

A Federated Lightweight Authentication Protocol for the Internet of Things

Maria L. B. A. Santos^a, Jéssica C. Carneiro^a, Antônio M. R. Franco^a, Fernando A. Teixeira^b,
Marco A. Henriques^c, Leonardo B. Oliveira^{d,a,*}

^a*UFMG, Belo Horizonte, Brazil*

^b*UFSJ, Ouro Branco, Brazil*

^c*Unicamp, Campinas, Brazil*

^d*Visiting professor at Stanford University, Stanford, USA*

Abstract

Considering the world's IoT development and market, it is necessary to guarantee the security of the developed IoT applications as well as the privacy of their end users. In this sense, Federated Identity Management (FIdM) systems can be of great help as they improve user authentication and privacy. In this paper, we claim that traditional FIdM are mostly cumbersome and then ill-suited for IoT. As a solution to this problem, we come up with a federated identity authentication protocol exclusively tailored to IoT. Federated Lightweight Authentication of Things (FLAT), our solution, replaces weighty protocols and asymmetric cryptographic primitives used in traditional FIdM by lighter ones. For instance, FLAT synergistically combines symmetric cryptosystems and Implicit Certificates. The results show that FLAT can reduce the data exchange overhead by around 31% when compared to a baseline solution. FLAT's Client is also more efficient than the baseline solution in terms of data transmitted, data received, total data exchange, and computation time. Our results indicate that FLAT runs efficiently even on top of resource-constrained devices like Arduino.

Keywords: Internet of Things, authentication, federated identity management

1. Introduction

The development of the Internet of Things (IoT) Atzori et al. (2010); Gubbi et al. (2013); Borgia (2014) is a national priority in several countries around the world and is significantly impacting our society. Studies suggest that we will be surrounded by around 20 billion IoT devices in 2020¹. The IoT development has enabled a diverse number of applications in academy and industry.

When it comes to IoT, one of the most significant challenges to its full realization lies in the field of Identity Management (IdM). IdM refers to the identification of users in a given

*Corresponding author

Email addresses: mburga@dcc.ufmg.br (Maria L. B. A. Santos), jessicarneiro@dcc.ufmg.br (Jéssica C. Carneiro), franco@dcc.ufmg.br (Antônio M. R. Franco), teixeira@ufs.j.edu.br (Fernando A. Teixeira), marco@dca.fee.unicamp.br (Marco A. Henriques), leob@dcc.ufmg.br (Leonardo B. Oliveira)

¹<https://www.gartner.com/en/newsroom/press-releases/2017-02-07-gartner-says-8-billion-connected-things-will-be-in-use-in-2017-up-31-percent-from-2016>

system (e.g., a network, application, or service) and controlling their access to resources within that system. (Here, the term *identification* means the process for authenticating the identity of a user Stallings (2016).). Ideally, IdM provides administrators with the tools to manage the full user's identity life-cycle (e.g., setup, maintenance, and tear down) and, thus, are vital to the security and productivity of organizations.

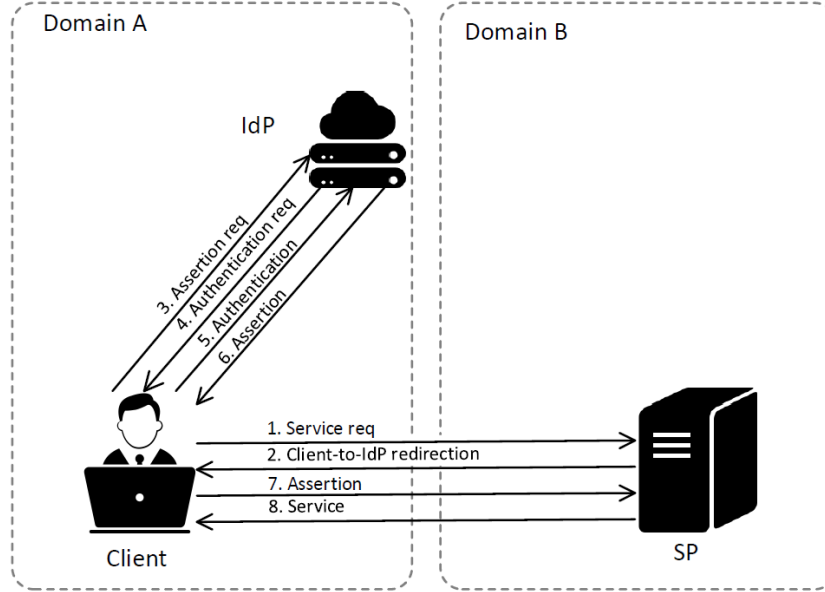


Figure 1: Traditional FIdM (adapted from Birrell and Schneider (2013), Figure 2).

Federated Identity Management (FIdM) Shim et al. (2005), in turn, improves the IdM idea by enabling a domain to control accesses to its resources from external users. For instance, it allows an external user to authenticate to a local server and utilize its services without having to create an identity or register credentials locally. Instead, the authentication process between the user and the Service Provider (SP) is mediated by the Identity Provider (IdP) of the user's home domain. FIdM, hence:

- enables applications like Single-Sign-On (SSO);
- increases privacy by limiting the amount of information shared;
- and improves the end-users experience and security by eliminating the need for new accounts registration and restricting the number of entities that hold their password.

Fig. 1 illustrates how traditional FIdM works. The Client initiates the communication with the SP (Fig. 1, step 1). The SP, in turn, redirects the Client to the IdP (step 2). Next, the Client requests an assertion (or token) to the IdP (step 3) and authenticates itself to the IdP by presenting its credentials (steps 4 and 5). The IdP, in exchange, sends the assertion to the Client (step 6). This assertion is then used by the Client to get access to the SP service (steps 7 and 8). All these steps are normally protected and authenticated by public key infrastructure certificates, which have to be validated by all participants.

So far, unfortunately, IoT technology cannot fully enjoy the benefits of either IdM or FIdM. This is so because both IdM or FIdM widely adopted approaches are inadequate to IoT Fremantle et al. (2014). First and foremost, there is no such thing as IdM for IoT devices. Instead, in existing (F)IdM schemes, IoT devices make use of credentials of individual (human) users to log on and enjoy domain services. And this is both insecure and inappropriate as, respectively, devices most likely should not have the same clearance level as those users and some IoT devices just cannot be naturally linked to any individual user (e.g., whose user should a traffic light be?). Second, the IoT mobile nature and dynamics urge a higher level of scalability and interoperability across multiple domains when compared to conventional network elements Fremantle et al. (2014). And last but not least, the authentication process on existing (F)IdM schemes normally build upon the login/password paradigm, which is usually okay for humans but ill-suited for devices Fremantle and Aziz (2016), and leverage expensive RSA/DSA cryptosystems, hence, incurring significant computational resources overhead Perrig et al. (2002). So, there is patently a dire need for a FIdM able to meet IoT special needs.

As a solution to this problem, we propose a FIdM protocol exclusively tailored to IoT. In short, we design, developed, and evaluate a prototype of such a protocol. Our solution, called Federated Lightweight Authentication of Things (FLAT) Santos et al. (2018), replaces cumbersome protocols and asymmetric (public-key) cryptosystems used in traditional FIdM by more efficient ones and, thus, well-suited for IoT². Notably, FLAT synergistically combines symmetric cryptosystems and Implicit Certificates Brown et al. (2002) to come up with a federated authentication protocol. We built a prototype of FLAT and evaluated its performance.

This paper is organized as follows. We first argue the need for lightweight cross-domain authentication schemes (Section 2). Next, we present how FLAT meet this need (Section 3). We describe FLAT development (Section 4), discuss its performance figures (Section 5), and then sum up our findings (Section 7).

2. The Call for a Lightweight Cross-domain Authentication

There are many situations when IoT nodes from different domains need to talk to each other in a secure manner. A real-world situation that came to our attention by way of one of the biggest technology company is this: Suppose that an autonomous (driverless) truck that hauls iron ore has gotten out of order in the middle of a mine³. And that a technical rescue team has to provide *in-situ* emergency care in order to get the truck fixed. To do so, ideally, the rescue team needs its technical apparatus to communicate seamlessly with the truck’s engine to start providing assistance. Note, here, we have devices (the engine and the technical equipment) from distinct domains (the mining company and the technical rescue company) that need to inter-operate as soon as possible (as this truck broken down may nearly lead to million-dollar losses per day) and in an authenticated manner (because a truck like this is very expensive and cannot be accessed by someone or something it does not recognize).

Another example of such a need is automated cashless toll systems. Here, cars (clients) are equipped with tags that automatically authenticates to the toll gate, authorizing the car to get into the tollway after processing the payment. Note, however, that a plain implementation

²Santos et al. Santos et al. (2018) present the general concept of FLAT and preliminary results of the solution.

³<https://www.technologyreview.com/s/603170/mining-24-hours-a-day-with-robots/>

of such a system (i.e., without the employment of FIdM or other security countermeasures to protect privacy) is inappropriate to both car users and tollway companies. The system is inappropriate to the user because it trades privacy for convenience⁴. For instance, it makes it possible for the toll company to keep track of the users' mobility pattern and then infer part of users daily routine. And the system is also inappropriate to the toll company because it may not comply with contemporary data privacy laws (*e.g.*, The EU General Data Protection Regulation).

Note the employment of a FIdM system could easily solve the above privacy issue were it not for the resources it requires. For instance, one of the most popular protocol for cross-domain message exchange, the Security Assertion Markup Language (SAML) Maler and Reed (2008), relies on asymmetric cryptography and then is inadequate to the IoT scenario, due to the computational resources required. There are indeed streamlined FIdM proposals, some even tailored to IoT (*e.g.*, Fremantle and Aziz (2016), Cirani et al. (2015), Domenech et al. (2016)). However, those proposals still require asymmetric operations from Clients and, therefore, are not ideal for resource-constrained environments. So, the major challenge for solving the privacy issue lies in designing a FIdM that fits the resources of IoT devices.

3. FLAT: A Federated Lightweight Authentication Protocol for IoT

FLAT adapts the traditional FIdM authentication to achieve a federated lightweight authentication solution for IoT. In what follows, we talk about the FLAT various assumptions (Section 3.1), present FLAT itself (Section 3.2), discuss FLAT design decision making (Section 3.3), and then present FLAT message description (Section 3.4).

3.1. Assumptions

FLAT focus on a critical part of a FIdM model: authentication. For other parts, FLAT assumes the use of existing approaches. Below, we describe FLAT assumptions.

3.1.1. Device capability

FLAT assumes the Client, SP, and IdP will be run over devices of low, medium, and high computational capabilities, respectively. This is so because:

- IoT devices are normally resource-constrained, making the Client the main motivation of our work;
- the SP, in the end, is still a kind of server and, most likely, will be endowed with more resources than the Client;
- and the IdP is probably like any other IdPs, *i.e.*, a Cloud server that will run 24/7.

By definition, IoT devices are interconnected via the Internet. As for the Client, this can be done either through the devices' own capabilities (*e.g.*, by featuring a mobile chip like modern e-book readers/smartwatches usually do) or indirectly by requiring the device to connect to a

⁴<https://www.theverge.com/2013/3/27/4150702/golden-gate-bridges-new-cashless-tollway-promises-convenience-for-privacy>

hotspot close to the SP. After obtaining the SP service broadcast, the Client could also ask the SP to forward its packets to the IdP, as it is expected that the SP is permanently connected to the Internet.

3.1.2. Service Discovery

Service discovery allows participants of a network to do things like service announcement and consultation Ververidis and Polyzos (2008). In IoT, devices use services provided by SPs physically close to them and, thus, SPs convey service information using beacons, *i.e.*, periodically broadcasted messages within a certain radius. FLAT assumes that the Client discovers the SP and its services using beacons.

3.1.3. Trust Establishment

FLAT assumes the trust between IdPs and Clients as well as IdPs and SPs takes place *a priori*. FLAT assumes the trust between IdPs and Clients is established during a Client setup, where shared symmetric keys are pre-loaded into IdPs and Clients.

As for IdPs and SPs, they are normally from different domains and, previously, not known to each other. FLAT assumes the trust between them derives from digital certificates issued by a common Certification Authority.

3.1.4. Interfederation

In case the Client tries to use a service offered in a different federation, FLAT assumes a model based on the GÉANT Authorisation INfrastructure for the research and education community (eduGAIN)⁵ for communication between different federations. eduGAIN is an interfederation service that allows the connection of academic identity federations and their respective infrastructures for authentication and authorization. The interfederation can be seen as a way of making certificates issued by one federation Certification Authority valid in another federation. This is done through cross-certification or the creation of a new root Certification Authority that will issue certificates for the Certification Authorities of all participant federations.

3.1.5. Setup

FLAT assumes that a number of procedures will take place prior to the Clients deployment. A central point for security is the generation of cryptographic keys. Physical Unclonable Functions (PUFs) are known to be resistant to physical attacks. FLAT, therefore, assumes the use of PUFs for generating the device's cryptographic keys Suh and Devadas (2007). In FLAT, briefly, we assume the PUF output is generated, formatted and stored in each device, shared with the IdP and then used as a key to secure the Client↔IdP communication. We assume this whole process takes place in a secure environment (*e.g.*, at a physically secure room inside the device's home domain.)

⁵https://www.geant.org/Services/Trust_identity_and_security/eduGAIN

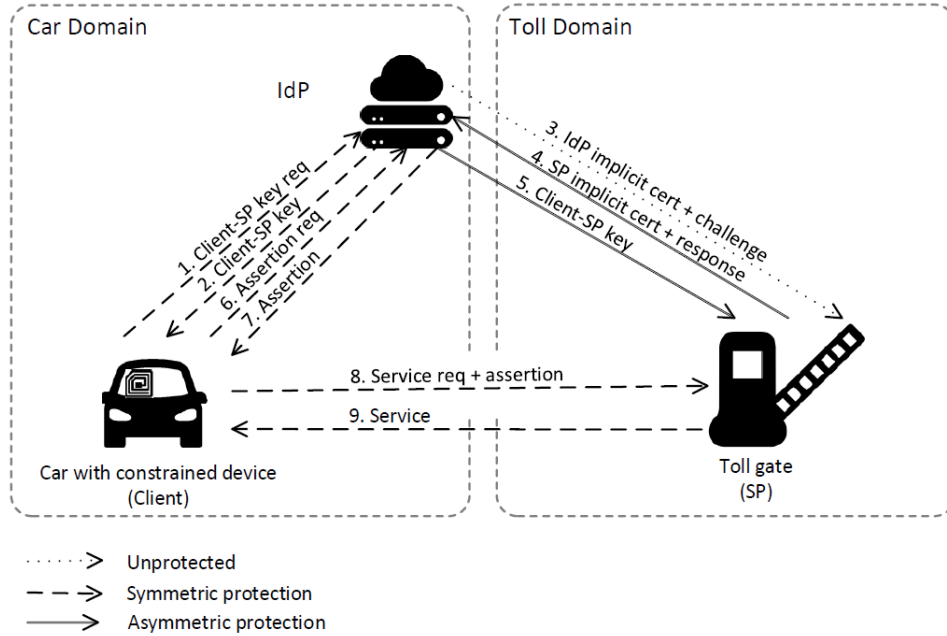


Figure 2: FLAT applied to a tollway scenario.

3.2. Description

Fig. 2 shows a high-level overview of FLAT’s operation. For readability, we decided not to represent all the cryptographic primitives in Fig. 2, individually. There, instead, dotted lines denote no cryptosystem being used, dashed lines denote symmetric cryptosystems (MACs and secret-key encryption), and solid lines, asymmetric cryptosystems (digital signatures and public-key encryption).

We use a scenario where a car with an attached constrained IoT device interacts with an automated cashless toll system to illustrate FLAT (Fig. 2). Here, the toll company represents the SP; the device attached to the car, the Client; and the car domain (the car rental company, the car company owner, the department of motor vehicles etc.) the IdP. In context, the service here is the tollway, *i.e.*, the passage through the toll road. Broadly, the Client requests to its IdP a session key and an assertion. The IdP, in turn, sends securely the session key to both the Client and SP. Besides, the IdP sends a signed assertion to the Client. Last, the Client uses the session key to secure the communication with the SP and forward the signed assertion to be validated by it in order to allow access to the requested service.

Notably, whenever the Client wants to access a service (*e.g.*, go through a toll gate), it requests the IdP to issue a key to the SP and the Client itself (Fig. 2, step 1), so the Client and SP can communicate with each other, securely. The IdP respond to this request and issues the key to the Client (Fig. 2, step 2) and SP (Fig. 2, steps 3, 4, and 5). Regarding the SP, this is a three-hand shake process because the SP and IdP first exchange certificates (in this case, Implicit Certificates – Fig. 2, steps 3 and 4) to establish a secure channel for the key issuing (Fig. 2, step 5). Next, as usual in FIdM, the Client requests and receives an assertion (Fig. 2, step 6 and 7). Last, the Client requests the service by presenting the assertion to the SP (Fig. 2, step 8), then the SP provides the service (Fig. 2, step 9) – *e.g.*, by opening the toll

OPERATION(*Client C*, *Identity Provider IdP*, *Service Provider SP*)

- 1.1. $C \rightarrow IdP : \text{MAC}(k_{IdP,C}, n_C \mid SP \mid \text{key_req})$
- 2.1. $IdP \rightarrow SP : n_{IdP} \mid \text{cert}_{IdP}$
- 2.2. $SP \rightarrow IdP : \text{SIGN}(S_{SP}, \text{cert}_{SP} \mid n_{IdP} \mid n_{SP} \mid n'_{SP})$
- 2.3. $IdP \rightarrow SP : \text{SIGN}(S_{IdP}, \text{ENC}(P_{SP}, k_{SP,C}) \mid n_{SP} \mid n'_{IdP})$
- 2.4. $SP \rightarrow IdP : \text{SIGN}(S_{SP}, n'_{IdP})$
- 1.2. $IdP \rightarrow C : \text{MAC}(k_{IdP,C}, \text{ENC}(k_{IdP,C}, k_{SP,C}) \mid \text{'serv[1...n]'} \mid n_C \mid n'_{IdP})$
- 3.1. $C \rightarrow IdP : \text{MAC}(k_{IdP,C}, \text{ENC}(k_{IdP,C}, \text{assert_req}(\text{'serv[i]'})) \mid n'_{IdP} \mid n'_C)$
- 3.2. $IdP \rightarrow C : \text{MAC}(k_{IdP,C}, \text{ENC}(k_{IdP,C}, \text{assert} = \text{SIGN}(S_{IdP}, \text{'serv[i]'} \mid n'_{SP}))) \mid n'_C)$
- 4.1. $C \rightarrow SP : \text{MAC}(k_{SP,C}, \text{ENC}(k_{SP,C}, \text{assert}) \mid n'_C \mid n'_{SP})$
- 4.2. $SP \rightarrow C : \text{MAC}(k_{SP,C}, \text{ENC}(k_{SP,C}, \text{serv[i]}) \mid n'_C)$

The symbols denote:

- \rightarrow : unicast transmission
- \mid : concatenation
- $\text{MAC}(k, m)$: MAC over m calculated using key k
- $k_{X,Y}$: symmetric key shared by X and Y
- n_X : nonce generated by X
- key_req : key request label
- assert_req : assert request label
- cert_X : X's certificate
- S_X : X's private key
- P_X : X's public key
- $\text{SIGN}(k, m)$: signature over message m using key k
- $\text{ENC}(k, m)$: encryption over message m using key k
- $\text{'serv[1...n]'$: list of services
- $\text{'serv[i]'$: description of service i
- serv[i] : service i
- assert : assertion

Protocol 1: FLAT protocol description.

gate.

Protocol 1 shows the complete description of FLAT, with all steps taken in the authentication process. First, Client sends a key request to IdP, indicating the SP providing the service it would like to access (step 1.1, Protocol 1). The IdP then exchange certificates with the SP (steps 2.1 and 2.2, Protocol 1) and after verifying SP's certificate, sends a symmetric key to the SP, so it will be able to communicate with the Client (step 2.3, Protocol 1). The SP sends a signed nonce to the IdP as a confirmation of the reception of the key (step 2.4, Protocol 1). Next, the IdP also sends the symmetric key to the Client (step 1.2, Protocol 1). The Client can then send an assertion request to the IdP (step 3.1, Protocol 1), that responds with the assertion (step 3.2, Protocol 1). This assertion signed by the IdP will be forwarded to the SP (step 4.1, Protocol 1), that will then be able to provide the service to the Client (step 4.2, Protocol 1).

3.3. Design Highlights

3.3.1. Efficiency

In FLAT, we have taken some measurements to guarantee the efficiency of the protocol, especially on the Client side. First, we decided that all communication involving the Client — the most constrained device — will be carried out by using symmetric cryptosystems. These are more efficient than asymmetric cryptosystems regarding not only computation but also communication (*e.g.*, they do not require certificate exchange). To make this come true and, yet, secure the communication between the Client and SP we use the IdP as a Key Distribution Center, or KDC for short.

Second, we tried to decrease the message exchange load on the Client side. By way of example, instead of the Client first contacting the SP (Fig. 1, step 1) for only then being redirected to the IdP (Fig. 1, step 3), Clients, in FLAT, start by directly contacting the IdP (Fig. 2, step 1) as soon as it gets the SP service announcement. (Recall we assume Clients get aware of the SP and the services it provides through the use of Service Discovery – c.f. Section 3.1, Service Discovery.)

Last, FLAT replaces conventional certificates with Implicit Certificates Brown et al. (2002) — also known as Elliptic Curve Qu-Vanstone (ECQV) — during the SP-IdP communication (Fig. 2, steps 3 and 4). Implicit Certificates reconstructs public keys from public information and, hence, there is no need of explicitly storing them in certificates. This, in turn, reduces the sizes of the certificates and then saves bandwidth. Further, the certificate verification in this cryptosystem is performed implicitly which, in turn, makes it more efficient, too. (For detailed information on Implicit Certificates, please refer to Brown et al. (2002).).

3.3.2. Security

Threat Model – We assume that the adversary has access to the messages exchanged in the protocol, *i.e.*, she can eavesdrop, collect, modify and replay messages previously sent in the protocol. We also consider that the attacker might have physical access to the devices. In order to mitigate risks related to the physical access, as mentioned in Section 3.1, the device’s key is generated using PUFs. We also assume there is a secure channel to transmit the symmetric key shared between Client and IdP in the setup phase.

Table 1: Countermeasures adopted by FLAT.

Property	Countermeasure
Authenticity	MACs and digital signatures
Confidentiality	Symmetric and asymmetric encryption
Liveness	Nonces
Integrity	MACs, digital signatures and PUFs
Availability	Authentication between the parties
Privacy	FIdM model: limit the access to identity information

FLAT uses (i) Message Authentication Codes (MACs) and digital signatures for providing both authentication and integrity; (ii) ciphers for confidentiality; and (iii) nonces for liveness, *i.e.*, for preventing replay-attacks Stallings (2016). Table 1 summarizes the countermeasures

adopted by FLAT in order to reduce the possibilities of attacks related to each of the security properties.

Authenticity – FLAT uses MACs and digital signatures to ensure authenticity in its communications. In Client \leftrightarrow IdP communication, FLAT relies on a pre-shared symmetric key. When the IdP sends a message to Client using this key, Client knows it is its IdP, since only the IdP has access to the symmetric key shared through a secure channel in the pre-deployment stage. The same applies to the case when the Client is sending a message to its IdP. The symmetric key is also used in MACs that protect the messages being exchanged and ensures its authenticity. Regarding the IdP \leftrightarrow SP communication, it starts with a certificate exchange. As both SP and IdP certificates are signed by a common CA, they can be verified by both entities. When the SP receives a message signed by the IdP and verifies it using the certificate's public key, the SP knows it was really the IdP who sent the message. Similarly, the IdP can verify the authenticity of the messages sent by the SP. Communication between Client and SP also makes use of a symmetric key distributed by the IdP to both SP and Client. The authenticity of messages sent between Client and SP is ensured by MACs.

SP's certificate sent in step 2.2, Protocol 1 is included within the signature. This signature is not necessary, as the certificate bears the CA's signature. This certificate was included in the signature for the sake of readability. FLAT uses implicit certificates (ECQV) to authenticate IdP \leftrightarrow SP communication. The implicit certificate scheme is considered secure¹⁶, i.e., an attacker is not able to successfully falsify an implicit certificate and a private key that was not issued by a legitimate CA Brown et al. (2009).

Confidentiality – In order to protect the confidentiality of the messages containing sensitive information, FLAT uses symmetric encryption in Client \leftrightarrow IdP and Client \leftrightarrow SP communication, and also asymmetric encryption in IdP \leftrightarrow SP communication. Messages not carrying sensitive information, such as nonces and certificates, are not encrypted. FLAT mitigates the possibility of message eavesdropping, since messages containing sensitive information are encrypted, and thus, even if an attacker is able to intercept one of FLAT's messages, she will not be able to decrypt its contents. FLAT also adopts countermeasures to prevent Man-In-The-Middle attacks (MITM), by requiring proof that they are really whom they claim to be during the communication. In FLAT's operation, the IdP \leftrightarrow SP communication is authenticated through the use of digital signatures, with certificate exchange before the shared key is sent to the SP. Additionally, Client's identity is confirmed by the IdP and vice-versa, by the pre-shared symmetric key. Client \leftrightarrow SP communication is also authenticated since their identities have already been verified by the IdP before the shared symmetric key was sent to each of them.

Liveness – FLAT uses in its operation randomly generated nonces to ensure the liveness of the messages being sent. Thus, even in case an attacker is able to intercept a message and replays it, the attack will not be successful since the liveness of the message can be verified by the nonces. FLAT also uses a nonce to identify the session with the SP. Besides ensuring the liveness of the message, the nonce n'_{SP} (received by the IdP in step 2.2, Protocol 1) is sent to the Client (step 3.2, Protocol 1), so when the SP will receive this message (step 4.1, Protocol 1) it will know the message refers to the same service request made in the previous steps. Although it is an unlikely possibility, if an attacker has access to a large number of messages and one of the nonces is repeated at some time, she might be able to perform an attack using an old message. For the sake of readability, nonce encryption is suppressed from the protocol description.

Integrity – FLAT uses MACs and digital signatures to ensure the integrity of the messages. In Client \leftrightarrow IdP and Client \leftrightarrow SP communication MACs are used to provide authenticity and integrity to the messages. IdP \leftrightarrow SP communication is protected using digital signatures to guarantee non-repudiation, authenticity, and integrity to the messages being exchanged. FLAT’s adopted countermeasure to protect against tampering and MITM attacks to integrity is to use MACs or digital signatures, as explained before. To protect against physical tampering, FLAT uses PUFs to generate keys. Thus, even if a physical tampering will occur in the IoT device, there will be changes in its behavior and the key generation will be compromised. In this case, it will not be possible to communicate with the IdP and the protocol will not run.

Availability – Availability ensures that the system will be available when the users will need it. Note that in FLAT the Service Discovery (Section 3.1, Service Discovery) precedes any authentication. Therefore, bogus service advertisement messages can be received by the Client. A fake SP doing such advertising will not have a proper certificate. As the protocol runs by, however, the IdP will eventually figure this out (Fig. 2, steps 3 and 4) and then abort the communication with the source of these advertisements. Thus, the IdP will neither provide the SP with a session key nor the Client with a signed assertion for the required service. Therefore, the Client will be protected by its IdP against fake SPs.

Privacy – FLAT increases privacy by limiting the number of entities that hold the device’s identity information by only its own IdP, as it adopts the federated model. This model also increases privacy as it allows the devices to limit the identity information that they would like to share with the SPs.

Type	Seq	Destination ID	Source ID	Payload size	Payload
1 byte	1 byte	3 bytes	3 bytes	2 bytes	Variable (up to 280 bytes)

Figure 3: FLAT Message description.

3.4. Message description.

The message used in FLAT is shown in Fig. 3. The first byte is used for message type – in FLAT, there are ten message types, namely: **key request**, **Client key**, **certificate-challenge**, **certificate-response**, **SP key**, **key acknowledgment**, **assertion request**, **assertion**, **service request**, and **service**. The second byte is used for storing the sequence number. The following two fields have three bytes each and are reserved to store the source and destination identifiers in the application layer. Hence, FLAT supports domains up to 2^{24} (≈ 16 million) clients in size. Next, we have a two-byte field to store the payload size. Recipients use this field to get to know the amount of data they should read. Last, we set a payload of at most 280 bytes so that FLAT can accommodate data and cryptographic elements typical of an IoT secure communication protocol.

4. Development

We developed a prototype of FLAT. The prototype comprises three modules: Client, SP, and IdP. Both our Client and SP modules are efficient enough to be run on top of resource-

constrained IoT devices like Arduino. The IdP, conversely, is supposed to be hosted on a server-like device. This is so because we envision the IdP being the single identity provider for all IoT devices in a given domain.

```

if (state == 0) // sends key request to IdP
    if (run_state_0() == EXIT_SUCCESS)
        state = 1;
else if (state == 1) // process key response from IdP
    if (run_state_1() == EXIT_SUCCESS)
        state = 2;
else if (state == 2) // sends assert to IdP
    if (run_state_2() == EXIT_SUCCESS)
        state = 3;
else if (state == 3) // process assert response from IdP
    if (run_state_3() == EXIT_SUCCESS)
        state = 4;
else if (state == 4) // send service request to SP
    if (run_state_4() == EXIT_SUCCESS)
        state = 5;
else if (state == 5) // process service response from SP
    if (run_state_5() == EXIT_SUCCESS)
        state = 0;

```

Figure 4: Client’s finite-state machine-like implementation.

Broadly, our development involved four stages: (i) the development of the communication functions, (ii) the implementation of the cryptosystems, (iii) the integration of the communication functions and the cryptosystems, (iv) the port of the integration to Arduino.

4.1. Communication

The communication functions have been developed using the Arduino Wi-Fi library in C/C++. When an Arduino Wi-Fi Shield is attached to an Arduino board, this library makes message transmission using Wi-Fi (802.11b/g) possible. The library provides both UDP and TCP protocols. FLAT was built on top of UDP and was developed based on the idea of a finite-state machine (Fig. 4) going through each step of the authentication protocol. If the synchronization between any pair of interlocutors is lost during the protocol, the process can be restarted through the transmission of a special message.

4.2. Cryptography

All asymmetric cryptosystems used in FLAT are based on Elliptic Curve Cryptography (ECC), as they incur less computation, communication, and storage overhead and therefore suit better the IoT world Stallings (2016). Precisely, we use Elliptic Curve Digital Signatures (ECDSA) and the Elliptic Curve Integrated Encryption Scheme (ECIES) as opposed to their traditional counterparts DSA and RSA Stallings (2016).

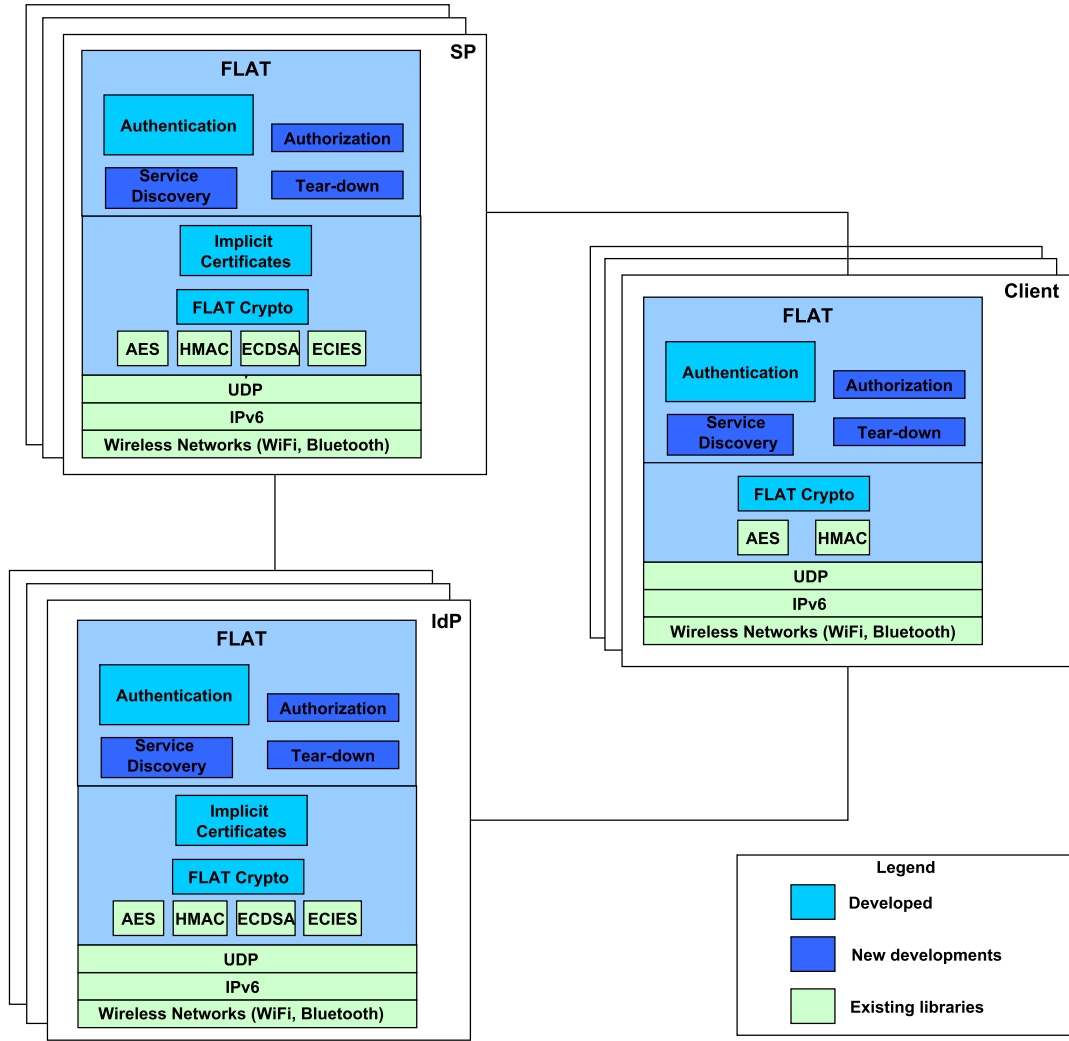


Figure 5: FLAT architecture.

We used RELIC⁶ as an underlying cryptographic library. On top of RELIC, we implemented higher-level cryptosystems like Implicit Certificates. Besides, the toolkit already implemented more common cryptosystems needed in FLAT like ECDSA, ECIES, AES, and HMAC. Respectively, FLAT employs these cryptosystems for digitally signing, asymmetrically encrypting, symmetrically encrypting, and message authentication.

4.3. Security level

Our development of FLAT offers, approximately, a 128-bit security level. In the symmetric part, we have AES using 128-bit keys and HMAC using the hash SHA-256 as a cryptographic hash function. In the asymmetric part—here, all based on ECC—we have the BN-254 elliptic

⁶<https://github.com/relic-toolkit>

curve as the foundation of the cryptosystems⁷.

4.4. Architecture

Fig. 5 shows the architecture of FLAT, *i.e.*, the modules (Client, SP, and IdP) and submodules (communication and cryptography) that comprise our solution. On the Client module, there are only symmetric cryptosystems (AES and HMAC). On the SP and IdP modules, besides the symmetric components, there are also asymmetric cryptosystems (Implicit Certificates, ECDSA, and ECIES).



Figure 6: FLAT demo: access (a) granted and (b) denied.

4.5. Demonstration

We developed a demo to showcase our FLAT prototype by demonstrating the functionalities of our architecture and executing operations on multiple classes of devices. Our demo covers the authentication part of FLAT applied to a toy cashless tollway similar to that described in Section 2. We say toy because we run FLAT over a mock-up tollway comprised of a toy car and toy gate. (Fig. 6).

In our demo, a car (client) runs FLAT and tries to authenticate to a toll gate (SP) by using an assertion supplied by its own domain. If the authentication is successful, a green LED lights, the toll gate opens, and the car is granted access to the tollway. Otherwise, red LED lights, the gate keeps closed, and the car’s access to the tollway is denied.

The Client, SP, and IdP have been respectively instantiated by an Arduino Due (84 MHz Atmel, 96 KB SRAM, 512 KB flash), a Raspberry Pi (Raspberry Pi zero W ARM1176JZF-S 1GHz single-core, 512MB RAM), and a Laptop Dell Inspiron (Intel Core i7 2.7 GHz, 8 GB

⁷Until recently, this curve was thought to offer a 128-bit security level; it offers a 100-bit, though Barbulescu and Duquesne (2017). The curve can be changed to a more secure one.

RAM). They correspond to our vision that roles in FLAT will be played by heterogeneous devices (c.f. Section 3.1). For communication, the Arduino board is equipped with an Arduino Wi-Fi shield (802.11b/g), and a MikroTik router plays the role of an access point. For controlling the toll gate, we used a Tower Pro servo motor 9G SG90. Last, we employ some LED bulbs to signal access granted or denied.

5. Results

We have evaluated FLAT authentication protocol in terms of computation, communication, protocol total run-time, SRAM and storage, and scalability. For simplicity, we will refer to the authentication protocol as FLAT in this section. Here, our performance numbers correspond to the average value of 100 runs.

Unless otherwise noticed, the numbers for the Client, SP, and IdP in this section refer to numbers measured by running FLAT on top of the demo hardware configuration, namely: Arduino Due, Raspberry Pi, and Laptop Dell Inspiron – c.f. Section 4.5 for details.

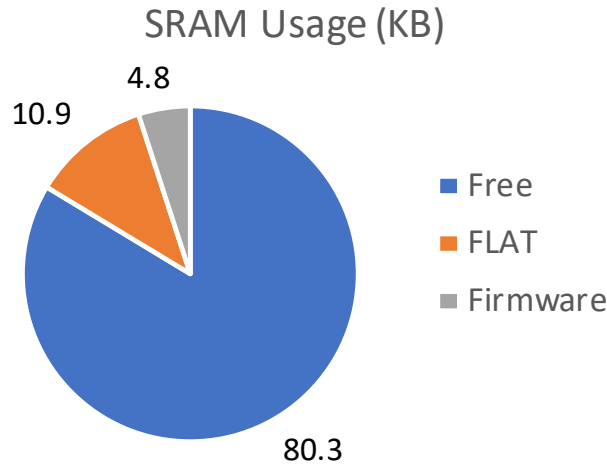


Figure 7: SRAM costs.

5.1. RAM & Storage

We make use of another protocol as a starting point for some comparisons and call it *baseline*. The baseline is similar to a traditional FIdM protocol (Fig. 1) and does not use Implicit Certificates as FLAT does. For fairness, however, we instantiate baseline using the same parameters as FLAT: certificate fields, level of security, message format, ECC-based cryptosystems, and so forth. Baseline certificate size is 134 bytes, as opposed to the Implicit Certificate size used in FLAT which is 70 bytes. This is so because baseline certificate includes, besides the identification information (in our case, 37 bytes), the public key (32 bytes) and the Certificate Authority signature (65 bytes). On the other hand, Implicit Certificates require only the identification information (37 bytes) and the public key extraction data (33 bytes).

We have evaluated FLAT Client requirements regarding SRAM (Fig. 7) and storage. The results show that FLAT, including the needed Arduino libraries to run the protocol, takes

around 16% of the Client’s total memory (Arduino, 15.7 KB out of 96 KB of SRAM). As for storage, FLAT requires 125.8 KB, *i.e.*, around 24% of the Client’s total storage (Arduino, 512 KB of flash), approximately. Notably, Arduino’s native libraries take 73.5 KB, FLAT communication functions takes 6.8 KB, and cryptography takes 45.5 KB.

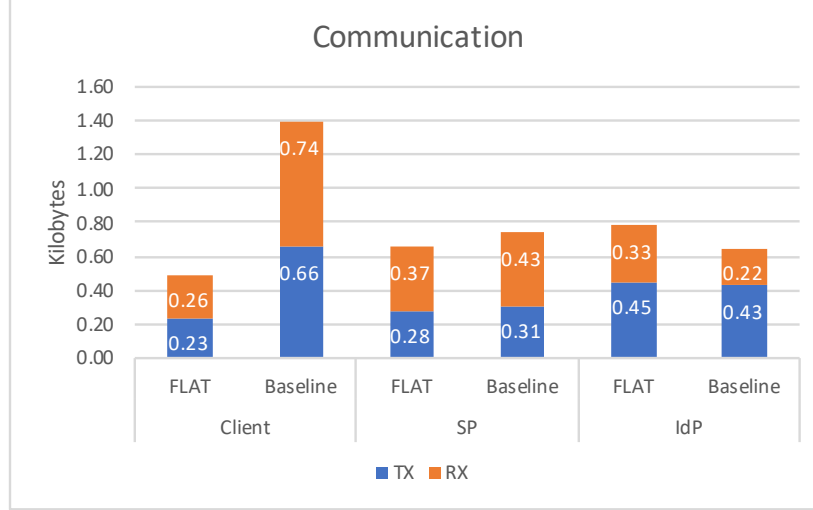


Figure 8: Communication costs.

5.2. Communication

Fig. 8 shows the amount of data sent (Tx) and received (Rx) by the Client, SP, and IdP in both FLAT and the baseline solution. In total, FLAT exchanges around 500 bytes on the Client side. It is worth pointing out that the FLAT Client is around 65% more efficient regarding Tx, Rx, and total data exchanged. Likewise, FLAT, as a whole, is 31% more efficient than the baseline regarding data exchange (Tx plus Rx). This lower communication overhead on the Client side is mainly because we use symmetric cryptosystems – they do not require certificate exchange. FLAT SP is around 9%, 14%, and 12%, respectively, more efficient than baseline SP, too. These light Client and SP, however, come to a price. To alleviate the Client and SP, FLAT demands more from the IdP. And then, in FLAT, even using (shorter) Implicit Certificates, the IdP total message exchange is around 20% higher than in the baseline. We claim, however, the IdP is a powerful device and, therefore, able to handle the total load.

5.3. Computation

FLAT Client is around 520 times faster than baseline (Fig. 9, Client). There are two main reasons for that. First, FLAT runs solely symmetric cryptosystems on the Client side. In the baseline, on the other hand, the Client needs to compute the following asymmetric operations: ECIES encryption (1 time), ECIES decryption (1), ECDSA signature (2) and ECDSA signature verification (5). And while symmetric operations are known to be negligible Perrig et al. (2002)—even in resource-constrained devices—, asymmetric operations—even ECC-based—are known to be inefficient Malan et al. (2008). Second, the Client delegates several tasks to the SP and IdP. A disadvantage of the second reason, however, is that the FLAT SP and IdP

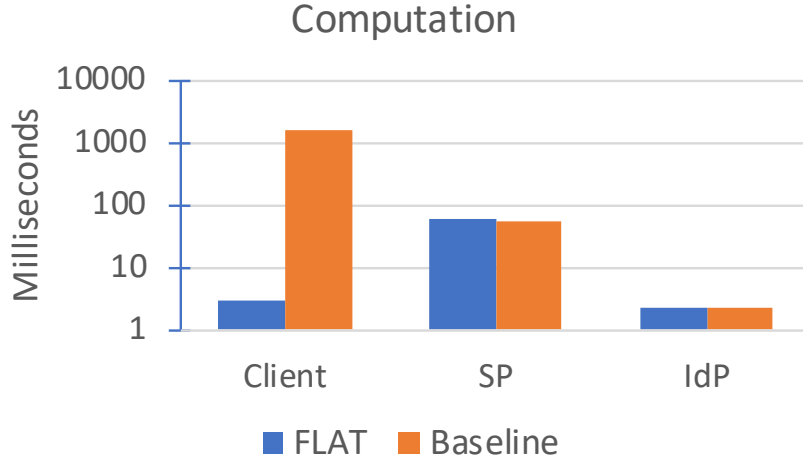


Figure 9: Computation costs.

end up being 4% and 2% less computationally efficient than baseline, respectively. But this is a very small price to pay compared to the gains obtained on other fronts.

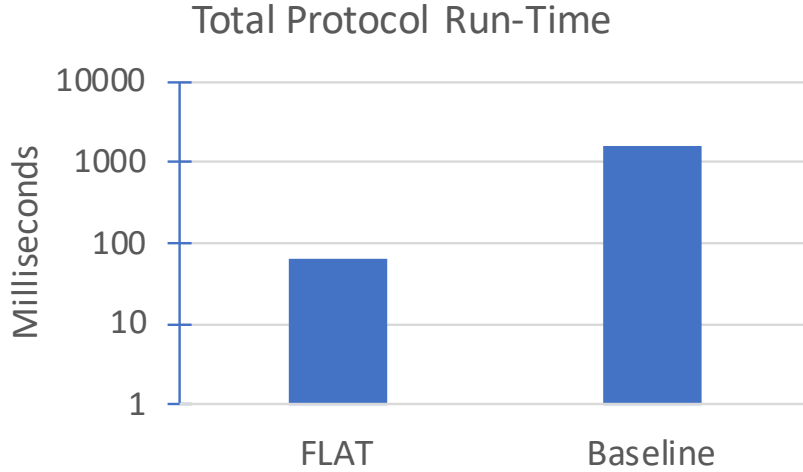


Figure 10: Total protocol run-time.

5.4. Total Run-time

We have measured the total amount of time that takes to run FLAT—*i.e.*, around 65 ms (Fig. 10)—and compared that to the baseline solution. Our results show that FLAT total run-time is, on average, only a fraction (around 4%) of the baseline total run-time (Fig. 10). This is a direct consequence of the computation (Section 5.3) and communication (Section 5.2) numbers previously shown, *i.e.*, FLAT exchanges fewer and smaller messages as well as requires significantly less computational work from the Client.

Notably, the fact that FLAT is computationally more efficient in the slow part of the protocol (Client – Section 5.3) — and only slightly less (4% and 2%) in the fast part (SP and IdP, respectively) — helps in improving its total run-time speedup compared to the baseline solution.

5.5. Scalability

Besides, FLAT scales smoothly regarding the number of Clients. If the number of Clients gets too large for a given IdP, we only need to create one more IdP—certified by the same Certificate Authority—and redistribute the Clients among them, evenly. Of course, this could be done in any federation. However, in FLAT, this process is very simple, as we can adopt a limit on the number of clients that an IdP can attend and, if such limit is reached, another IdP would be installed. New clients would be communicating with this new IdP, without any changes to the previous configuration.

6. Related Work

FIdM is already a consolidated solution in the traditional Internet, with several standards and protocols being widely used, such as Shibboleth, OpenID and OAuth. Several works address the authentication and authorization issue for the traditional Internet (e.g. Steiner et al. (1988); Shim et al. (2005); Maler and Reed (2008); Chadwick (2009); Chadwick and Inman (2009); Birrell and Schneider (2013); Isaakidis et al. (2016); Miettinen et al. (2016); Zuo et al. (2017)).

However, considering the IoT scenario, there is still no consensus on how to provide authentication and authorization to users and devices. In this sense, different IdM solutions for IoT have been proposed (Wang and Wang (2014); Xi et al. (2016); Neto et al. (2016); Kim et al. (2017a); Li et al. (2018)).

Wang et al. Wang and Wang (2014) propose a two-factor authentication scheme for wireless sensor networks that can provide privacy protection to users. They analyze case studies to show that previous approaches to the privacy issue in wireless sensor networks are not able to fully address the problem. They investigate possible solutions, concluding that public key cryptography is a suitable approach.

Xi et al. Xi et al. (2016) present a key agreement and authentication protocol for IoT devices named TDS (The Dancing Signals). The designed solution provides faster key generation when compared to other solutions, and is resistant to predictable channel attacks. TDS is modeled to only provide keys to physically close devices.

Neto et al. Neto et al. (2016) propose an authentication and access control solution for each step in the life-cycle of IoT devices. Using attribute and identity-based cryptography, they developed several protocols to enable Attribute-based access control (ABAC) and authentication in IoT environments. The proposed solution can be run in both powerful and constrained devices.

Hammi et al. Hammi et al. (2018) present an authentication solution for IoT named "Bubbles of Trust". Their approach is to provide authentication using a decentralized system based on blockchain in order to ensure availability and integrity. Their solution is evaluated in an implementation using Ethereum blockchain.

Zeng et al. Zeng et al. (2018) propose a multi-server approach to provide anonymous user authentication for IoT devices. They also performed a security evaluation of the solution.

Kumari et al. Kumari et al. (2018) present an ECC-based authentication protocol targeted to IoT and cloud servers. Their solution is evaluated in terms of performance and its security is formally analyzed using the AVISPA tool.

Ometov et al. Ometov et al. (2019) provides a comparison between different multi-factor authentication solutions for IoT, listing its advantages and disadvantages.

Some authentication and authorization proposals for IoT are based on (F)IdM solutions for the traditional Internet Liu et al. (2012); Fremantle et al. (2014); Cirani et al. (2015); Fremantle and Aziz (2016); Domenech et al. (2016)). The main advantage of such solutions is that they allow integration with traditional FIdM systems while trying to reduce the costs of these solutions by changing some aspects of the original approach.

Liu et al. Liu et al. (2012) design an architecture for authentication and authorization in IoT. The authors briefly discuss existing IdM approaches and opt for ECC-based PKC with OpenID for the authentication and RBAC for authorization. The authors also provide a brief security analysis of the proposed solution.

Fremantle et al. Fremantle et al. (2014) propose a federated identity model to provide access control to IoT devices, evaluating the possibility of an authentication and authorization model for IoT based on OAuth 2.0 with the MQTT protocol. In another work, Fremantle et al. (Fremantle and Aziz, 2016) propose an extension to the OAuth 2.0 protocol to provide a FIdM solution for IoT devices, as well as enhance the privacy of users. Their solution is called OAuthing and enables users and devices to be registered automatically, providing each user/device anonymous identities.

Cirani et al. Cirani et al. (2015) propose an authorization architecture for IoT using an external authorization service based on OAuth, called IoT-OAS. Their architecture transfers the authorization logic from the constrained device to the external service, allowing the device or smart object to protect its resources by only making a request to the authorization framework.

Domenech et al. Domenech et al. (2016) describe an authentication and authorization model for IoT based on SAML and XACML standards, the AAI4WoT. The authors provide a prototype as well as an evaluation in terms of storage, communication, CPU usage, and power consumption.

These works develop solutions based on traditional FIdM approaches. FLAT, however, is not based on existing FIdM solutions for the traditional Internet, being exclusively tailored to IoT.

There are solutions which approach the authentication and authorization issue in IoT through asymmetric cryptography B. Oliveira et al. (2009); Yavuz (2013); Porambage et al. (2014).

Oliveira et al. B. Oliveira et al. (2009) propose a solution to provide end-to-end authentication between a sensor node and multiple users. They claim that given the number of applications that a sensor can be connected, the use of digital signatures is the most suitable option to provide end-to-end authentication. They evaluated both ECDSA and BLS signature algorithms in constrained devices, showing the feasibility of the proposed solution.

Yavuz Yavuz (2013) proposes a new signature scheme for authentication of resource-constrained devices called ETA (Efficient and Tiny Authentication). The author claims that traditional signature schemes are not suitable for constrained devices as they require high computational power and energy consumption. ETA also provides smaller signatures and key sizes than other existing similar solutions.

Table 2: Related work comparison.

Work	Symmetric Crypto on Client	Gateway	Client device	Context	Authorization	Authentication
Oliveira et al.	—	—	Constrained	General	—	Yes
Horrow et al. 2012	—	Yes	Constrained	General	Yes	Yes
Liu et al. 2012	—	Yes	—	General	Yes	Yes
Hummen et al. 2013	—	Yes	Constrained	General	—	Yes
Yavuz 2013	—	—	Constrained	General	—	Yes
Porambage 2014	—	—	Constrained	General	—	Yes
Fremantle et al. 2014	—	Yes	Constrained	General	Yes	Yes
Wang et al. 2014	—	—	Constrained	General	—	Yes
Markmann et al. 2015	—	Yes	Constrained	General	—	Yes
Cirani et al. 2015	—	Yes	Constrained	General	Yes	—
Witkowski et al. 2015	Yes	Yes	Constrained	Smart home	Yes	Yes
Fremantle et al. 2016	—	Yes	Constrained	General	Yes	Yes
Domenech et al. 2016	—	Yes	Intermediary	General	Yes	Yes
Hong et al. 2016	—	Yes	Constrained	Smart home	Yes	—
Xi et al. 2016	—	Yes	Constrained	Sensors	—	Yes
Neto et al. 2016	—	—	Constrained	General	Yes	Yes
FLAT	Yes	—	Constrained	General	—	Yes

Porambage et al. Porambage et al. (2014) also address the end-to-end authentication between users and sensors in IoT. The authors present an authentication protocol that takes into consideration the heterogeneity of the network. Their approach uses asymmetric cryptography for authentication through the use of implicit certificates, in order to make the solution more suitable for resource-constrained devices.

These solutions, however, are based on asymmetric cryptography. FLAT, on the other hand, considers an IoT-tailored protocol for authentication, that makes use of only symmetric cryptographic primitives in the Client device.

Other works try to transfer the authentication and authorization tasks to an external gateway or to a more powerful node in the network, in order to preserve the resource-constrained IoT devices Horrow and Sardana (2012); Hummen et al. (2013); Turkanović et al. (2014); Markmann et al. (2015); Kim et al. (2017b).

Horrow and Sardana Horrow and Sardana (2012) present the requirements for an IdM system for IoT and propose an IdM framework for IoT using cloud-based technologies. Instead of having computing nodes to process the information sent by sensors, the computation is performed in the cloud. The authors present a general architecture to approach the identity management issue in IoT, but they do not describe in details how would be the protocols to implement this architecture.

Hummen et al. Hummen et al. (2013) propose some optimizations to DTLS in order to use certificates in IoT devices, such as pre-validation of certificates in a gateway and a delegation procedure that allows another entity to validate the certificate instead of the constrained device. However, they present preliminary results and only evaluate the computational overhead, not addressing other aspects of the solution.

In a short work, Markmann et al. Markmann et al. (2015) approach the authentication for resource-constrained devices in IoT using Identity-Based Cryptography (IBC) and ECC. They assume the presence of a node with more computational power in each domain in order to provide the end-to-end authentication utilizing IBC. This node works as a trusted authority (similarly to a CA), and a federation of these trusted authorities is required in order to provide authentication between different domains.

The solutions proposed by Horrow and Sardana, Hummen et al., and Markmann et al. make use of another entity to perform authentication and authorization tasks. FLAT, however, do not transfer the computation to an external entity, developing a protocol that can be executed even by restricted devices.

There are also works that address FIdM solutions for specific IoT scenarios He and Zeadally (2015); Witkovski et al. (2015); Hong et al. (2016).

Witkovski et al. Witkovski et al. (2015) propose a solution for IoT device authentication based on session keys and symmetric cryptography with the use of a gateway to integrate the traditional Internet and IoT in the maintenance of electronic devices in a smart home scenario.

Hong et al. Hong et al. (2016) claim that Bluetooth Low Energy (BLE) can be used as an access control alternative in smart homes. A prototype of the BLE access control solution was built on top of resource-constrained devices and applied to specific smart home scenarios.

Differently from these approaches (targeted to very specific scenarios), FLAT is a broad solution that can be applied to different IoT scenarios.

Several works propose FIdM solutions for IoT. These solutions, however, do not address completely the IoT IdM issues, as they are based on IdM solutions for the traditional Internet, rely on asymmetric cryptography or make use of external entities to perform computational tasks. FLAT is a federated authentication protocol that can be used to implement a FIdM system even in restricted devices, as it is based on only symmetric cryptographic primitives in the Client side.

7. Conclusion

As IoT gains more visibility and IoT solutions are being incorporated in the most diverse fields, there is a real need to ensure the security of the developed IoT applications and preserve the privacy of their users.

In this work, we proposed a federated identity authentication protocol exclusively tailored to IoT called FLAT. FLAT considers that Clients will run over devices of low computational capabilities and, thus, requires only symmetric (light) cryptographic operations from them. Besides, FLAT employs Implicit Certificates rather than regular certificates in the IdP \leftrightarrow SP communication for saving both computation and bandwidth. FLAT also replaces inefficient cryptosystems like RSA/DSA by others more suitable to IoT and reduces the message load in the Client device.

This paper presented the assumptions necessary to the correct operation of FLAT, a prototype of the proposed solution applied to a cashless toll system and an evaluation of FLAT in terms of storage, SRAM, communication, computation, and total run-time.

FLAT computation in the Client is 520 times more efficient than the baseline solution. FLAT uses around 16% of Client's SRAM and 24% of Client's flash memory when running in an Arduino Due device. Moreover, FLAT total run-time represents 4% of the total time necessary to run Baseline. Regarding the communication costs, FLAT is also 31% more efficient than the Baseline protocol when considering the total data exchange. Our results confirm that FLAT can run efficiently on top of IoT devices.

FLAT is a step towards the development of a complete IdM system that could take care of all the lifecycle of an IoT device, as it provides an authentication solution that is lightweight, contemplates the mobility aspect of devices considering different security domains and has an operational prototype.

As future works, we consider the development of a complete IdM system. By now, FLAT is an authentication solution for IoT. The next steps are completing the IdM solution with the implementations of the authorization, service discovery process, and tear-down.

Another future work includes the proposal and evaluation of different communication possibilities for the Client device. The Client could use the SP as an intermediate to communicate with the IdP, using SP's permanent Internet connection and enabling the Client to only have a low power Wi-Fi connection. In another possible scenario, the Client will not need a network connection or to communicate directly with the IdP, with the SP resending the messages between Client and IdP.

References

- Atzori, L., Iera, A., Morabito, G., 2010. The internet of things: A survey. *Computer networks* 54, 2787–2805.
- B. Oliveira, L., Kansal, A., Priyantha, B., Goraczko, M., Zhao, F., 2009. Secure-TWS: Authenticating Node to Multi-user Communication in Shared Sensor Networks, in: *ACM International Conference on Information Processing in Sensor Networks (IPSN'09)*, ACM. pp. 289–300.
- Barbulescu, R., Duquesne, S., 2017. Updating Key Size Estimations for Pairings. *Journal of Cryptology* , 1–39.
- Birrell, E., Schneider, F.B., 2013. Federated Identity Management Systems: A Privacy-based Characterization. *IEEE Security & Privacy* 11, 36–48.
- Borgia, E., 2014. The internet of things vision: Key features, applications and open issues. *Computer Communications* 54, 1–31.
- Brown, D.R., Campagna, M.J., Vanstone, S.A., 2009. Security of ecqv-certified ecdsa against passive adversaries. *IACR Cryptology ePrint Archive* 2009, 620.
- Brown, D.R.L., Gallant, R.P., Vanstone, S.A., 2002. Provably Secure Implicit Certificate Schemes, in: *International Conference on Financial Cryptography (FC'02)*, pp. 156–165.
- Chadwick, D., 2009. Federated Identity Management. *Foundations of Security Analysis and Design V* , 96–120.
- Chadwick, D.W., Inman, G., 2009. Attribute Aggregation in Federated Identity Management. *IEEE Computer* 42, 33–40.
- Cirani, S., Picone, M., Gonizzi, P., Veltri, L., Ferrari, G., 2015. IoT-OAS: An OAuth-based Authorization Service Architecture for Secure Services in IoT Scenarios. *IEEE Sensors Journal* 15, 1224–1234.
- Domenech, M.C., Boukerche, A., Wangham, M.S., 2016. An authentication and Authorization Infrastructure for the Web of Things, in: *ACM Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet'16)*, pp. 39–46.

- Fremantle, P., Aziz, B., 2016. OAuthing: Privacy-enhancing Federation for the Internet of Things, in: Cloudification of the Internet of Things (CIoT'16), pp. 1–6.
- Fremantle, P., Aziz, B., Kopecký, J., Scott, P., 2014. Federated Identity and Access Management for the Internet of Things, in: International Workshop on Secure Internet of Things (SIoT'14), pp. 10–17.
- Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M., 2013. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems* 29, 1645–1660.
- Hammi, M.T., Hammi, B., Bellot, P., Serhrouchni, A., 2018. Bubbles of trust: A decentralized blockchain-based authentication system for iot. *Computers & Security* 78, 126 – 142.
- He, D., Zeadally, S., 2015. An Analysis of RFID Authentication Schemes for Internet of Things in Healthcare Environment Using Elliptic Curve Cryptography. *IEEE Internet of Things Journal* 2, 72–83.
- Hong, J., Levy, A., Levis, P., 2016. Demo: Building Comprehensible Access Control for the Internet of Things Using Beetle, in: ACM International Conference on Mobile Systems, Applications, and Services (MobiSys'16).
- Horror, S., Sardana, A., 2012. Identity Management Framework for Cloud Based Internet of Things, in: International Conference on Security of Internet of Things (SecurIT'12), pp. 200–203.
- Hummen, R., Ziegeldorf, J.H., Shafagh, H., Raza, S., Wehrle, K., 2013. Towards Viable Certificate-based Authentication for the Internet of Things, in: Workshop on Hot Topics on Wireless Network Security and Privacy (HotWiSec'13), ACM. pp. 37–42.
- Isaakidis, M., Halpin, H., Danezis, G., 2016. UnlimitID: Privacy-preserving Federated Identity Management Using Algebraic MACs, in: Workshop on Privacy in the Electronic Society (WPES'16), ACM. pp. 139–142.
- Kim, H., Kang, E., Lee, E.A., Broman, D., 2017a. A Toolkit for Construction of Authorization Service Infrastructure for the Internet of Things, in: ACM/IEEE International Conference on Internet-of-Things Design and Implementation (IoTDI'17), ACM/IEEE. pp. 147–158.
- Kim, J.Y., Hu, W., Sarkar, D., Jha, S., 2017b. ESIoT: Enabling Secure Management of the Internet of Things, in: ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec'17), ACM. pp. 219–229.
- Kumari, S., Karuppiah, M., Das, A.K., Li, X., Wu, F., Kumar, N., 2018. A secure authentication scheme based on elliptic curve cryptography for iot and cloud servers. *The Journal of Supercomputing* 74, 6428–6453.
- Li, X., Niu, J., Kumari, S., Wu, F., Sangaiah, A.K., Choo, K.K.R., 2018. A Three-factor Anonymous Authentication Scheme for Wireless Sensor Networks in Internet of Things Environments. *Journal of Network and Computer Applications* 103, 194–204.

- Liu, J., Xiao, Y., Chen, C.P., 2012. Authentication and Access Control in the Internet of Things, in: International Conference on Distributed Computing Systems Workshops (ICDCSW), IEEE. pp. 588–592.
- Malan, D.J., Welsh, M., Smith, M.D., 2008. Implementing Public-Key Infrastructure for Sensor Networks. *ACM Transactions on Sensor Networks* 4, 22:1–22:23.
- Maler, E., Reed, D., 2008. The Venn of Identity: Options and Issues in Federated Identity Management. *IEEE Security & Privacy* 6, 16–23.
- Markmann, T., Schmidt, T.C., Wählisch, M., 2015. Federated End-to-end Authentication for the Constrained Internet of Things Using IBC and ECC. *ACM SIGCOMM Computer Communication Review* 45, 603–604.
- Miettinen, M., Huang, J., Nguyen, T.D., Asokan, N., Sadeghi, A.R., 2016. POSTER: Friend or Foe? Context Authentication for Trust Domain Separation in IoT Environments, in: ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec’16), ACM. pp. 225–226.
- Neto, A.L.M., Souza, A.L., Cunha, I., Nogueira, M., Nunes, I.O., Cotta, L., Gentile, N., Loureiro, A.A., Aranha, D.F., Patil, H.K., et al., 2016. AoT: Authentication and Access Control for the Entire IoT Device Life-cycle, in: ACM Conference on Embedded Network Sensor Systems (Sensys’16), ACM. pp. 1–15.
- Ometov, A., Petrov, V., Bezzateev, S., Andreev, S., Koucheryavy, Y., Gerla, M., 2019. Challenges of multi-factor authentication for securing advanced iot applications. *IEEE Network* 33, 82–88.
- Perrig, A., Szewczyk, R., Wen, V., Culler, D., Tygar, J.D., 2002. SPINS: Security Protocols for Sensor Networks. *Wireless Networks* 8, 521–534.
- Porambage, P., Schmitt, C., Kumar, P., Gurtov, A., Ylianttila, M., 2014. Two-phase Authentication Protocol for Wireless Sensor Networks in Distributed IoT Applications, in: IEEE Wireless Communications and Networking Conference (WCNC’14), IEEE. pp. 2728–2733.
- Santos, M.L.B.A., Carneiro, J.C., Teixeira, F.A., Franco, A.M.R., Henriques, M.A.A., Oliveira, L.B., 2018. Federated authentication of things: Demo abstract, in: Proceedings of the 17th ACM/IEEE International Conference on Information Processing in Sensor Networks, IEEE Press. pp. 136–137.
- Shim, S.S.Y., Bhalla, G., Pendyala, V., 2005. Federated Identity Management. *IEEE Computer* 38, 120–122.
- Stallings, W., 2016. *Cryptography and Network Security: Principles and Practice*. Pearson.
- Steiner, J.G., Neuman, B.C., Schiller, J.I., 1988. Kerberos: An Authentication Service for Open Network Systems, in: *USENIX Winter*, USENIX. pp. 191–202.
- Suh, G.E., Devadas, S., 2007. Physical Unclonable Functions for Device Authentication and Secret Key Generation, in: ACM/IEEE Design Automation Conference (DAC’07), ACM/IEEE. IEEE. pp. 9–14.

- Turkanović, M., Brumen, B., Hölbl, M., 2014. A Novel User Authentication and Key Agreement Scheme for Heterogeneous Ad Hoc Wireless Sensor Networks, Based on the Internet of Things Notion. *Ad Hoc Networks* 20, 96–112.
- Ververidis, C.N., Polyzos, G.C., 2008. Service Discovery for Mobile Ad Hoc Networks: a Survey of Issues and Techniques. *IEEE Communications Surveys & Tutorials* 10, 30–45.
- Wang, D., Wang, P., 2014. On the Anonymity of Two-factor Authentication Schemes for Wireless Sensor Networks: Attacks, Principle and Solutions. *Computer Networks* 73, 41–57.
- Witkovski, A., Santin, A., Abreu, V., Marynowski, J., 2015. An IdM and Key-based Authentication Method for Providing Single Sign-on in IoT, in: *IEEE Global Communications Conference (GLOBECOM'15)*, IEEE. pp. 1–6.
- Xi, W., Qian, C., Han, J., Zhao, K., Zhong, S., Li, X.Y., Zhao, J., 2016. Instant and Robust Authentication and Key Agreement Among Mobile Devices, in: *ACM Conference on Computer and Communications Security (CCS'16)*, ACM. pp. 616–627.
- Yavuz, A.A., 2013. ETA: Efficient and Tiny and Authentication for Heterogeneous Wireless Systems, in: *ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec'13)*, ACM. pp. 67–72.
- Zeng, X., Xu, G., Zheng, X., Xiang, Y., Zhou, W., 2018. E-aue: An efficient anonymous user authentication protocol for mobile iot. *IEEE Internet of Things Journal* 6, 1506–1519.
- Zuo, C., Zhao, Q., Lin, Z., 2017. Authscope: Towards Automatic Discovery of Vulnerable Authorizations in Online Services, in: *ACM Conference on Computer and Communications Security (CCS'17)*, ACM. pp. 799–813.