zürich.

Double Verification Protocol

Step 1:

- Vendor sends specified proofSpec to IHG
- IHG selects appropriate IoT devices to start the communication
- Devices send “Verify command"

Step 2:

- Vendor → **$challenge_1$** sent to IoT device
- IoT device →
  - uses **$challenge_1$** to generate **$Proof_{IoT}$** to authenticate itself
  - Or generate **$fakeProof_{IoT}$**
  - Generate **$challenge_2$**
  - For integrity → **$HMAC^1$ of $challenge_2$**
- IoT device → **$challenge_2$, HMAC of $challenge_2$, $Proof_{IoT}$** to Vendor

***HMAC:** Keyed-Hashing for Message Authentication

*For the update case
*All communications are handled by IHG (Innovative Home Gateway)
*Mutual authentication ensured

Sabancı Üniversitesi

[1]Krawczyk, H., Bellare, M., and Canetti, R., HMAC: Keyed-hashing for message authentication, 1997, RFC-2104.

Cont’d

Step 3: (Ensure mutual authentication)

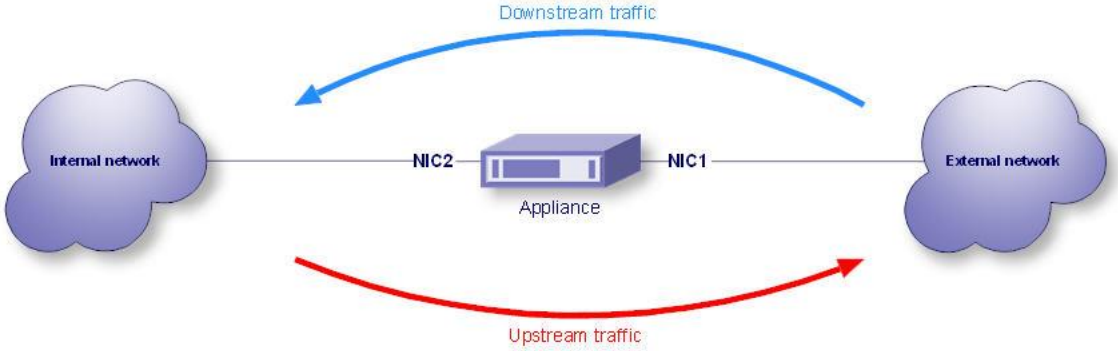
- Vendor →
  - Verify **HMAC[5] of challenge$_2$**
  - If not; give error message
  - Verify **Proof$_{IoT}$**
  - If verified; crate **Proof$_{Vendor}$**
  - If not verified; create **fakeProof$_{Vendor}$** to preserve privacy
  - For integrity → **HMAC of update message**
- Vendor → **update message, HMAC of update message, Proof$_{Vendor}$** to IoT device
- IoT device→
  - Verify **HMAC of update message**
  - If not; give error message
  - Verify **Proof$_{Vendor}$**
  - If verified; get the update message

Sabancı Üniversitesi

[5]Krawczyk, H., Bellare, M., and Canetti, R., HMAC: Keyed-hashing for message authentication, 1997, RFC-2104.

Performance Evaluation

- Experiments → downstream and upstream traffic scenarios in the proposed architecture
- downstream traffic: sending updates from Vendor to IoT devices
  - Successful update
  - Failed update
- Upstream traffic: sending error report from IoT devices to Vendor



Sabancı Üniversitesi

Setup

- Implemented applications:
  - IoT devices, IHG and Vendor; using Java Programming Language
- All of the IoT devices → IHG via a WLAN network
- In a real-life household; connecting the IHG box to the regular router via an ethernet cable and connecting the smart devices to the IHG

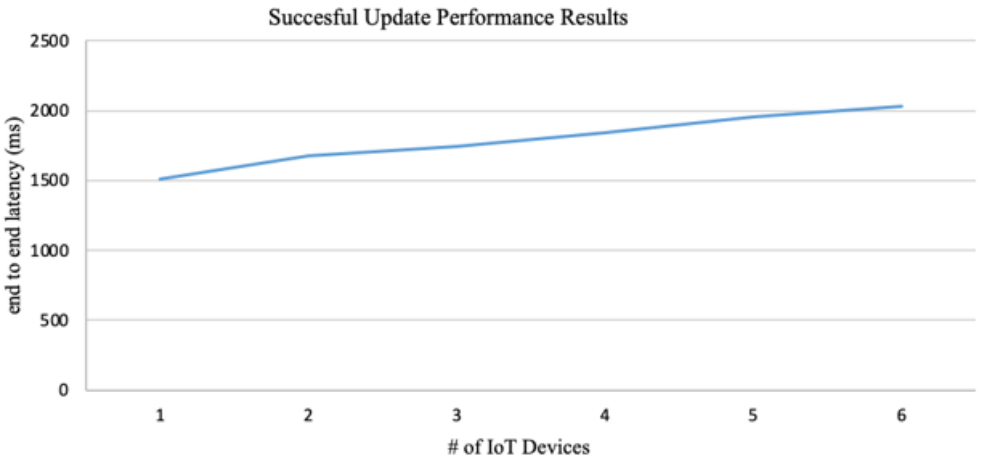
Raspberry Pi[6] for IoT devices and IHG

Laptops for Vendor and MQTT server

Sabancı Üniversitesi

[6]Raspberry Pi Foundation Accessed: 30 May 2019, https://www.raspberrypi.org/.

Successful Update Scenario



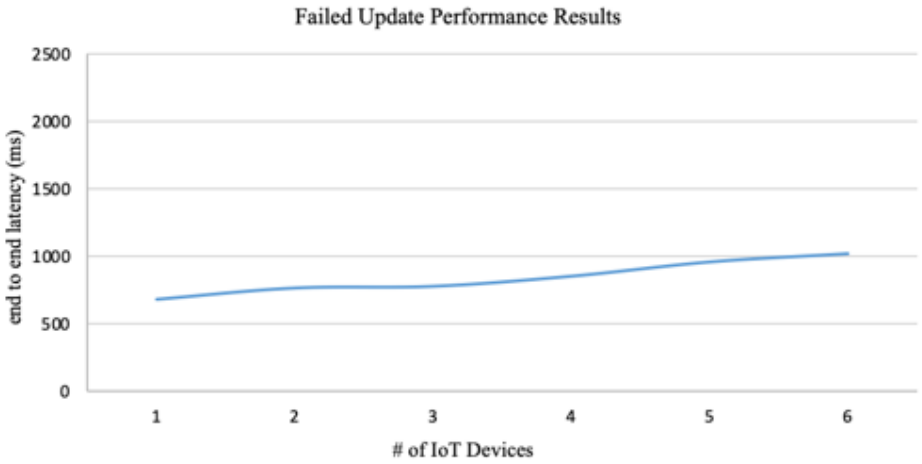
Multiple IoT devices have undergone simultaneous update operations

Overall end-to-end latency results of these multiple devices update cases with 1-to-6 IoT devices

The increase in latency with respect to increase in the number of IoT devices is linear

Sabancı Üniversitesi

Failed Update Scenario



Preserve privacy → generating fakeProof

- complete the protocol no matter whether the update is addressed to them or not
- hides its real identity from third parties
- outsider cannot differentiate the actual target IoT devices of the update

Between 0.6 - 1.1 seconds with linear increase at approximately 0.1 second per device

Sabancı Üniversitesi

Mixed Update Scenario

Successful/Failed Update	Time(ms) RaspberryPi4 gets always successful update	Time(ms) RaspberryPi4 gets always failed update
1 Successful/ 5 Failed	1193	1733
2 Successful/ 4 Failed	1746	1786
3 Successful/ 3 Failed	1785	1935
4 Successful/ 2 Failed	1925	1981
5 Successful/ 1 Failed	1946	2004

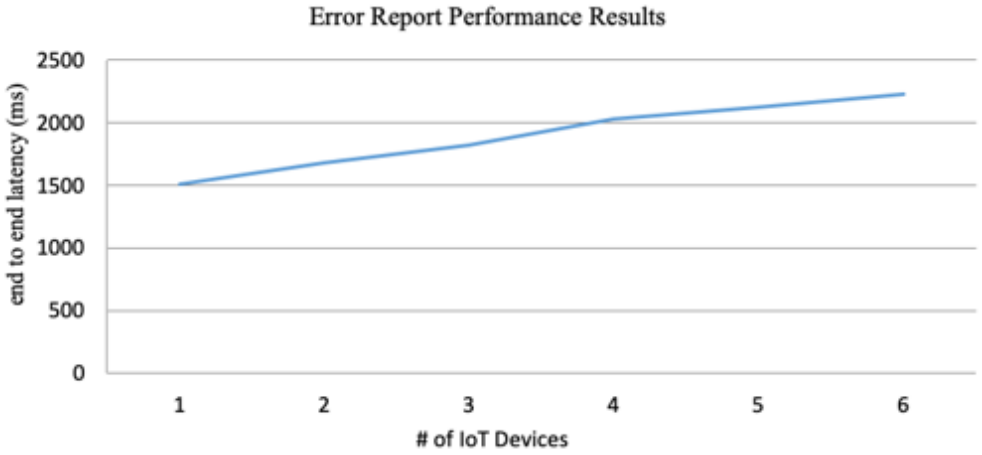
Experiments conducted for different scenarios with mixed and simultaneous failed and successful cases

Since the double verification protocol is ensured for successful updates in all cases, latencies are close to each other

Even one successful update increases the computational complexity

Sabancı Üniversitesi

Error Report Scenario



Results of the successful delivery of error reports

The results for end-to-end latency of multiple IoT Devices scenarios → in the figure

Latency values increase linearly

Sabancı Üniversitesi

Discussions & Concluding Remarks

- Failed update scenario is much faster →
  - Generating fakeProof faster than generating Proof
  - Reduce calculation for proof verification
- All scenarios → linear graphs
  - Due to increasing communication and computation
- Slope of increases → quite low showing the scalability of the proposed approach

Summary

- privacy-aware secure identification and authentication model
- Scheme : IoT devices, IHG, HMS and Vendor
- IoT devices →
  - communicate via IHG
  - not open to external communication
- IHG →
  - Gateway at home
  - Relays messages
- Secure identification and authentication →
  - Double Verification Protocol to ensure mutual authentication
- Data integrity →
  - Using HMAC
- Privacy-preserving →
  - Idemix Credential System adaptation
  - fakeProof generation
- Experimental results → high efficiency, scalability

Sabancı Üniversitesi

Future Work

- Development of enhanced privacy-preserving mechanism
  - To obfuscate the communication between IHG and Vendor
- Development of the application for HomeOwners
  - Test with an extended test set

Sabancı Üniversitesi

ACKNOWLEDGMENTS

This work has been partially supported by TÜBİTAK (Scientific and Technological Research Council of Turkey) under grant 117E017

Sabancı Üniversitesi