Privacy-Preserving Delegable Authentication in the Internet of Things

Clémentine Gritti
NTNU, Trondheim, Norway
clementine.gritti@ntnu.no

Melek Önen, Refik Molva
EURECOM, Sophia Antipolis, France
[melek.onen,refik.molva]@eurecom.fr

ABSTRACT

The expanding Internet of Things (IoT) technology offers the ease of communication with and access to multiple services for companies and individuals. However, because of the limited trustworthiness set on smart devices, as well as the ever-increasing amount of them, challenges for security and privacy protection have been growing. In this paper, we propose a new authentication solution that enables a smart device to securely connect to services, based on attribute-based credentials. Our solution allows IoT devices to authenticate to various services in an efficient way, without compromising their privacy. Indeed, during the authentication of an IoT device to a particular service, a new credential is generated such that only relevant attributes are disclosed to the actual service. Moreover, this operation is delegated to a gateway in order to relieve the workload at devices' side.

CCS CONCEPTS

• **Security and privacy → Mobile and wireless security**; • **Networks → Mobile and wireless security**;

KEYWORDS

IoT authentication, privacy, attribute-based credentials

ACM Reference Format:
Clémentine Gritti and Melek Önen, Refik Molva. 2019. Privacy-Preserving Delegable Authentication in the Internet of Things. In *Proceedings of The 34th ACM/SIGAPP Symposium on Applied Computing, Limassol, Cyprus, April 8–12, 2019 (SAC '19),* 9 pages.
https://doi.org/10.1145/3297280.3297365

1 INTRODUCTION

The continuously growing Internet of Things (IoT) technology offers companies and individuals many opportunities for new services and functionalities. While this technology eases the interconnection between individuals and devices, the high number of devices and their hetrogeneity raise several problems in terms of security and privacy. Successful authentication of devices to services hence becomes mandatory as it enables the latter to ensure genuineness of devices, before establishing communication among the intended parties. Existing solutions for authentication include Attribute-Based Credentials (ABC) technology [5, 27] that permits flexible and privacy-preserving authentication of devices within IoT environments.

Unfortunately, an attribute-based authentication mechanism usually generates a credential validating a set of several attributes that may not always be required by the services. Hence, credentials may bring the devices to disclose unnecessary and sometimes sensitive attributes. Data minimization[1] states that data exposition should be limited to what is required to successfully accomplish a given purpose and should claim further consent upstream for any re-purposing. This principle has been enforced with the General Data Protection Regulation (GDPR)[2], that has been active since May 2018.

While devices could generate or construct different dedicated credentials for each service, making sure to disclose relevant attributes only, they may not be able to afford when the number of services is high. Indeed, devices have constrained resources in terms of memory, CPU and storage, while updating their attribute-based credentials in function of the services that they wish to reach could be computationally costly. Another option can simplify the authentication process by letting devices delegate the authentication and hence inherently the data minimization process on their credentials to an internal trusted party.

In this paper, we propose a privacy-preserving delegable authentication protocol[3] that uses attribute-based credentials. The protocol involves a third party, named as the gateway, that is in charge of discarding all sensitive attributes irrelevant to the actual service that the device wants to securely connect and authenticate to. The proposed solution consists of two phases: a full authentication of the device to the gateway, and a delegated authentication of the device to the service, run between the service and the gateway on behalf of the device. Attribute privacy (hence data minimization) is ensured from the service's view by discarding sensitive attributes for delegated authentication. First, the device authenticates to the gateway that links the former to various services. This internal phase requires an authentication proof generated with the full credential of the device, that possibly includes attributes with sensitive information not relevant for the services to be reached. Once the device has successfully authenticated to the gateway, it delegates the authentication operations to the gateway which, for each server, modifies the authentication proof in order to discard sensitive attributes. Such task is let to the gateway instead of the

[1]https://gdpr.report/news/2018/01/08/gdpr-important-data-minimisation/
[2]https://gdpr-info.eu/art-5-gdpr/
[3]Our protocol is similar to existing research papers using the incorrect "delegatable" term.

SAC '19, April 8–12, 2019, Limassol, Cyprus

ACM ISBN 978-1-4503-5933-7/19/04…$15.00
https://doi.org/10.1145/3297280.3297365

device itself due to the lack of computational resources. The remaining attributes embedded in the minimized proof do not leak sensitive information (other than the one the service can discover), but still guarantee successful authentication at service's stage.

In the next section, we illustrate the problem with an IoT use case and give more details about it. Section 3 presents the building blocks on which the construction will be based and describes the latter. In Section 4, we provide the security models and proofs of our construction, along with the performance analysis and the related work.

2 PROBLEM STATEMENT

2.1 Use Case Scenario

We illustrate the utility of our privacy-preserving delegable authentication protocol with a smart-home use case. Multiple smart devices compose a smart home, including thermostat, camera, lighting and clock. The devices connect to remote services to forward data to or obtain data from them. There is a gateway acting as a bridge between the smart devices and the services for delegated authentication.

A smart fridge (e.g., Samsung Family Hub refrigerator[4]) connects to a supermarket center for grocery orders. The fridge's owner has subscribed to the center and registered his/her fridge, such that information referring to subscription and registration is given inside the credential of the fridge. Let $At(sub)$ and $At(reg)$ be the attributes in the credential describing the subscription and registration states respectively. Before enabling grocery orders, the center will verify whether the fridge belongs to an owner with a valid subscription and has a valid registration.

In parallel, the smart fridge meets a technical problem. For instance, the fridge can no longer adjust and monitor temperature inside. Hence, connection to the maintenance center of the fridge's brand is required (e.g., Samsung Home Appliances Support[5]). In order to launch the repair of the temperature controller, the center will verify the serial number, model and year of purchase of the fridge. Let $At(snb)$, $At(mod)$ and $At(year)$ be the attributes in the credential representing the serial number, model and year of purchase respectively.

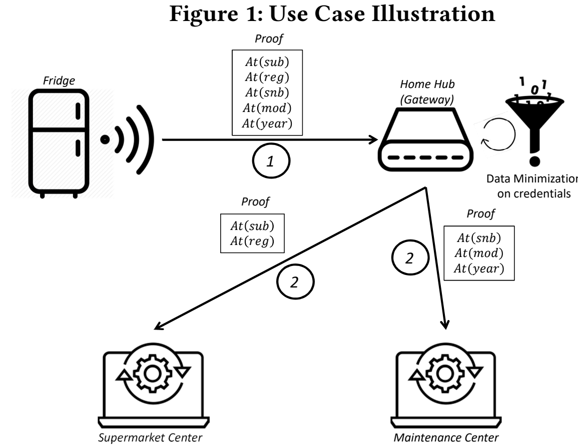
First, the fridge connects to the owner's hub (e.g., Samsung SmartThings Hub[6]), acting as the gateway, by being internally authenticated using its full credential. The gateway checks that the attributes in the fridge's credential satisfy the full authentication policy. Attributes to be verified include $At(sub)$ and $At(reg)$ along with $At(snb)$, $At(mod)$ and $At(year)$. Since the hub represents the owner, the policy may claim for extra attributes not necessarily relevant for services, in addition to the aforementioned attributes required for delegated authentications.

The fridge then connects to the supermarket center by being externally authenticated. The gateway has modified the authentication proof embedding the full credential by keeping the attributes $At(sub)$ and $At(reg)$ on valid subscription and registration respectively, and possibly other neutral ones, and by discarding the rest, including attributes $At(snb)$, $At(mod)$ and $At(year)$. The supermarket center checks that the attributes in the minimized proof satisfy its delegated authentication policy.

Meanwhile, the fridge connects to the maintenance center by being externally authenticated. The gateway has adapted the authentication proof embedding the full credential by only keeping the attributes $At(snb)$, $At(mod)$ and $At(year)$ describing the fridge's serial number, model and year of purchase respectively. Other attributes, such as attributes $At(sub)$ and $At(reg)$ on valid subscription and registration, are not required, and are thus discarded, remaining hidden from the maintenance center. The latter checks that the attributes in the minimized proof satisfy its delegated authentication policy.

As described above, the gateway adapts authentication proofs of the fridge in function of the targeted services. Attributes $At(sub)$ and $At(reg)$ used for authentication to the supermarket center are not used for authentication to the maintenance center, and vice versa for the attributes $At(snb)$, $At(mod)$ and $At(year)$. Attributes discarded for delegated authentication are not revealed to the corresponding center. In addition, workload of devices is reduced: the fridge authenticates to the gateway only once, while the latter undertakes two delegated authentications to services. Figure 1 illustrates the above use case.

Figure 1: Use Case Illustration



2.2 Environment

Four parties, namely the trusted third party TTP, the device D, the gateway G and the service S, participate in the privacy-preserving delegable authentication protocol. The trusted third party TTP assigns the attributes of the device D and generates the public and secret key material of the latter according to these attributes. TTP also delivers the public and secret key material to the gateway G and service S for full and delegated authentication processes respectively.

When the device D wishes to communicate with the service S, a delegated authentication is required. First, the device D internally authenticates to the gateway G, by generating a proof that its attributes fulfill the gateway's policy. Second, the device D delegates to the gateway G the data minimization process on its proof

[4]https://www.samsung.com/us/explore/family-hub-refrigerator/overview/
[5]https://www.samsung.com/support/home-appliance/refrigerators
[6]https://www.samsung.com/us/smart-home/smartthings/

such that the remaining (i.e. non-discarded) attributes satisfy the service's policy, and the device D externally authenticates to the service S. If the two steps are successful, then the device D can communicate with the service S. Attribute privacy is preserved in such way that no information is leaked about the attributes of the device D beyond the fact that they satisfy the authentication policies. Moreover, attributes that are not involved in the delegated authentication remain completely hidden from the service's view.

3 PRIVACY-PRESERVING DELEGABLE AUTHENTICATION

We aim to design a privacy-preserving delegable authentication protocol in IoT as follows. There are three steps, a first one for key material issuance to all participating parties, a second one for proof generation and full authentication, and a third one for proof minimization and delegated authentication.

Key Material Issuance. The trusted third party TTP issues the key material of the device D. Let $\mathcal{T} \subseteq \mathcal{U}$ be the set of attributes of the device, where $\mathcal{U}$ is the attribute universe. The key material of the device D secretly embeds all the attributes in $\mathcal{T}$. TTP also generates the key material of the gateway G and the service S, containing secret components in order to dedicate full and delegated authentication processes to these specific parties respectively.

Full Authentication. A device D requests to be authenticated by G according to an authentication policy $\mathcal{P}$. It gives a signature as its authentication proof to the gateway G. The gateway G checks the signature according to $\mathcal{P}$. If the signature is valid, then this means that the attributes in $\mathcal{T}$ satisfy the policy $\mathcal{P}$ and full authentication is successful at the gateway's stage.

Delegated Authentication. The gateway G modifies the signature of the device D regarding the subpolicy $\mathcal{P}' \subseteq \mathcal{P}$ defined by S. Informally, the gateway G discards some of the attributes in $\mathcal{T}$, obtaining a minimized signature containing only attributes in a subset $\mathcal{T}' \subseteq \mathcal{T}$. Attributes in $\mathcal{T}'$ will be revealed to S while attributes in $\mathcal{T} \setminus \mathcal{T}'$ will not be disclosed to S. The service S then checks the minimized signature according to the policy $\mathcal{P}'$. If the signature is valid, then this means that the attributes in $\mathcal{T}'$ satisfy the policy $\mathcal{P}'$ and delegated authentication is successful at the service's stage.

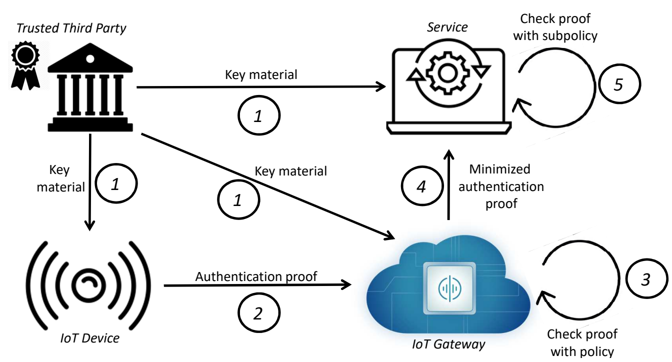
Figure 2 illustrates the attribute-based delegable authentication protocol where a trusted third party, an IoT device, an IoT gateway, and a service participate. The trusted third party delivers the key material to the other participating parties (1). The device generates its authentication proof for full authentication to the gateway (2). The latter checks it (3) and if the authentication is successful, then the gateway modifies the proof for delegated authentication to the service (4). The latter checks it (5) and if the authentication is successful, then the device can communicate with the service.

3.1 Building Blocks

3.1.1 *Attribute-Based Authentication.*

Device's Key Material. The secret key material of the device D is generated by TTP based on the weak Boneh-Boyen signature scheme [12, 13] such that the secret key component of D corresponds to a Boneh-Boyen signature.

Figure 2: Privacy-Preserving Delegable Authentication Protocol Overview



As defined in [12, 13], let $g, g_1, g_2, h_1, \cdots, h_k$ be elements in $\mathbb{G}$, a cyclic group of prime order p. Let $x \in \mathbb{Z}_p$ be the device's secret key and g^x be the corresponding public key. Let at be an attribute in $\mathbb{Z}_p^*$ and r, t be two random exponents in $\mathbb{Z}_p^*$. The signature on attribute at is set as (σ, r, t) where:

$$\sigma = (g_1 g_2^r h_1^{at})^{\frac{1}{x+t}},$$

and is valid if the equation $e(\sigma, g^x g^t) = e(g_1 g_2^r h_1^{at}, g)$ holds. Given k attributes $at_1, at_2, \cdots, at_k$, the signature on all these attributes is set as (σ, r, t) where:

$$\sigma = (g_1 g_2^r h_1^{at_1} \cdots h_k^{at_k})^{\frac{1}{x+t}},$$

and is valid if the equation $e(\sigma, g^x g^t) = e(g_1 g_2^r h_1^{at_1} \cdots h_k^{at_k}, g)$ holds.

Following the multiple-attribute case, the public elements h_i are raised to power the attributes of D, and then multiplied together. By doing so, only one key component is needed for all the attributes rather than one key component per attribute. By multiplying the attribute-based product with the randomized element g_2^r, attributes are ensured to be hidden in the secret key $sk_{\mathcal{T}} = (g_1 g_2^r h_1^{at_1} \cdots h_k^{at_k})^{\frac{1}{x+t}}$, where $\mathcal{T} = \{at_1, \cdots, at_k\}$.

Device's Authentication Proof. The device forwards an authentication proof to the gateway G. The proof aims to show that the attributes of D satisfy the full authentication policy. If the proof is valid, then the device D successfully authenticates to G.

The signing process is based on the hashing technique from Waters' signature scheme [28]. Given public elements $u_0, u_1, \cdots, u_l \in \mathbb{G}$ and an l-bit-length message $m = m_1 \cdots m_l$, the hashed message $h(m)$ is computed as $h(m) = u_0 \prod_{i \in [1,l]} u_i^{m_i} = u_0 \prod_{j \in M} u_j$ where $M = \{j \in [1, l]; m_j = 1\}$. Then, using its attribute-based secret key component $sk_{\mathcal{T}} = (g_1 g_2^r h_1^{at_1} \cdots h_k^{at_k})^{\frac{1}{x+t}}$ and the Waters hashed message $h(m)$, the device D generates the signature as its authentication proof. Moreover, in order to privatize the verification of the proof to the gateway G only, the device D raises part of the signature to the power of $H(e(pk_G, sk_2))$, where $H : \mathbb{G}_T \to \mathbb{Z}_p$ is a hash function, pk_G is the public key of G and sk_2 is a secret element of D. By doing so, the gateway G is forced to use its secret key sk_G

and the public key pk_D of the device to check the validity of the signature with its policy.

3.1.2 Delegated Authentication. The device D generates an authentication proof using its key material that embeds all its attributes. In order to successfully authenticate to G, the proof should contain the attributes that satisfy the authentication policy $\mathcal{P}$. Thereafter, in order to authenticate to S, the gateway G on behalf of the device D specifically selects which of the attributes are relevant and modifies the device's proof according to these attributes. The proof generated by D and the proof derived by G can be linked; however, the minimized proof forwarded to the service S does not leak the non-necessary attributes.

More precisely, the device D first generates its authentication proof under the form of a signature using its secret key material $sk_{\mathcal{T}}$ that embeds attributes in a set $\mathcal{T}$. The gateway G checks the signature based on an authentication policy $\mathcal{P}$. A valid signature means that the attributes in $\mathcal{T}$ satisfy the policy $\mathcal{P}$ and thus, the device D successfully authenticates to G. Once the full authentication is done, the gateway G can modify the signature such that some of the attributes that are not relevant for delegated authentication are discarded. Using its secret key material sk_G, the service's public key material pk_S and the device's public key material pk_D, the gateway G updates the signature according to a restricted policy $\mathcal{P}' \subseteq \mathcal{P}$ by computing:

$$C = e(g_1 g_2^r \prod_{at \in \mathcal{P} \setminus \mathcal{P}'} h_i^{at_i}, g)^{\frac{H(e(sk_G, pk_D))}{H(e(sk_G, pk_S))}}$$

where $g_1 g_2^r \prod_{at \in \mathcal{P} \setminus \mathcal{P}'} h_i^{at_i}$ can be generated using components from the device's signature. The gateway G forwards the minimized signature to S and the latter checks it based on the authentication policy $\mathcal{P}'$ using its secret key sk_S and the public key pk_G of the gateway G. A valid signature means that the remaining attributes (i.e. the ones that were not discarded by being included in the component C) satisfy the policy $\mathcal{P}'$ and thus, the device D successfully authenticates to S.

3.2 Construction

Let $\mathcal{U} = \{at_1, \cdots, at_n\}$ be a small universe of n attributes. We assume that attributes are set in lexical order: the attributes at_1 should always be at the first position, the attribute at_2 should always be at the second position, and more generally, the attribute at_i should always be at the i-th position in the sequence of the universe $\mathcal{U}$. Moreover, we suppose that every participating party knows this order, and hence ensure that the element h_i is always raised to the same power, i.e. to the attribute at_i. Let $\mathcal{M}$ be the space of messages available for signing and l be the bit-length of messages.

Key Material Issuance.

• Setup$(\lambda, \mathcal{U}, \mathcal{M}) \rightarrow (params, msk)$. On inputs the security parameter λ, the attribute universe $\mathcal{U}$ and the message space $\mathcal{M}$, the algorithm outputs the public parameters *params* and the master secret key *msk* of the algorithm *TTP*.

The algorithm first generates two cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p. Let $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be a bilinear map and g be a generator of $\mathbb{G}$. Let $g_1, g_2, h_1, \cdots, h_n, u_0, u_1, \cdots, u_l$ be random elements from $\mathbb{G}$. The parameter n corresponds to the size of the universe $\mathcal{U}$ and the parameter l is determined for the bit-length of messages in $\mathcal{M}$. Let x, y be picked at random in $\mathbb{Z}_p$ and the algorithm computes g^x. Let $H : \mathbb{G}_T \rightarrow \mathbb{Z}_p$ be a hash function seen as a random oracle. The public parameters are set as $params = (p, \mathbb{G}, \mathbb{G}_T, e, g, g^x, g_1, g_2, h_1, \cdots, h_n, u_0, u_1, \cdots, u_l, H)$ and the master secret key is set as $msk = (x, y)$.

• KeyGenG$(params, msk) \rightarrow (pk_G, sk_G)$. On inputs the public parameters *params* and the master secret key *msk*, the algorithm *TTP* outputs the public and secret key pair (pk_G, sk_G) of the gateway G.

The algorithm *TTP* picks at random $\alpha \in_R \mathbb{Z}_p$ and computes the public key $pk_G = g^\alpha$ and the secret key $sk_G = g^{y\alpha}$.

• KeyGenS$(params, msk) \rightarrow (pk_S, sk_S)$. On inputs the public parameters *params* and the master secret key *msk*, the algorithm *TTP* outputs the public and secret key pair (pk_S, sk_S) of the service S.

The algorithm *TTP* picks at random $\gamma \in_R \mathbb{Z}_p$ and computes the public key $pk_S = g^\gamma$ and the secret key $sk_S = g^{y\gamma}$.

• KeyGenD$(params, \mathcal{T}, msk) \rightarrow (pk_D, sk_D, sk_{\mathcal{T}})$. On inputs the public parameters *params*, the attribute set $\mathcal{T} = \{at_{i_1}, \cdots, at_{i_k}\}$ of the device D and the master secret key *msk*, the algorithm *TTP* outputs the public and secret key tuple $(pk_D, sk_D, sk_{\mathcal{T}})$ of the device D. Let $\mathcal{T} \subseteq \mathcal{U}$ be the set of attributes granted to the device D such that $|\mathcal{T}| = k \leq n$. We suppose that the device D is given the attributes as well as a description of their positions in the universe $\mathcal{U}$. The description is a sequence as for the universe $\mathcal{U}$, where a bit 1 at position i means that $at_i \in \mathcal{T}$ while a bit 0 at position j means that $at_j \notin \mathcal{T}$.

The algorithm *TTP* picks at random $r, t, \beta \in_R \mathbb{Z}_p$ and computes the public key pk_D, the secret key $sk_D = (sk_1, sk_2, sk_3)$ and the secret attribute key $sk_{\mathcal{T}}$ as:

$$pk_D = g^\beta$$
$$sk_D = (sk_1 = g^t, sk_2 = (g^\beta)^y = g^{y\beta}, sk_3 = g_2^r)$$
$$sk_{\mathcal{T}} = (g_1 g_2^r \prod_{at_i \in \mathcal{T}} h_i^{at_i})^{\frac{1}{x+t}}$$

N.B.: The element $sk_{\mathcal{T}}$ is similarly generated as a weak Boneh-Boyen signature [12, 13].

Full Authentication.

• Sign$(params, m, pk_G, sk_D, sk_{\mathcal{T}}) \rightarrow \sigma$. On inputs the public parameters *params*, a message m, the public key pk_G of the gateway G, the secret key sk_D and the secret attribute keys $sk_{\mathcal{T}}$, the device D outputs the signature σ. Let $m = m_1 \cdots m_l \in \mathcal{M}$ be the message to be signed and $M = \{j \in [1, l]; m_j = 1\}$ be the set of indices such that the j-th bit of the message m is equal to 1.

Given $sk_D = (sk_1, sk_2, sk_3)$ and $sk_{\mathcal{T}}$, the algorithm run by the device D picks at random $s \in_R \mathbb{Z}_p$ and computes the signature $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$ as:

$$\begin{aligned}
\sigma_1 &= sk_{\mathcal{T}}^{H(e(pk_G, sk_2))} \cdot (u_0 \prod_{j \in M} u_j)^s \\
&= ((g_1 g_2^r \prod_{at_i \in \mathcal{T}} h_i^{at_i})^{\frac{1}{x+t}})^{H(e(g^\alpha, g^{y\beta}))} \\
&\quad \cdot (u_0 \prod_{j \in M} u_j)^s \\
\sigma_2 &= sk_1 = g^t, \quad \sigma_3 = sk_1^s = g^{st}, \\
\sigma_4 &= (g^x)^s = g^{sx}, \quad \sigma_5 = sk_3 = g_2^r
\end{aligned}$$

and forwards σ to the gateway G.

N.B.: The element σ_1 is similarly generated as a Waters signature [28].

• VerifG$(params, \mathcal{P}, m, \sigma, pk_D, sk_G) \to \{0,1\}$. On inputs the public parameters $params$, the authentication policy $\mathcal{P}$, the message m, the signature σ, the public key pk_D of the device D and the secret key sk_G, the gateway G outputs either 0 if the signature is valid and 1 otherwise.

The gateway G checks whether the signature $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$ is valid for the message $m = m_1 \cdots m_l \in \mathcal{M}$ given the policy $\mathcal{P} \subseteq \mathcal{U}$. Let $M = \{j \in [1, l]; m_j = 1\}$ be the set of indices such that the j-th bit of the message m is equal to 1.

The signature is accepted and the gateway G outputs 0 if the following equation holds:

$$\begin{aligned} e(\sigma_1, g^x \cdot \sigma_2) &= e(g_1 \cdot \sigma_5 \prod_{at_i \in \mathcal{P}} h_i^{at_i}, g)^{H(e(sk_G, pk_D))} \\ &\quad \cdot e(u_0 \prod_{j \in M} u_j, \sigma_4 \cdot \sigma_3) \end{aligned}$$

Delegated Authentication.

• ReSign$(params, \mathcal{P}, \mathcal{P}', \sigma, pk_S, pk_D, sk_G) \to \sigma'$. On inputs the public parameters $params$, the authentication policies $\mathcal{P}$ and $\mathcal{P}'$ such that $\mathcal{P}' \subseteq \mathcal{P}$, the signature σ, the public key pk_S of the service S, the public key pk_D of the device D, and the secret key sk_G, the gateway G outputs the minimized signature σ'.

The gateway G modifies the device's signature σ into a signature σ' for the message $m = m_1 \cdots m_l \in \mathcal{M}$ given the policies $\mathcal{P}' \subseteq \mathcal{P}$. The algorithm run by the gateway G first computes:

$$\begin{aligned} C &= e(g_1 \cdot \sigma_5 \prod_{at_i \in \mathcal{P} \setminus \mathcal{P}'} h_i^{at_i}, g)^{\frac{H(e(sk_G, pk_D))}{H(e(sk_G, pk_S))}} \\ B &= g^{\frac{H(e(sk_G, pk_D))}{H(e(sk_G, pk_S))}} \end{aligned}$$

and forwards $\sigma' = (C, B, \sigma) = (C, B, (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5))$ to the service S.

• VerifS$(params, \mathcal{P}', m, \sigma', pk_G, sk_S) \to \{0,1\}$. On inputs the public parameters $params$, the authentication subpolicy $\mathcal{P}'$, the message m, the signature σ', the public key pk_G of the gateway G and the secret key sk_S, the service S outputs either 0 if the signature is valid and 1 otherwise.

The service S checks whether the signature $\sigma' = (C, B, \sigma) = (C, B, (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5))$ is valid for a message $m = m_1 \cdots m_l \in \mathcal{M}$ given the subpolicy $\mathcal{P}'$. Let $M = \{j \in [1, l]; m_j = 1\}$ be the set of indices such that the j-th bit of the message m is equal to 1.

The signature is accepted and the service S outputs 0 if the following equation holds:

$$\begin{aligned} e(\sigma_1, g^x \cdot \sigma_2) &= C^{H(e(pk_G, sk_S))} \cdot e(u_0 \prod_{j \in M} u_j, \sigma_4 \cdot \sigma_3) \\ &\quad \cdot e(g_1 \cdot \sigma_5 \prod_{at_i \in \mathcal{P}'} h_i^{at_i}, B)^{H(e(pk_G, sk_S))} \end{aligned}$$

4 SECURITY AND PERFORMANCE

4.1 Security Models and Proofs

Three security models are defined below, for correctness, unforgeability and privacy respectively, such that participating parties are assumed not to collude.

Correctness.

For $(params, msk) \leftarrow$ Setup$(\lambda, \mathcal{U}, \mathcal{M})$, for $(pk_G, sk_G) \leftarrow$ KeyGenG $(params, msk)$ and $(pk_S, sk_S) \leftarrow$ KeyGenS$(params, msk)$, for $(pk_D, sk_D, sk_{\mathcal{T}}) \leftarrow$ KeyGenD$(params, \mathcal{T}, msk)$ for an attribute set $\mathcal{T} \subseteq \mathcal{U}$, for two policies $\mathcal{P}', \mathcal{P}$ such that $\mathcal{P}' \subseteq \mathcal{P} \subseteq \mathcal{U}$ and a message m:

• If the signature is computed as $\sigma \leftarrow$ Sign$(params, m, pk_G, sk_D, sk_{\mathcal{T}})$, then $0 \leftarrow$ VerifG$(params, \mathcal{P}, m, \sigma, pk_D, sk_G)$.

• If the minimized signature is computed as $\sigma' \leftarrow$ ReSign$(params, \mathcal{P}, \mathcal{P}', \sigma, pk_S, pk_D, sk_G)$, then $0 \leftarrow$ VerifS$(params, \mathcal{P}', m, \sigma', pk_G, sk_S)$.

Proof. Let $\mathcal{P}', \mathcal{P}$ be the authentication policies such that $\mathcal{P}' \subseteq \mathcal{P} \subseteq \mathcal{U}$. Let $m = m_1 \cdots m_l \in \mathcal{M}$ be a message and $M = \{j \in [1, l]; m_j = 1\}$ be the set of indices such that the j-th bit of the message m is equal to 1. Let a device D hold a public key pk_D, a secret key $sk_D = (sk_1, sk_2, sk_3)$ and a secret attribute key $sk_{\mathcal{T}}$. We suppose that the attributes in $\mathcal{T}$ satisfy the authentication policy $\mathcal{P}$, meaning that either $at_i \in \mathcal{T} = at_i \in \mathcal{P}$ or $\perp = \perp$ at each position i. The device D computes a signature $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$ on the message $m \in \mathcal{M}$ as described in the algorithm Sign, and σ is accepted by the gateway G since:

$$\begin{aligned} & e(\sigma_1, g^x \cdot \sigma_2) \\ = \;& e(sk_{\mathcal{T}}^{H(e(pk_G, sk_2))} \cdot (u_0 \prod_{j \in M} u_j)^s, g^x \cdot g^t) \\ = \;& e((g_1 g_2^r \prod_{at_i \in \mathcal{T}} h_i^{at_i})^{\frac{1}{x+t}}, g^{x+t})^{H(e(g^\alpha, g^{y\beta}))} \\ & \cdot e((u_0 \prod_{j \in M} u_j)^s, g^{x+t}) \\ = \;& e(g_1 g_2^r \prod_{at_i \in \mathcal{T}} h_i^{at_i}, g)^{H(e(g^{y\alpha}, g^\beta))} \cdot e(u_0 \prod_{j \in M} u_j, g^{sx} \cdot g^{st}) \\ = \;& e(g_1 \cdot \sigma_5 \prod_{at_i \in \mathcal{P}} h_i^{at_i}, g)^{H(e(sk_G, pk_D))} \cdot e(u_0 \prod_{j \in M} u_j, \sigma_4 \cdot \sigma_3) \end{aligned}$$

Let the gateway G with a public and secret key pair (pk_G, sk_G). We suppose that there exists a subset $\mathcal{T}'$ of $\mathcal{T}$ such that the attributes in $\mathcal{T}'$ satisfy the authentication subpolicy $\mathcal{P}'$, meaning that either $at_i \in \mathcal{T}' = at_i \in \mathcal{P}'$ or $\perp = \perp$ at each position i. The gateway G updates σ as a signature $\sigma' = (C, B, \sigma)$ on a message $m \in \mathcal{M}$ as described in the algorithm ReSign, and σ' is accepted by the service

S since:

$$\begin{aligned}
& e(\sigma_1, g^x \cdot \sigma_2) \\
= \quad & e(g_1 \cdot \sigma_5 \prod_{at_i \in \mathcal{P}} h_i^{at_i}, g)^{H(e(sk_G, pk_D))} \\
& \cdot e(u_0 \prod_{j \in M} u_j, \sigma_4 \cdot \sigma_3) \text{ // as shown previously} \\
= \quad & (e(g_1 \cdot \sigma_5 \prod_{at_i \in \mathcal{P}} h_i^{at_i}, g)^{\frac{H(e(sk_G, pk_D))}{H(e(sk_G, pk_S))}})^{H(e(pk_G, sk_S))} \\
& \cdot e(u_0 \prod_{j \in M} u_i, \sigma_4 \cdot \sigma_3) \\
& \text{// where } e(sk_G, pk_S) = e(g,g)^{y\alpha\gamma} = e(pk_G, sk_S) \\
= \quad & (e(g_1 \cdot \sigma_5 \prod_{at_i \in \mathcal{P} \setminus \mathcal{P}'} h_i^{at_i}, g)^{\frac{H(e(sk_G, pk_D))}{H(e(sk_G, pk_S))}})^{H(e(pk_G, sk_S))} \\
& \cdot (e(g_1 \cdot \sigma_5 \prod_{at_i \in \mathcal{P}'} h_i^{at_i}, g)^{\frac{H(e(sk_G, pk_D))}{H(e(sk_G, pk_S))}})^{H(e(pk_G, sk_S))} \\
& \cdot e(u_0 \prod_{j \in M} u_i, \sigma_4 \cdot \sigma_3) \\
= \quad & C^{H(e(pk_G, sk_S))} \cdot e(g_1 \cdot \sigma_5 \prod_{at_i \in \mathcal{P}'} h_i^{at_i}, B)^{H(e(pk_G, sk_S))} \\
& \cdot e(u_0 \prod_{j \in M} u_j, \sigma_4 \cdot \sigma_3)
\end{aligned}$$

Unforgeability. Let an attacker be an active adversary that knows the public parameters and the public keys of the gateway G, the service S and the device D. It has access to key generation oracles running KeyGenG, KeyGenS and KeyGenD. It also has access to signing oracles running algorithms Sign and ReSign. Unforgeability ensures that no signature can be forged except on messages that have been previously selected.

Let $\mathcal{B}$ be a challenger interacting with an adversary $\mathcal{A}$. We define the success probability of $\mathcal{A}$ in winning the forgery attack under chosen message and chosen attribute as follows:

$$\begin{aligned}
Pr[(params, msk) \leftarrow \mathsf{Setup}(\lambda, \mathcal{U}, \mathcal{M}), \\
(pk_G, sk_G) \leftarrow \mathcal{A}^{O_{\mathsf{KeyGenG}}(_)}(params), \\
(pk_S, sk_S) \leftarrow \mathcal{A}^{O_{\mathsf{KeyGenS}}(_)}(params), \\
(pk_D, sk_D, sk_{\mathcal{T}}) \leftarrow \mathcal{A}^{O_{\mathsf{KeyGenD}}(\mathcal{T})}(params), \\
(\mathcal{P}, \mathcal{P}', m, \sigma, \sigma') \leftarrow \mathcal{A}^{O_{\mathsf{Sign}}(\mathcal{T}, m), O_{\mathsf{ReSign}}(m)}(params, pk_G, pk_S, pk_D): \\
\mathsf{VerifG}(params, \mathcal{P}, m, \sigma, pk_D, sk_G) \rightarrow 0 \\
\wedge \mathsf{VerifS}(params, \mathcal{P}', m, \sigma', pk_G, sk_S) \rightarrow 0 \\
\wedge (\mathcal{P}, \mathcal{P}', m, \sigma, \sigma') \notin Q] < \varepsilon(\lambda)
\end{aligned}$$

where O_{KeyGenG} produces the output of KeyGenG($params, msk$), O_{KeyGenS} produces the output of KeyGenS($params, msk$), O_{KeyGenD} takes as input an attribute set $\mathcal{T}$ and produces the output of KeyGenD ($params, \mathcal{T}, msk$), O_{Sign} takes as inputs an attribute set $\mathcal{T}$ and a message m, and produces the output of Sign($params, m, pk_G, sk_D, sk_{\mathcal{T}}$), and O_{ReSign} takes as input a message m, and produces the output of ReSign($params, \mathcal{P}, \mathcal{P}'$, Sign($params, m, pk_G, sk_D, sk_{\mathcal{T}}$)), pk_S, pk_D, sk_G). Q denotes the set of tuples $(\mathcal{P}, \mathcal{P}', m, \sigma, \sigma')$ where $\mathcal{A}$ obtained a signature σ and a minimized signature σ' on m under policies $\mathcal{P}$ and $\mathcal{P}' \subseteq \mathcal{P}$ by querying O_{Sign} or O_{ReSign}, such that the attributes in the set $\mathcal{T}$ queried to the oracle O_{KeyGenD} satisfy $\mathcal{P}$. We say that the authentication protocol is secure against forgery under chosen message and chosen attribute attacks if $\varepsilon(\lambda)$ is negligible.

Sketch of Proof. The secret attribute key $sk_{\mathcal{T}}$ for the attribute set $\mathcal{T}$ of the device is generated as a weak Boneh-Boyen signature [12, 13]. The Boneh-Boyen scheme is proved secure under the Strong Diffie-Hellman (SDH) assumption [21]. The SDH problem is stated as follows: Given the tuple $(g, g^x, \cdots, g^{x^q}, h, h^x) \in \mathbb{G}^{q+3}$, it remains hard to output $(c, g^{\frac{1}{x+c}})$ for some exponent $c \in \mathbb{Z}_p$ such that $x + c \not\equiv 0 \mod p$.

The algorithm Sign is based on the one in Waters' signature scheme [28]. Waters proves that his scheme is existentially unforgeable assuming the Decisional Bilinear Diffie-Hellman (DBDH) assumption holds [23]. The DBDH problem is stated as follows: Given the tuple $(g, g^a, g^b, g^c) \in \mathbb{G}^4$ for unknown exponents $a, b, c \in \mathbb{Z}_p$, and $T \in \mathbb{G}_T$, it remains to determine if $T = e(g,g)^{abc}$ or a random element.

Since we assume that participating parties do not collude, both secret attribute key $sk_{\mathcal{T}}$ and signature σ are securely generated based on [12, 13] and [28] respectively. Moreover, key material for participating parties is randomized, the hash function H is seen as a random oracle, and verification and re-signing processes are privatized such that the parties taking care of these processes require their secret keys.

Privacy. Let an attacker be an active adversary that knows the public parameters and the public keys of the gateway G, the service S and the device D. It has access to key generation oracles running KeyGenG, KeyGenS and KeyGenD. Two signatures generated with the device's secret key material and whose attribute sets are different should remain indistinguishable in the view of the adversary. In other words, no information about the attributes that have been used to generate the signature is leaked beyond the fact that they satisfy the policy, while non-used attributes remain completely hidden.

Let $\mathcal{B}$ be a challenger interacting with an adversary $\mathcal{A}$. We define the success probability of $\mathcal{A}$ in winning the attribute privacy attack as follows:

$$\begin{aligned}
Pr[(params, msk) \leftarrow \mathsf{Setup}(\lambda, \mathcal{U}, \mathcal{M}), \\
(pk_G, sk_G) \leftarrow \mathcal{A}^{O_{\mathsf{KeyGenG}}(_)}(params), \\
(pk_S, sk_S) \leftarrow \mathcal{A}^{O_{\mathsf{KeyGenS}}(_)}(params), \\
(pk_D, sk_D, sk_{\mathcal{T}}) \leftarrow \mathcal{A}^{O_{\mathsf{KeyGenD}}(\mathcal{T})}(params), \\
(\mathcal{P}, \mathcal{P}', (\mathcal{T}^0, \mathcal{T}^1), m) \leftarrow \mathcal{A}(_), \\
b \leftarrow \{0, 1\}, \\
\sigma \leftarrow \mathsf{Sign}(params, m, pk_G, sk_D, sk_{\mathcal{T}^b}), \\
\sigma' \leftarrow \mathsf{ReSign}(params, \mathcal{P}, \mathcal{P}', \sigma, pk_S, pk_D, sk_G), \\
b' \leftarrow \mathcal{A}(\sigma, \sigma'), \\
b' = b] - 1/2 < \varepsilon(\lambda)
\end{aligned}$$

where O_{KeyGenG} produces the output of KeyGenG($params, msk$), O_{KeyGenS} produces the output of KeyGenS($params, msk$), O_{KeyGenD} takes as input an attribute set $\mathcal{T}$ and produces the output of KeyGenD

Table 1: Computational cost for each algorithm

	Multiplication in $\mathbb{G}$	Exponentiation in $\mathbb{G}$	Pairing in $\mathbb{G}_T$
KeyGenG	-	2	-
KeyGenS	-	2	-
KeyGenD	$k+1$	$2k+5$	-
Sign	$\|M\|+1$	4	1
VerifG	$\|\mathcal{P}\|+\|M\|+4$	$\|\mathcal{P}\|+2$	6
ReSign	$\|\mathcal{P}\|-\|\mathcal{P}'\|+1$	2	4
VerifS	$\|\mathcal{P}'\|+\|M\|+3$	$\|\mathcal{P}'\|+2$	2

$(params, \mathcal{T}, msk)$. We say that the authentication protocol is private regarding the attributes if $\varepsilon(\lambda)$ is negligible.

Sketch of Proof. A secret attribute key $sk_{\mathcal{T}} = (g_1 g_2^r \prod_{at_i \in \mathcal{T}} h_i^{at_i})^{\frac{1}{x+t}}$ embeds the attributes at_i in $\mathcal{T} \subseteq \mathcal{U}$. The signature σ and minimized signature σ' comprise $sk_{\mathcal{T}}$ into the component σ_1 where $\sigma_1 = sk_{\mathcal{T}}^{H(e(pk_G, sk_2))} \cdot (u_0 \prod_{j \in M} u_j)^s$. Let $\mathcal{R} \subseteq \mathcal{U}$ be an attribute set such that $\mathcal{R} \neq \mathcal{T}$. Since there exists a secret attribute key $sk_{\mathcal{R}}$ for $\mathcal{R}$ such that $sk_{\mathcal{R}} = (g_1 g_2^{r'} \prod_{at_i \in \mathcal{R}} h_i^{at_i})^{\frac{1}{x+t'}} = (g_1 g_2^r \prod_{at_i \in \mathcal{T}} h_i^{at_i})^{\frac{1}{x+t}} = sk_{\mathcal{T}}$ for exponents $r' \neq r$ and $t' \neq t$, σ_1 can be seen as a signature component for attribute set $\mathcal{R}$ and elements $r', t' \in \mathbb{Z}_p$. Therefore, the privacy of all the attributes in σ and σ' is preserved.

4.2 Performance Analysis

In the following tables, let $\mathcal{U}$ be the universe with $|\mathcal{U}| = n$, and $\mathcal{T} \subseteq \mathcal{U}$ be the attribute set of the device D with $|\mathcal{T}| = k \leq n$. Let $\mathcal{P}', \mathcal{P} \subseteq \mathcal{U}$ be the authentication policies such that $|\mathcal{P}'|, |\mathcal{P}| \leq n$, and $M = \{j \in [1, l]; m_j = 1\}$ for a message $m = m_1 \cdots m_l$ such that $|M| \leq l$. Policies are supposed to contain up to 30 attributes regarding real scenarios [8, 29], hence the universe $\mathcal{U}$ remains relatively small (say, around 50 attributes) and the device holds a limited number of attributes (say, around 20 attributes).

Table 1 provides the computational cost induced by all the algorithms. The generation of the key material for the gateway G and the service S is simple since it requires a constant number of basic operations. The generation of the key material of the device D is more demanding since it depends on the number k of its attributes; however this number is limited, and thus key generation appears reasonable.

The generation of the signature σ by the device D requires few computations, relying on the number of bits equal to 1 in the message m. The minimized signature σ' computed by the gateway G requires more effort, due to the number of attributes in the policies $\mathcal{P}'$ and $\mathcal{P}$ along with pairing operations on key components from the gateway G, service S and device D. Verification of the signatures is the most costly part since policies $\mathcal{P}', \mathcal{P}$ and message m are involved. Yet, the verification of the minimized signature σ is faster than the verification of the device's signature σ since $|\mathcal{P}'| \leq |\mathcal{P}|$.

Table 2 provides the storage cost evaluated at parties' side. The number of public parameters *params*, that all parties should store, is equal to $n + l + 4$. These parameters remain the elements that most impact storage. Key material of the device D, gateway G and service S, as well as signatures σ and σ', have a constant number of elements, i.e. not depending on the number of attributes in attribute sets and policies; this is the optimal result.

Table 2: Storage cost for each party (number of stored elements)

TTP	G			S			D		
msk	pk_G	sk_G	σ	pk_S	sk_S	σ'	pk_D	sk_D	$sk_{\mathcal{T}}$
2	1	1	5	1	1	7	1	3	1

Table 3: Timing in milliseconds

	Configuration 1	Configuration 2
KeyGenG	4.05	4.30
KeyGenS	4.05	4.33
KeyGenD	20.63	155.87
Sign	8.51	107.23
VerifG	6.44	98.35
ReSign	6.80	97.96
VerifS	7.20	100.04

Table 3 provides the timing results from all the algorithms. Each algorithm is run 10 times and the average on these runs is given in the table. The experiments are tested considering two different configurations, one for the trusted third party, gateway and service and one for the smart device. Hence, we use two computers: 1) the first one with a processor Intel Core $i5-2500$ CPU @3.30GHz ×4 with RAM 16GiB and OS Linux Ubuntu 14.04 LTS (Configuration 1 for the trusted third party, gateway and service); 2) the second one at 10% execution cap of a processor Intel Core $i5-2500$ CPU @3.30GHz ×4 with RAM 512MB and OS Linux Ubuntu 18.04 LTS (Configuration 2 for the IoT device). Experiments consider a universe $\mathcal{U}$ with $n = 20$ attributes, an attribute set $\mathcal{T}$ with $k = 10$ attributes, a policy $\mathcal{P}$ with 6 attributes and a subpolicy $\mathcal{P}'$ with 3 attributes. The length of messages is set to $l = 256$ bits. All the algorithms are executed following the Configurations 1 and 2, in order to emphasize the gain for the device to devolve delegated authentications to a gateway.

Key generation for the gateway and service is a fast process, while key generation for the device spends more time since it depends on the number of attributes in the set $\mathcal{T}$. Verification of the signatures σ and σ' does not charge much the gateway and service respectively.

Execution times of the algorithm ReSign and of the algorithm Sign are similar, around 100ms with Configuration 2. Nevertheless, execution time of the algorithm ReSign with Configuration 1 is much less, around 7ms. Hence, without authentication delegation to the gateway, the device would use 100ms for each delegated authentication, while with the assistance of the gateway, the device spends 100ms only once, at full authentication, and the gateway only consumes 7ms for each delegated authentication. Therefore, such results validate the need of the proposed solution.

4.3 Related Work

Attribute-Based Authentication for IoT. Authentication and privacy in IoT have been discussed in the literature [2, 4, 24, 31].

Most of the papers emphasize that privacy issues emerge with the heterogeneous nature of devices and the huge amount of collected data. Extension of the sensor functionalities has been suggested in order to enable data recording and processing in addition to data collection, but nothing precise is given.

Attribute-Based Credentials (ABC) technology [27] has been developed for flexible and privacy-preserving authentication, offering unlinkability and selective disclosure properties. ABC-based lightweight infrastructures are presented in [6, 7, 18, 26], but cannot suit IoT environments. The position paper [5] discusses on authentication and privacy using ABC technology, but no concrete solution is given.

Authentication Protocol with Data Minimization. In order to perform authentication with data minimization, there exist cryptographic protocols such as privacy-enhancing attribute-based credentials [11], anonymous credentials [17, 19] and minimal-disclosure tokens [14]. Users are allowed to obtain certified credentials from an issuer. Then, users use these credentials to authenticate to services such that they are enabled to only disclose some of the attributes and to keep hidden the other ones from the services. Homomorphic [22], sanitizable [9], redactable [30] and content extracting signature [15] methodologies enable a gateway to decrease the part of verifiable information in signed messages while keeping the service authentication process on the resulting signature be available. Slightly homomorphic signatures [3] and transitive signatures [25] allow data minimization on signed messages as well as other options such as authenticable relationship and derivability. The aforementioned works focus on modifying the signed messages.

Other notions include delegable anonymous credentials [10] where credentials are repetitively delegated while the identity of the delegators remain hidden, and structure-preserving signatures [1] and commuting signatures [20] where the public key, the message and the signature are in the same mathematical group, helping to design delegable credentials. Nevertheless, the protocols proposed in the previously mentioned works cannot bind attributes to a delegable credential.

The work in [16] presents an attribute-based privacy-preserving audit protocol that considers presentation tokens derived from signed credentials. By doing so, a malicious gateway that tries to impersonate a user at other gateways cannot successfully use the presentation tokens. Hence, at first sight, this work is really similar to ours. However, the size of authentication proofs remains linear in the number of attributes, making the solution be not applicable to IoT where lightweight computations and storage are essential.

While existing solutions in the literature do not completely suit IoT requirements by lacking of practicality and/or security, our solution appears to fully satisfy these requirements by allowing constant size for components stored and generated by devices, and by preserving attribute privacy.

5 CONCLUSION

In this paper, we proposed a new solution to authenticate IoT devices to services, with gateway assistance and constant-size key material and signature. Delegating the authentication process to the gateway relieves the workload at devices' side. Moreover, attribute privacy is preserved from services by allowing the gateway to apply data minimization mechanism, i.e. to discard sensitive attributes from devices' credentials. These contributions make our privacy-preserving delegable authentication scheme suitable for IoT environments.

A centralized gateway can incur a single point of failure. If the gateway fails, then the entire system would stop being functional, impacting its availability and reliability. Therefore, to remove such flaws, some future work could extend our solution to a decentralized setting, possibly based on distributed ledger technology.

ACKNOWLEDGMENTS

This work was supported by the EU FP7 ERANET program under grant CHIST-ERA-2016 UPRISE-IOT.

REFERENCES

[1] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. 2010. Structure-Preserving Signatures and Commitments to Group Elements. In *Proceedings of CRYPTO'10*. Springer Berlin Heidelberg, Berlin, Heidelberg, 209–236.

[2] C. C. Aggarwal, N. Ashish, and A. Sheth. 2013. The Internet of Things: A Survey from the Data-Centric Perspective. In *Managing and Mining Sensor Data*. Springer US, Boston, MA, 383–428.

[3] J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, A. Shelat, and B. Waters. 2015. Computing on Authenticated Data. *Journal of Cryptology* 28, 2 (01 Apr 2015), 351–395.

[4] A. Alcaide, E. Palomar, J. Montero-Castillo, and A. Ribagorda. 2013. Anonymous Authentication for Privacy-preserving IoT Target-driven Applications. *Computer Security* 37 (Sept. 2013), 111–123.

[5] G. Alpár, L. Batina, L. Batten, V. Moonsamy, A. Krasnova, A. Guellier, and I. Natgunanathan. 2016. New Directions in IoT Privacy Using Attribute-based Authentication. In *Proceedings of CF '16*. ACM, New York, NY, USA, 461–466.

[6] G. Alpár, L. Batina, and W. Lueks. 2013. Designated Attribute-Based Proofs for RFID Applications. In *Proceedings of RFIDSec'12*. Springer Berlin Heidelberg, Berlin, Heidelberg, 59–75.

[7] G. Alpár and J.-H. Hoepman. 2013. A Secure Channel for Attribute-based Credentials. In *Proceedings of DIM '13*. ACM, New York, NY, USA, 13–18.

[8] M. Ambrosin, A. Anzanpour, M. Conti, T. Dargahi, S. R. Moosavi, A. M. Rahmani, and P. Liljeberg. 2016. On the Feasibility of Attribute-Based Encryption on Internet of Things Devices. *IEEE Micro* 36, 6 (Nov 2016), 25–35.

[9] G. Ateniese, D. H. Chou, B. de Medeiros, and G. Tsudik. 2005. Sanitizable Signatures. In *Proceedings of ESORICS'05*. Springer Berlin Heidelberg, Berlin, Heidelberg, 159–177.

[10] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, and H. Shacham. 2009. Randomizable Proofs and Delegatable Anonymous Credentials. In *Proceedings of CRYPTO'09*. Springer Berlin Heidelberg, Berlin, Heidelberg, 108–125.

[11] P. Bichsel, J. Camenisch, M. Dubovitskaya, R. R. Enderlein, S. Krenn, I. Krontiris, A. Lehmann, G. Neven, C. Paquin, F.-S. Preiss, K. Rannenberg, and A. Sabouri. 2015. An Architecture for Privacy-ABCs. In *Attribute-based Credentials for Trust: Identity in the Information Society*. Springer International Publishing, Cham, 11–78.

[12] D. Boneh and X. Boyen. 2008. Short Signatures Without Random Oracles and the SDH Assumption in Bilinear Groups. *Journal of Cryptology* 21, 2 (2008), 149–177.

[13] D. Boneh, X. Boyen, and H. Shacham. 2004. Short Group Signatures. In *Proceedings of CRYPTO'04*. Springer Berlin Heidelberg, Berlin, Heidelberg, 41–55.

[14] S. A. Brands. 2000. Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press, Cambridge, MA, USA.

[15] C. Brzuska, H. Busch, O. Dagdelen, M. Fischlin, M. Franz, S. Katzenbeisser, M. Manulis, C. Onete, A. Peter, B. Poettering, and D. Schröder. 2010. Redactable Signatures for Tree-Structured Data: Definitions and Constructions. In *Proceedings of ACNS'10*. Springer Berlin Heidelberg, Berlin, Heidelberg, 87–104.

[16] J. Camenisch, A. Lehmann, G. Neven, and A. Rial. 2014. Privacy-Preserving Auditing for Attribute-Based Credentials. In *Proceedings of ESORICS'14*. Springer International Publishing, Cham, 109–127.

[17] J. Camenisch and A. Lysyanskaya. 2003. A Signature Scheme with Efficient Protocols. In *Proceedings of SCN'02*. Springer-Verlag, Berlin, Heidelberg, 268–289.

[18] Jan Camenisch and Els Van Herreweghen. 2002. Design and Implementation of the Idemix Anonymous Credential System. In *Proceedings of CCS '02*. ACM, New York, NY, USA, 21–30.

[19] D. Chaum. 1985. Security Without Identification: Transaction Systems to Make Big Brother Obsolete. *Communication ACM* 28, 10 (Oct. 1985), 1030–1044.

[20] G. Fuchsbauer. 2011. Commuting Signatures and Verifiable Encryption. In *Proceedings of EUROCRYPT'11*. Springer Berlin Heidelberg, Berlin, Heidelberg, 224–245.

[21] D. Jao and K. Yoshida. 2009. Boneh-Boyen Signatures and the Strong Diffie-Hellman Problem. In *Proceedings of Pairing'09*. Springer-Verlag, Berlin, Heidelberg, 1–16.

[22] R. Johnson, D. Molnar, D. Song, and D. Wagner. 2002. Homomorphic Signature Schemes. In *Proceedings of CT-RSA'02*. Springer Berlin Heidelberg, Berlin, Heidelberg, 244–262.

[23] A. Joux. 2004. A One Round Protocol for Tripartite Diffie-Hellman. *Journal of Cryptology* 17, 4 (Sept. 2004), 263–276.

[24] F. Mattern and C. Floerkemeier. 2010. From the Internet of Computers to the Internet of Things. In *From Active Data Management to Event-based Systems and More*. Springer-Verlag, Berlin, Heidelberg, 242–259.

[25] S. Micali and R. L. Rivest. 2002. Transitive Signature Schemes. In *Proceedings of CT-RSA'02*. Springer Berlin Heidelberg, Berlin, Heidelberg, 236–243.

[26] C. Paquin. 2013. *U-Prove Technology Overview V1.1*. Technical Report (revision 2). Microsoft Research.

[27] K. Rannenberg, J. Camenisch, and A. Sabouri. 2014. Attribute-based Credentials for Trust: Identity in the Information Society. Springer Publishing Company, Incorporated.

[28] B. Waters. 2005. Efficient Identity-based Encryption Without Random Oracles. In *Proceedings of EUROCRYPT'05*. Springer-Verlag, Berlin, Heidelberg, 114–127.

[29] X. Yao, Z. Chen, and Y. Tian. 2015. A lightweight attribute-based encryption scheme for the Internet of Things. *Future Generation Computer Systems* 49, Supplement C (2015), 104 – 112.

[30] T. H. Yuen, W. Susilo, J. K. Liu, and Y. Mu. 2008. Sanitizable Signatures Revisited. In *Cryptology and Network Security*. Springer Berlin Heidelberg, Berlin, Heidelberg, 80–97.

[31] J. H. Ziegeldorf, O. G. Morchon, and K. Wehrle. 2014. Privacy in the Internet of Things: Threats and Challenges. *Security and Communication Networks* 7, 12 (2014), 2728–2742.