CHARIOT: Cloud-Assisted Access Control for the Internet of Things

Clémentine Gritti
Eurecom
Sophia Antipolis, France
gritti@eurecom.fr

Melek Önen
Eurecom
Sophia Antipolis, France
onen@eurecom.fr

Refik Molva
Eurecom
Sophia Antipolis, France
molva@eurecom.fr

***Abstract*—The Internet of Things (IoT) technology has expanded widely across the world, promising new data management opportunities for industries, companies and individuals in different sectors, such as health services or transport logistics. This trend relies on connecting devices/things to collect, exchange and store data. The exponentially increasing number of IoT devices, their origin diversity, their limited capabilities in terms of resources, as well as the ever-increasing amount of data, raise new challenges for security and privacy protection, precluding traditional access control solutions to be integrated to this new environment. In this paper, we propose a reliable server-aided policy-based access control mechanism, named CHARIOT, that enables an IoT platform to verify credentials of different devices requesting access (read/write) to the data stored within it. CHARIOT permits IoT devices to authenticate themselves to the platform without compromising their privacy by using attribute-based signatures. Our solution also allows secure delegation of costly computational operations to a cloud server, hence relieving the workload at IoT devices' side.**

***Index Terms*—Access Control, Cloud Computing, Internet of Things**

I. Introduction

The Internet of Things (IoT) has been developed to enable the interconnection between various devices, such as mobile phones, sensors and actuators, that collect and transmit data at large-scale. This technology has been integrated to multiple and diversified environments such as environmental production, health services and traffic monitoring. With the explosion of the amount of data exchanges between these devices and their large number, the need for security and privacy countermeasures becomes crucial. Several studies [1]–[3] report on different vulnerabilities of the technology.

While many platforms are available for the IoT, access control issues are often overlooked. Various attacks exist against such platforms; for example, vulnerabilities in ToyTalk products were discovered among which the majority originated from the lack of security at the platform level whereby an adversary was easily able to log-in to the platform at any time[1]. Preventing unauthorized access to the platform and consequently to the data stored in there becomes extremely challenging due to the nature of IoT devices. The latter, that are often simple and resource-constrained, cannot perform any costly computational operations. Since platforms are usually installed at untrusted third parties, such as data centers and cloud servers, such parties must not obtain any information about the identity of IoT devices and the data that is being collected or accessed.

Traditional credential-based access control systems do not suit this environment as such resource-limited devices are often not able to generate credentials or signatures to satisfy the access control policy defined for an IoT platform. We hence aim to develop a server-aided access control solution, named CHARIOT, which ensures the privacy of the devices' identity and of the data towards the platform. CHARIOT relies on the use of attribute-based signatures to ensure the proper authentication of devices, by defining an access control policy for the data stored at the IoT platform. Furthermore, CHARIOT enables an IoT device to delegate most of the access control operations to a more powerful third party such as a cloud server, and only perform minor operations at its side in order to optimize the use of its resources. The outsourced computation of the credentials is based on a simple secret sharing of the signing key between the cloud server and the IoT device. Once the cloud server has executed the main part of the generation of the signature, the IoT device performs very few additional operations to finalize it and sends the resulting signature to the platform. CHARIOT ensures that no information about the devices' attributes in the credentials is leaked to the platform nor to the cloud server.

In the following section, we detail the problem raised by designing an access control protocol suitable to the IoT environment. In Section I-B, we highlight the main features of our cloud server-assisted access control protocol for IoT. Section II describes the proposed solution CHARIOT and analyzes its security. In Section III, we evaluate the performance of our solution and recall the existing work. We finally conclude the paper in Section IV.

A. Problem Statement

As already introduced, serious issues on security have been encountered because of giving access to sensitive information to unauthorized parties in the IoT context [1]–[3]. Access control systems appear to be the essential strategy to overcome these threats. Nevertheless, traditional access control solutions relying on public-key infrastructures fall short for this technol-

[1]https://motherboard.vice.com/en_us/article/more-security-vulnerabilities-found-in-hello-barbie-toys-servers

ogy mainly because of the very limited computing resources of IoT devices in terms of memory, CPU, storage and battery.

A suitable technique allowing the secure authentication of devices to the platform is Attribute-Based Signature (ABS) [4]. Informally speaking, an ABS protocol in the IoT environment ensures that given an access policy, whenever a device signs a message using its attributes, if the attributes satisfy this access policy, then this signature is valid and the device successfully authenticates and accesses the platform. Additionally, ABS ensures that these attributes remain hidden in the signature, and thus the device's identity remains private. Indeed, during the verification phase, the platform cannot discover any information except that the access control policy is satisfied.

However, current ABS solutions [5], [6] suffer from the computational burden of the signature generation. Since the signature computation involves multiple modular exponentiations and their number increases linearly with the policy's size, the existing tools are not yet suitable to the IoT environment.

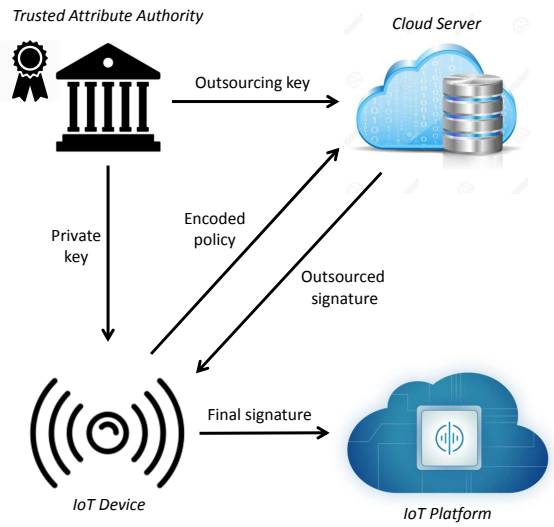
B. Idea

Most of the existing ABS solutions are very costly in terms of computation and storage: the generation of the signature and its size depend on the number of attributes in the signing policies. In the context of an IoT environment, a typical access control policy may contain numerous attributes because of the large number of IoT devices and their heterogeneity. Herranz et al. [6] present a Threshold ABS scheme with compact signatures whereby their size does not depend on the number of attributes in the policies. While the solution is suitable for the IoT technology, the computational cost still remains significant: the number of modular exponentiations is linear with the number of attributes and therefore cannot be afforded by resource-constrained IoT devices.

We propose to improve the computational cost of the above solution at the device's side by delegating most of the signature generation to a powerful cloud server. Similarly to [7], the signing key is secretly shared between the cloud server and the IoT device, ensuring a secure delegation of the signature computation. The cloud server computes a partial signature using an outsourcing key, and sends it to the IoT device. The latter finalizes the signature by performing additional lightweight operations using its private key, and forwards it to the platform. The signature is accepted by the platform if the device's attributes, embedded into the signature, satisfy the access control policy.

Yet, the ABS solution in [6] incurs extra computational and storage overhead since the signature generation is fixed to an upper bound due to the use of dummy attributes. Following a technique introduced in [8], we modify the original ABS in [6] by removing the presence of dummy attributes. We hence obtain a more efficient Threshold ABS scheme with constant-size signatures and no dummy attributes whereby the most computationally-intensive operations are securely delegated to a cloud server. The proposed solution guarantees privacy of participating IoT devices against the platform and cloud server.

Fig. 1. CHARIOT protocol overview



II. CHARIOT

A. Environment

We define an Outsourced Threshold Attribute-Based Signature protocol as follows. The first two algorithms, Setup and Keygen, are run to initialize the protocol. The Setup algorithm is run to generate the public parameters accessible to all participating parties, and a master secret key forwarded to an off-line trusted attribute authority. The Keygen algorithm is run by the trusted attribute authority to create the outsourcing key for the cloud server and the private key for the IoT device, regarding the access attributes of this device.

The following algorithms, Request, Sign_{out}, Sign and Verify, are the core algorithms to enable secure authentication of the IoT device towards the platform. First, the IoT device runs Request to hash the access policy for the cloud server using a Keyed-Hash Message Authentication Code (HMAC). The access policy is defined by the platform and made accessible to the device, but should remain hidden from the cloud server's view. Then, the cloud server, given this hashed access policy, creates the outsourced signature using its outsourcing key. It forwards the outsourced signature to the IoT device. From there, the device finalizes the signature generation by using its private key and choosing the message. It forwards the final signature to the IoT platform. The latter executes Verify to check the validity of the device's signature and thus its right for access.

Figure 1 illustrates the CHARIOT protocol where a trusted attribute authority, an IoT platform, a cloud server and an IoT device participate. More details on the components (keys and signatures) are given in the next section.

B. Preliminaries

Notations. Let Y be a finite set. The notation $y \in_R Y$ stands for y is a random variable uniformly chosen from Y.

Let $g \in \mathbb{G}$ and $\vec{v} = (v_1, v_2, v_3)^\top$ where $v_i \in \mathbb{G}$ for $i \in [1,3]$. We denote $E(g, \vec{v})$ the pairing-based vector $(e(g, v_1), e(g, v_2), e(g, v_3))^\top$ where $e(g, v_i) \in \mathbb{G}_T$ for $i \in [1,3]$. In addition, we let the multiplication between two column vectors be $(v_1, v_2, v_3)^\top \cdot (v'_1, v'_2, v'_3)^\top = (v_1 \cdot v'_1, v_2 \cdot v'_2, v_3 \cdot v'_3)^\top$ where $v'_i \in \mathbb{G}$ for $i \in [1,3]$.

Let X be an attribute set, at be an attribute in X, τ be an injective encoding such that all $\tau(at)$ are pairwise-distinct, and γ be an element in $\mathbb{Z}_p$. We denote $F_X(\gamma) = \prod_{at \in X}(\gamma + \tau(at))$ the polynomial of degree $|X|$ for attributes $at \in X$.

Bilinear Pairings. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic multiplicative groups of prime order p. Let e be an efficiently computable mapping $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ such that:

- $e(g^a, h^b) = e(g, h)^{ab}$ for any $g, h \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$,
- $e(g, h) \neq 1_{\mathbb{G}_T}$ whenever $g, h \neq 1_{\mathbb{G}}$.

We call e an admissible pairing. Moreover, e is symmetric: for all $g, h \in \mathbb{G}$, $e(g, h) = e(h, g)$.

DLIN Assumption [6]. The Decisional LINear problem related to the group $\mathbb{G}$ is defined as follows. Let g be a generator of $\mathbb{G}$ and γ be an unknown exponent in $\mathbb{Z}_p$. Given an integer $n \in \mathbb{N}$ and the values $g, \{g^{\gamma^i}\}_{i \in [1,2n] \setminus \{n+1\}}$ in $\mathbb{G}$, the aim is to compute $g^{\gamma^{n+1}}$.

***(q,n,s,t)*-aMSE-CDH Assumption [6], [8].** The Augmented Multi-Sequence of Exponents Computational Diffie-Hellman problem related to the group pair $(\mathbb{G}, \mathbb{G}_T)$ is defined as follows. Let g_0, h_0 be two generators of $\mathbb{G}$. Let $f(x), g(x)$ be two co-prime polynomials defined as $f(x) = \prod_{i=1}^{q}(x + x_i)$ and $g(x) = \prod_{i=1}^{s}(x + x'_i)$ where x_i, x'_i are given. Given $q, n, s, t, f(x), g(x)$ and the following values in $\mathbb{G}$:

$$g_0, h_0, \{g_0^{\alpha\gamma^i}\}_{i \in [0,q+n]}, g_0^{\alpha f(\gamma)\gamma^{s-t}}, \{g_0^{\beta\gamma^i}\}_{i \in [0,q+t]},$$
$$\{g_0^{\omega\gamma^i}\}_{i \in [0,q+t]}, \{h_0^{\alpha\gamma^i}\}_{i \in [0,2n]}, h_0^{\alpha g(\gamma)\gamma^n},$$
$$\{h_0^{\beta\gamma^i}\}_{i \in [0,n+t-2]}, \{h_0^{\omega\gamma^i}\}_{i \in [0,n+t]}$$

where $\alpha, \beta, \gamma, \omega$ are unknown exponents of $\mathbb{Z}_p$, the aim is to compute $e(g_0, h_0)^{\alpha\beta f(\gamma)\gamma^{n+s-1}}$.

C. Building Blocks

Similarly to Herranz et al. [6], our ABS scheme relies on two main features, namely the Attribute-Based Encryption (ABE) scheme with constant-size ciphertexts proposed by [9] and the Groth-Sahai proof systems for bilinear groups [10].

The ABE scheme in [9] is designed for the threshold case, where users are authorized to decrypt if they have at least t attributes matching the ones from an attribute universe, for some threshold t chosen by the party who encrypts the message. Such scheme relies on expressing a polynomial as the product of irreducible factors of degree 1 with coefficients equal to the attributes from either the user's set or the universe. Fraction of such polynomials can thus be simplified when t attributes among the user's set and the universe match.

Herranz et al. [6] design their Threshold ABS scheme by enabling the signer to implicitly prove that it can decrypt a ciphertext generated as in the ABE scheme [9]. To do so, the signer generates a Groth-Sahai proof [10] in which the message and access policy are binded using a technique from [11]. Informally, by hashing the to-be-signed message using Waters' techniques [12] and embedding it into the Groth-Sahai Common Reference String (CRS), the technique in [11] allows signatures of knowledge.

In addition, we modify the ABS scheme presented in [6] by enabling the outsourcing of the signature generation to a cloud server. Similarly to Chen et al. [7], we let the cloud server and the IoT device hold shares of a secret element chosen by the trusted attribute authority, and use them to generate the signature such that the combination of the two shares recover the secret.

Groth-Sahai Proofs. Groth-Sahai proofs [10] are used to construct signatures in our protocol. Based on the DLIN assumption and symmetric pairings, they guarantee unforgeability and privacy of signatures. Let $g_1, g_2, g \in \mathbb{G}$ and vectors

$$\vec{g}_1 = (g_1, 1, g)^\top, \vec{g}_2 = (1, g_2, g)^\top, \vec{g}_3 \in \mathbb{G}^3$$

be defined in the CRS used in Groth-Sahai proof systems. Let $r, s, t \in \mathbb{Z}_p$ and

$$\vec{C} = (1, 1, X)^\top \cdot (\vec{g}_1)^r \cdot (\vec{g}_2)^s \cdot (\vec{g}_3)^t$$

where $X \in \mathbb{G}$ is the element to commit. Let $\xi_1, \xi_2 \in \mathbb{Z}_p^*$ be randomly chosen such that $\vec{g}_3 = (\vec{g}_1)^{\xi_1} \cdot (\vec{g}_2)^{\xi_2}$. Let the elements of

$$\vec{C} = (g_1^{r+\xi_1 t}, g_2^{s+\xi_2 t}, X \cdot g^{r+s+t(\xi_1+\xi_2)})^\top$$

be the resulting commitments. As mentioned in [6], [10], the elements of $\vec{C}$ are ciphertexts from the Boneh-Boyen-Shacham encryption scheme [13] that can be decrypted using $\log_g(g_1)$ and $\log_g(g_2)$.

Let $\vec{g}_3 = (\vec{g}_1)^{\xi_1} \cdot (\vec{g}_2)^{\xi_2} \cdot (1, 1, g^{-1})^\top$. Hence, $\vec{g}_1, \vec{g}_2, \vec{g}_3$ are linearly independent vectors and $\vec{C}$ is a perfectly hiding commitment. Thus, proofs are perfectly witness indistinguishable.

The committed group elements are proved to satisfy some relations when there are one commitment per variable and one proof element (made of a constant number of group elements) per relation. In the context of pairing-based systems, let t_T be constant in $\mathbb{G}_T$, A_i be constant in $\mathbb{G}$, $B_i \in \mathbb{G}$ and $a_{i,j} \in \mathbb{Z}_p$, for $i, j \in [1, k]$. The pairing-product equations are $\prod_{i=1}^{k} e(A_i, B_i) \cdot \prod_{i=1}^{k}\prod_{j=1}^{k} e(B_i, B_j)^{a_{i,j}} = t_T$.

In addition, Non-Interactive Zero-Knowledge (NZIK) proofs are available with pairing-product relations. Let S_i, T_i, for $i \in [1, k']$ and $k' \in \mathbb{N}$ be constant values and $t_T = \prod_{i=1}^{k_1} e(S_i, T_i)$ be a target element. These extra variables incur additional costs. A trapdoor makes possible to simulate the proofs without knowing the witnesses, such that witness indistinguishability is satisfied by the CRS set. In CHARIOT, we use pairing-product linear equations, meaning that $a_{i,j} = 0$ for $i, j \in [1, k]$, containing three group elements.

Threshold ABS. Let $\Omega_S = \Omega \cap S$ where Ω is the attribute set of the signer and S is the attribute set included in the signing policy $\Gamma = (t, S)$, where $|S| = s$ and t is the threshold. Let τ be an injective encoding such that for all attributes at in Ω_S, the values $\tau(at)$ are pairwise distinct.

Let r be the randomness in the cloud server's outsourcing key and γ be part of the master secret key held by the trusted attribute authority. The cloud server receives the tuple $\{g^{\frac{r}{\gamma+\tau(at)}}, \tau(at)\}_{at\in\Omega}$ in its outsourcing key. The outsourced signature generation requires the cloud server to pick the components $g^{\frac{r}{\gamma+\tau(at)}}$ and $\tau(at)$ for $at \in \Omega_S \subseteq \Omega$ and to compute $g^{\frac{r}{\prod_{at\in\Omega_S}(\gamma+\tau(at))}} \in \mathbb{G}$ given these components. This is done using the algorithm Aggregate [14].

Let $x_i = \tau(at)$ for all $at \in \Omega_S$ where $|\Omega_S| = t$. Given $\nu_i = g^{\frac{r}{\gamma+x_i}}$ and x_i for $i \in [1,t]$, for any (j,l) such that $1 \leq j < l \leq t$,

$$L_{j,l} = \nu_l^{\frac{1}{\prod_{i=1}^{j}(\gamma+x_i)}} = (g^{\frac{r}{(\gamma+x_l)}})^{\frac{1}{\prod_{i=1}^{j}(\gamma+x_i)}}.$$

The Aggregate algorithm consists in computing sequentially $L_{j,l}$ for $j \in [1, t-1]$ and $l \in [j+1, t]$ using the induction $L_{j,l} = (L_{j-1,j}/L_{j-1,l})^{\frac{1}{x_l - x_j}}$ and setting $L_{0,l} = \nu_l$ for $l \in [1,t]$. The algorithm finally outputs $L_t = L_{t-1,t}$.

The Threshold ABS scheme in [6] uses polynomial fractions to enable the pairing-based verification process. Informally, attributes at the denominator will cancel out if and only if the signer has t attributes matching the ones in the access policy. The algorithm Aggregate [14] enables to compute the denominator of these polynomial fractions by obtaining $g^{\frac{r}{F_{\Omega_S}(\gamma)}} \in \mathbb{G}$, while the numerator is calculated given the public parameters and the attributes in S.

Let the polynomial $F_{(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}(\gamma)$ from [6] be defined as follows:

$$\begin{aligned} &F_{(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}(\gamma) \\ &= \frac{\prod_{at\in S}(\gamma+\tau(at))\prod_{at\in\mathcal{D}_{n+t-1-s}}(\gamma+\tau(at))}{\prod_{at\in\Omega_S}(\gamma+\tau(at))} \\ &= \prod_{at\in(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}(\gamma+\tau(at)) \end{aligned}$$

The set $\mathcal{D}_{n+t-1-s}$ corresponds to the set of dummy attributes. When $\Omega_S = \Omega \cap S$ has exactly t elements, then the degree of the polynomial $F_{(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}(\gamma)$ is $s+(n+t-1-s)-t = n-1$. Let a_i be the coefficients of the polynomial. We re-write as:

$$F_{(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}(\gamma) = \sum_{i=0}^{n-1}\gamma^i a_i = \sum_{i=1}^{n-1}\gamma^i a_i + a_0$$

where $a_0 = \prod_{at\in(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}\tau(at)$. The private key contains the components $h^{r\gamma^i}$ for $i \in [0, n-2]$, and thus the element $h^{\frac{1}{\gamma}r(F_{(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}(\gamma)-a_0)} = h^{r\sum_{i=1}^{n-1}\gamma^{i-1}a_i} = \prod_{i=1}^{n-1}(h^{r\gamma^{i-1}})^{a_i} = \prod_{j=0}^{n-2}(h^{r\gamma^j})^{a_{j+1}}$ is computable. However, if $|\Omega_S| < t$, then the polynomial $F_{(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}(\gamma)$ has degree strictly greater than $n-1$. Thus, the element $h^{\frac{1}{\gamma}r(F_{(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}(\gamma)-a_0)}$ is not computable since the components $h^{r\gamma^i}$ for $i > n-2$ are missing in the private key.

Yet, the use of dummy attributes to enable the correctness of the polynomial setting brings extra storage and computational costs. Similarly to [8], we modify the Herranz et al.'s polynomial setting in a such way that dummy attributes are no longer required. Let the polynomial $F_{S\setminus\Omega_S}(\gamma)$ in CHARIOT be defined as follows:

$$F_{S\setminus\Omega_S}(\gamma) = \frac{\prod_{at\in S}(\gamma+\tau(at))}{\prod_{at\in\Omega_S}(\gamma+\tau(at))} = \prod_{at\in S\setminus\Omega_S}(\gamma+\tau(at))$$

When $\Omega_S = \Omega \cap S$ has exactly t elements, then the degree of the polynomial $F_{S\setminus\Omega_S}(\gamma)$ is $s-t$. Let b_i be the elements of the polynomial. We re-write as:

$$F_{S\setminus\Omega_S}(\gamma) = \sum_{i=0}^{s-t}\gamma^i b_i = \gamma^{s-t}b_{s-t} + \sum_{i=0}^{s-t-1}\gamma^i b_i$$

where $b_{s-t} = 1$. Given the index $i \in [0, s-t-1]$, we observe that $(i+n-s+t) \in [n-s+t, n-1]$ such that $n-s+t > 0$. In addition, $\gamma^{n-s+t}r(F_{S\setminus\Omega_S}(\gamma) - \gamma^{s-t}) = \gamma^{n-s+t}r(\gamma^{s-t} + \sum_{i=0}^{s-t-1}\gamma^i b_i - \gamma^{s-t}) = \gamma^{n-s+t}r(\sum_{i=0}^{s-t-1}\gamma^i b_i) = r(\sum_{i=0}^{s-t-1}\gamma^{i+n-s+t}b_i)$. The private key contains the components $h^{r\gamma^i}$ for $i \in [1, n-1]$, and thus the element $h^{\gamma^{n-s+t}r(F_{S\setminus\Omega_S}(\gamma)-\gamma^{s-t})} = \prod_{i=0}^{s-t-1}(h^{r\gamma^{i+n-s+t}})^{b_i}$ is computable. However, if $|\Omega_S| < t$, then the polynomial $F_{S\setminus\Omega_S}(\gamma)$ has degree strictly greater than $s-t$. Thus, the element $h^{\gamma^{n-s+t}r(F_{S\setminus\Omega_S}(\gamma)-\gamma^{s-t})} = \prod_{i=0}^{s-t-1}(h^{r\gamma^{i+n-s+t}})^{b_i}$ is not computable since components $h^{r\gamma^i}$ for $i > n-1$ are missing in the private key.

Outsourced Signature Generation. The off-line trusted attribute authority generates two keys regarding the set of the device's attributes by picking at random $\beta_1 \in_R \mathbb{Z}_p$ and using its master secret key embedding $\beta \in \mathbb{Z}_p$. The first key is an outsourcing key embedding $\beta_2 = \beta + \beta_1$, that is forwarded to the cloud server. The second key is a private key embedding β_1, that is kept secret by the IoT device. The cloud server computes an outsourced signature using its outsourcing key, and thus β_2, and the current signing policy. Then, the device finalizes the signing process by choosing the message and using its private key, and thus β_1. The exponent β_2 is canceled out and the final signature contains $\beta = \beta_2 - \beta_1$. The verification is hence possible since it requires the public parameters, whose components embed β, and the signature of the signing device. The device's signature is finally checked by a verifier: the signature is valid for the chosen message if device's attributes embedded in the signature satisfy the signing policy.

However, in [7], the cloud server must know the attributes of the signer (included in the outsourcing key) and the attributes of the signing policy when generating outsourced signatures. In addition, only outsider attackers are considered and do not enclose the cloud server when proving attribute privacy guarantee in [7]. Contrary, we suggest that the cloud server should be considered as untrusted and thus not have any

information about the attributes of the devices and of the signing policies. We use an HMAC τ such that inputs at are warranted to be authentic and have not been modified. We assume that τ sends an attribute $at \in \mathcal{P}$ onto an element $\tau(at) \in \mathbb{Z}_p^*$ such that all values $\tau(at)$ are different, and hides the attributes from the cloud server's view. The HMAC τ is shared among the attribute authority and the device, and used for generating the outsourcing key (such that all the device's attributes are hashed using HMAC) and the HMAC-hashed signing policy for the cloud server respectively. Hence, the latter computes the outsourced signature such that it does not learn anything from the attributes unless whether t attributes match between the signing policy and the device's attribute set.

D. Overview

In this section, we give a sight of our protocol in which three main phases occur. The first one enables to set up the protocol and to generate the public parameters and key material. The second phase lets a device ask for access to the IoT platform by outsourcing the computation of its request to a cloud server. The platform verifies the request and access rights of the device, and gives authorization for access accordingly.

Protocol Setup. First, a setup algorithm is run once to initiate the CHARIOT protocol. It generates the public parameters made available to all the participating parties, along with the master secret key given to the off-line trusted attribute authority. Then, the latter creates an outsourcing key delivered to the cloud server, a unique private key forwarded to the IoT device and a secret key given to the IoT platform. The outsourcing key embeds the attributes of the device in their HMAC-hashed form while the private key is calculated in order to enable the device to complete its access requests to the platform. Informally, the outsourcing key and the device's private key contain shares of a secret enabling to apply the outsourced signature generation technique.

Access Request. When the device wants to access the IoT platform, it first asks the cloud server for some assistance. Given the outsourcing key and the current (HMAC-hashed) policy, the cloud server generates an outsourced request, under the form of a signature, and forwards it to the device. The latter modifies it into its final request by using its private key and by choosing a personal message. It finally sends its request to the platform.

Outsourced and final signatures are constructed as in the threshold case, based on polynomial fractions defined over the attributes of the device and of the access policy. By doing so, if the device holds less attributes than a certain threshold, the signature verification will fail and the device will not have access to the platform. Contrary to [6], the signature generation does not require the use of dummy attributes into polynomials to reach correctness of the verification process. Moreover, signatures embed Groth-Sahai proof systems allowing the device to implicitly prove that it can decrypt a ciphertext corresponding to the ABE scheme [9].

Access Verification. Once receiving an access request from the IoT device, the platform checks the validity of the corresponding signature using the public parameters, the current policy and the chosen message. If the result is positive, meaning that the device has the required attributes satisfying the policy, then it is authorized for access.

The verification phase works thanks to the correctness of the ABE scheme [9] and to the perfect completeness, soundness and composable zero-knowledge of the Groth-Sahai proofs.

E. Construction

The CHARIOT construction is made of six algorithms run among a trusted attribute authority, an IoT platform, a cloud server and an IoT device:

• $\mathsf{Setup}(\lambda, \mathcal{P}, n) \rightarrow (params, msk)$**.** On inputs the security parameter λ, an attribute universe $\mathcal{P}$ and an integer n that is an upper bound on the size of threshold policies, the algorithm outputs the public parameters $params$ (which contain $(\lambda, \mathcal{P}, n)$) and the master secret key msk (for the trusted attribute authority) as follows:

The algorithm first chooses two cyclic groups $\mathbb{G}, \mathbb{G}_T$ of prime order $p > 2^\lambda$ with an efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Let g, h be two generators of $\mathbb{G}$ and $H : \{0,1\}^* \rightarrow \{0,1\}^k$ be a collision-resistant hash function for some k. Let τ be a HMAC that, given a key K, sends an attribute $at \in \mathcal{P}$ onto an element $\tau(K, at) \in \mathbb{Z}_p^*$ such that all output values are different. The algorithm then picks $\alpha, \beta, \gamma \in_R \mathbb{Z}_p^*$ and computes $u = g^\beta$, $v_i = g^{\frac{\alpha}{\gamma^i}}$ and $h_i = h^{\alpha\gamma^i}$ for $i \in [0, n]$. It generates the Groth-Sahai CRS by first choosing two generators g_1, g_2 of $\mathbb{G}$. Then, it defines the vectors $\vec{g}_1 = (g_1, 1, g)^\top$ and $\vec{g}_2 = (1, g_2, g)^\top$. For $i \in [0, k]$, it picks $\xi_{i,1}, \xi_{i,2} \in_R \mathbb{Z}_p$ and defines the vector $\vec{g}_{3,i} = (\vec{g}_1)^{\xi_{i,1}} \cdot (\vec{g}_2)^{\xi_{i,2}} = (g_1^{\xi_{i,1}}, g_2^{\xi_{i,2}}, g^{\xi_{i,1}+\xi_{i,2}})^\top$. Exponents $\{\xi_{i,1}, \xi_{i,2}\}_{i\in[0,k]}$ can then be discarded since they are no longer needed.

Finally, the algorithm sets the public parameters $params = (\lambda, \mathcal{P}, n, p, \mathbb{G}, \mathbb{G}_T, e, g, h, u, \{v_i\}_{i\in[0,n]}\{h_i\}_{i\in[0,n]}, \vec{g}_1, \vec{g}_2, \{\vec{g}_{3,i}\}_{i\in[0,k]}, H, \tau)$ and the master secret key $msk = (\alpha, \beta, \gamma)$.

• $\mathsf{KeyGen}(params, msk, \Omega) \rightarrow (osk_\Omega, sk_\Omega, sk_{PT})$**.** On inputs the public parameters $params$, the master secret key msk, an attribute set $\Omega \subset \mathcal{P}$, the trusted attribute authority outputs the outsourcing key osk_Ω (for the cloud server), the private key sk_Ω (for the IoT device) and the secret key sk_{PT} (for the IoT platform) as follows:

Let K be a random key to be shared between the device and the platform, that will be used to hash the attributes. Let $\Omega \subset \mathcal{P}$ be an attribute set. The authority picks $\beta_1 \in_R \mathbb{Z}_p^*$ and sets $\beta_2 = \beta + \beta_1$. It then chooses $r \in_R \mathbb{Z}_p^*$ and computes $g^{\frac{r}{\gamma+\tau(K,at)}}$ for $at \in \Omega$, $h^{r\gamma^i}$ for $i \in [1, n-1]$, $h^{(r-\beta_2)\gamma^n}$, g^{β_1} and $h^{\beta_1\gamma^n}$.

The authority sets the outsourcing key $osk_\Omega = (\{g^{\frac{r}{\gamma+\tau(K,at)}}, \tau(K,at)\}_{at\in\Omega},\ \{h^{r\gamma^i}\}_{i\in[1,n-1]}, h^{(r-\beta_2)\gamma^n}, g^{\beta_1})$ for the cloud server, the private key $sk_\Omega = (h^{\beta_1\gamma^n}, K)$ for the IoT device and the secret key $sk_{PT} = K$ for the IoT platform.

• $\mathsf{Request}(\Gamma, sk_\Omega) \rightarrow \tilde{\Gamma}$**.** On inputs a threshold signing policy $\Gamma = (t, S)$ where the set $S \subset \mathcal{P}$ has $|S| = s \leq n$ attributes and $1 \leq t \leq s$ and the private key sk_Ω, the IoT device hashes each attribute $at \in S$ with τ resulting into $\tau(K, at)$, and creates the HMAC-hashed set $\tilde{S}$ containing the values $\tau(K, at)$ for all $at \in S$. It sets the HMAC-hashed threshold signing policy $\tilde{\Gamma} = (t, \tilde{S})$ and forwards it to the cloud server.

• $\mathsf{Sign}_{out}(params, osk_\Omega, \tilde{\Gamma}) \rightarrow \sigma'$**.** On inputs the public parameters $params$, the outsourcing key osk_Ω and an HMAC-hashed threshold signing policy $\tilde{\Gamma} = (t, \tilde{S})$ where $\tilde{S}$ is the HMAC-hashed set of $S \subset \mathcal{P}$ and $1 \leq t \leq s \leq n$, the cloud server outputs an outsourced signature σ' as follows:

The cloud server returns 1 if $|\Omega \cap S| < t$; otherwise, it finds a subset $\Omega_S \subset \Omega \cap S$ such that $|\Omega_S| = t$. The cloud server works on the HMAC-hashed sets to verify the number of attributes contained in the intersection, since it should not get any information about the attributes in Ω and S except that there are at least t matching attributes.

For all $at \in \Omega_S$, the cloud server then runs the algorithm $\mathsf{Aggregate}(\{g^{\frac{r}{\gamma+\tau(K,at)}}, \tau(K,at)\}_{at\in\Omega_S}) = g^{\frac{r}{\prod_{at\in\Omega_S}(\gamma+\tau(K,at))}} = g^{\frac{r}{F_{\Omega_S}(\gamma)}} = T_1$. Let the polynomial $F_{S\setminus\Omega_S}(\gamma)$ be as $F_{S\setminus\Omega_S}(\gamma) = \prod_{at\in S\setminus\Omega_S}(\gamma + \tau(K, at)) = \sum_{i=0}^{s-t}\gamma^i b_i$ such that $b_{s-t} = 1$. Then, it computes $T_2' = h^{(r-\beta_2)\gamma^n} \cdot \prod_{i=0}^{s-t-1}(h^{r\gamma^{i+n-s+t}})^{b_i}$ that is possible given the public parameters $params$ and the outsourcing key osk_Ω. The obtained values T_1 and T_2' should satisfy the equality:

$$e(T_2', v_{n-s+t}^{-1}) \cdot e(T_1, h^{\alpha F_S(\gamma)}) = e(u \cdot g^{\beta_1}, h_{s-t})$$

Then, it picks $r_1, s_1, r_2, s_2 \in_R \mathbb{Z}_p$ and computes $\vec{C}'_{T_1} = (1, 1, T_1)^\top \cdot (\vec{g}_1)^{r_1} \cdot (\vec{g}_2)^{s_1}$ and $\vec{C}'_{T_2} = (1, 1, T_2')^\top \cdot (\vec{g}_1)^{r_2} \cdot (\vec{g}_2)^{s_2}$. Let $\theta \in \mathbb{G}$ with commitment $\vec{C}'_\theta = (1, 1, \theta)^\top \cdot (\vec{g}_1)^{r_\theta} \cdot (\vec{g}_2)^{s_\theta}$ for $r_\theta, s_\theta \in_R \mathbb{Z}_p$, which takes $\theta = h_{s-t}$ and proves the following:

$$\begin{aligned} e(T_1, H_S) &= e(u \cdot g^{\beta_1}, \theta) \cdot e(T_2', v_{n-s+t}) && (1)\\ e(g, \theta) &= e(g, h_{s-t}) && (2)\end{aligned}$$

where $H_S = h^{\alpha F_S(\gamma)} = h^{\alpha \prod_{at\in S}(\gamma+\tau(K,at))}$. Equations 1 and 2 are called proofs $\vec{\pi}'_1$ and $\vec{\pi}'_2$ respectively, and are given by $\vec{\pi}'_1 = (H_S^{r_1} \cdot (u g^{\beta_1})^{-r_\theta} \cdot v_{n-s+t}^{-r_2}, H_S^{s_1} \cdot (u g^{\beta_1})^{-s_\theta} \cdot v_{n-s+t}^{-s_2}, 1)^\top$ and $\vec{\pi}'_2 = (g^{r_\theta}, g^{s_\theta}, 1)^\top$. It also computes $g^{\beta_1 r_\theta}$ and $g^{\beta_1 s_\theta}$.

Finally, the cloud server sets the outsourced signature $\sigma' = (\vec{C}'_{T_1}, \vec{C}'_{T_2}, \vec{C}'_\theta, \vec{\pi}'_1, \vec{\pi}'_2, T_2', H_S, g^{\beta_1 r_\theta}, g^{\beta_1 s_\theta})$ and forwards it to the IoT device.

• $\mathsf{Sign}(params, sk_\Omega, \mathcal{M}, \sigma') \rightarrow \sigma$**.** On inputs the public parameters $params$, the private key sk_Ω, a message $\mathcal{M}$ and an outsourced signature σ', the IoT device outputs a signature σ as follows:

Given T_2' from the outsourced signature σ' and $h^{\beta_1\gamma^n}$ from the private key sk_Ω, the IoT device computes $T_2 = T_2' \cdot h^{\beta_1\gamma^n} = h^{(r-\beta_2)\gamma^n} \cdot \prod_{i=0}^{s-t-1}(h^{r\gamma^{i+n-s+t}})^{b_i} \cdot h^{\beta_1\gamma^n} = h^{(r-\beta)\gamma^n} \cdot \prod_{i=0}^{s-t-1}(h^{r\gamma^{i+n-s+t}})^{b_i}$. The obtained values T_1 and T_2 should satisfy the equality:

$$e(T_2, v_{n-s+t}^{-1}) \cdot e(T_1, h^{\alpha F_S(\gamma)}) = e(u, h_{s-t}) \quad (3)$$

Thereafter, it computes $M = m_1 \cdots m_k = H(\mathcal{M}) \in \{0,1\}^k$. It uses M to form a message-specific Groth-Sahai CRS $\mathbf{g_M} = (\vec{g}_1, \vec{g}_2, \vec{g}_{3,M})$. More specifically, for all $i \in [0, k]$, $\vec{g}_{3,i}$ is parsed as $(g_{X,i}, g_{Y,i}, g_{Z,i})^\top$ and the device sets $\vec{g}_{3,M} = (g_{X,0} \cdot \prod_{i=1}^k g_{X,i}^{m_i}, g_{Y,0} \cdot \prod_{i=1}^k g_{Y,i}^{m_i}, g_{Z,0} \cdot \prod_{i=1}^k g_{Z,i}^{m_i})^\top$. The device then generates the Groth-Sahai commitments to the values T_1 and T_2 using $\mathbf{g_M}$. It picks $t_1, t_2 \in_R \mathbb{Z}_p$ and computes:

$$\begin{aligned} \vec{C}_{T_1} &= \vec{C}'_{T_1} \cdot (\vec{g}_{3,M})^{t_1} \\ &= (1, 1, T_1)^\top \cdot (\vec{g}_1)^{r_1} \cdot (\vec{g}_2)^{s_1} \cdot (\vec{g}_{3,M})^{t_1} \\ \vec{C}_{T_2} &= \vec{C}'_{T_2} \cdot (1, 1, h^{\beta_1\gamma^n})^\top \cdot (\vec{g}_{3,M})^{t_2} \\ &= (1, 1, T_2)^\top \cdot (\vec{g}_1)^{r_2} \cdot (\vec{g}_2)^{s_2} \cdot (\vec{g}_{3,M})^{t_2} \end{aligned}$$

Then, it generates the NZIK proof that the pair of committed variables (T_1, T_2) satisfies the pairing-product in Equation 3. To do so, let the commitment $\vec{C}_\theta = \vec{C}'_\theta \cdot (\vec{g}_{3,M})^{t_\theta} = (1, 1, \theta)^\top \cdot (\vec{g}_1)^{r_\theta} \cdot (\vec{g}_2)^{s_\theta} \cdot (\vec{g}_{3,M})^{t_\theta}$ for $t_\theta \in_R \mathbb{Z}_p$, which takes $\theta = h_{s-t}$ and proves the following:

$$\begin{aligned} e(T_1, H_S) &= e(u, \theta) \cdot e(T_2, v_{n-s+t}) && (4)\\ e(g, \theta) &= e(g, h_{s-t}) && (5)\end{aligned}$$

where $H_S = h^{\alpha F_S(\gamma)}$. Equations 4 and 5 are called proofs $\vec{\pi}_1$ and $\vec{\pi}_2$ respectively, and are given by:

$$\begin{aligned} \vec{\pi}_1 &= \vec{\pi}'_1 \cdot (g^{-\beta_1 r_\theta}, g^{-\beta_1 s_\theta}, H_S^{t_1} \cdot u^{-t_\theta} \cdot v_{n-s+t}^{-t_2})^\top \\ &= (H_S^{r_1} \cdot u^{-r_\theta} \cdot v_{n-s+t}^{-r_2}, H_S^{s_1} \cdot u^{-s_\theta} \cdot v_{n-s+t}^{-s_2}, \\ &\quad H_S^{t_1} \cdot u^{-t_\theta} \cdot v_{n-s+t}^{-t_2})^\top \\ \vec{\pi}_2 &= \vec{\pi}'_2 \cdot (1, 1, g^{t_\theta})^\top = (g^{r_\theta}, g^{s_\theta}, g^{t_\theta})^\top \end{aligned}$$

Finally, the IoT device sets the signature $\sigma = (\vec{C}_{T_1}, \vec{C}_{T_2}, \vec{C}_\theta, \vec{\pi}_1, \vec{\pi}_2)$ and sends it to the IoT platform.

• $\mathsf{Verify}(params, sk_{PT}, \mathcal{M}, \sigma, \Gamma) \rightarrow 0/1$**.** On inputs the public parameters $params$, the secret key sk_{PT}, a message $\mathcal{M}$, a signature σ, a threshold policy $\Gamma = (t, S)$, the IoT platform outputs 0 if the signature is valid and 1 otherwise.

The IoT platform computes $M = m_1 \cdots m_k = H(\mathcal{M})$, forms the vector $\vec{g}_{3,M} = (g_{X,0} \cdot \prod_{i=1}^k g_{X,i}^{m_i}, g_{Y,0} \cdot \prod_{i=1}^k g_{Y,i}^{m_i}, g_{Z,0} \cdot \prod_{i=1}^k g_{Z,i}^{m_i})^\top$, and sets $H_S = h^{\alpha F_S(\gamma)} = h^{\alpha \prod_{at\in S}(\gamma+\tau(K,at))}$ where $K = sk_{PT}$. Let $\vec{\pi}_j = (\pi_{j,1}, \pi_{j,2}, \pi_{j,3})^\top$ for $j \in \{1, 2\}$. It returns 0 if and only if:

$$\begin{aligned} E(H_S, \vec{C}_{T_1}) &= E(u, \vec{C}_\theta) \cdot E(v_{n-s+t}, \vec{C}_{T_2}) \cdot E(\pi_{1,1}, \vec{g}_1) \\ &\quad \cdot E(\pi_{1,2}, \vec{g}_2) \cdot E(\pi_{1,3}, \vec{g}_{3,M}) \\ E(g, \vec{C}_\theta) &= E(g, (1, 1, h_{s-t})) \cdot E(\pi_{2,1}, \vec{g}_1) \\ &\quad \cdot E(\pi_{2,2}, \vec{g}_2) \cdot E(\pi_{2,3}, \vec{g}_{3,M}) \end{aligned}$$

Correctness. For any $\lambda \in \mathbb{N}$, any integer n, any universe $\mathcal{P}$, any public parameters and master secret key $(params, msk) \leftarrow \mathsf{Setup}(\lambda, \mathcal{P}, n)$, any set $\Omega \subset \mathcal{P}$, any threshold policy $\Gamma = (t, S)$ where $1 \leq t \leq |S| \leq n$, and any message $\mathcal{M}$, it is required that $\mathsf{Verify}(params, sk_{PT}, \mathcal{M}, \mathsf{Sign}(params, sk_\Omega, \mathcal{M}, \mathsf{Sign}_{out}(params, osk_\Omega, \mathsf{Request}(\Gamma, sk_\Omega))), \Gamma) = 0$ whenever $(osk_\Omega, sk_\Omega, sk_{PT}) \leftarrow \mathsf{KeyGen}(params, msk, \Omega)$ and $|\Omega \cap S| \geq t$. The correctness is demonstrated based on the one for Groth-Sahai proofs [10]. We verify the correctness of Equation 3:

$$
\begin{aligned}
& e(T_2, v_{n-s+t}^{-1}) \cdot e(T_1, h^{\alpha F_S(\gamma)}) \\
= \; & e(h^{(r-\beta)\gamma^n} \cdot \prod_{i=0}^{s-t-1} (h^{r\gamma^{i+n-s+t}})^{b_i}, g^{\frac{-\alpha}{\gamma^{n-s+t}}}) \\
& \cdot e(g^{\frac{r}{F_{\Omega_S}(\gamma)}}, h^{\alpha F_S(\gamma)}) \\
= \; & e(h^{(r-\beta)\gamma^n} \cdot \prod_{i=0}^{s-t-1} ((h^{r\gamma^i})^{\gamma^{n-s+t}})^{b_i}, g^{\frac{-\alpha}{\gamma^{n-s+t}}}) \\
& \cdot e(g, h)^{\alpha r F_{S \setminus \Omega_S}(\gamma)} \\
= \; & e(h^{(r-\beta)\gamma^{s-t}} \cdot \prod_{i=0}^{s-t-1} (h^{r\gamma^i})^{b_i}, g^{-\alpha}) \\
& \cdot e(g, h)^{\alpha r F_{S \setminus \Omega_S}(\gamma)} \\
= \; & e(h^{r\gamma^{s-t}} \cdot h^{(-\beta)\gamma^{s-t}} \cdot \prod_{i=0}^{s-t-1} (h^{r\gamma^i})^{b_i}, g^{-\alpha}) \\
& \cdot e(g, h)^{\alpha r F_{S \setminus \Omega_S}(\gamma)} \\
= \; & e(h^{(-\beta)\gamma^{s-t}} \cdot \prod_{i=0}^{s-t} (h^{r\gamma^i})^{b_i}, g^{-\alpha}) \\
& \cdot e(g, h)^{\alpha r F_{S \setminus \Omega_S}(\gamma)} [\text{where } b_{s-t} = 1] \\
= \; & e(g, h)^{\alpha\beta\gamma^{s-t}} \cdot e(g, h)^{-\alpha r F_{S \setminus \Omega_S}(\gamma)} \\
& \cdot e(g, h)^{\alpha r F_{S \setminus \Omega_S}(\gamma)} \\
= \; & e(g, h)^{\alpha\beta\gamma^{s-t}} = e(u, h_{s-t})
\end{aligned}
$$

F. Security Analysis

Unforgeability. The CHARIOT protocol can be proven selective-policy and adaptive-message unforgeable under chosen-message attacks assuming that H is a collision-resistant hash function, the DLIN problem holds in $\mathbb{G}$ and the (q, n, s^*, t^*)-aMSE-CDH problem holds in $(\mathbb{G}, \mathbb{G}_T)$. The adversary $\mathcal{A}$ (group of colluding parties that put their outsourcing/private keys together) selects the signing policy $\Gamma^* = (t^*, S^*)$ that it wishes to attack at the beginning of the game, while the message $\mathcal{M}^*$ linked to an eventually forged signature is not chosen in advance. $\mathcal{A}$ can query for valid signatures on messages and signing policies of its choice. We give a sketch of the proof and let the reader refer to [6], [8], [15] for more details.

Let $\vec{g}_1, \vec{g}_2, \{\vec{g}_{3,i}\}_{i \in [1,k]}$ be a modified distribution of the vectors that are inputs for the Groth-Sahai CRS generation. Let an integer $\nu \in_R [0, k]$, exponents $\xi_{i,1}, \xi_{i,2} \in_R \mathbb{Z}_p$ and integers $\rho_i \in_R [0, 2q_S - 1]$ be randomly chosen for $i \in [0, k]$ where q_S is the number of signature queries. Then, let a, b be picked at random in $\mathbb{Z}_p$, $g_1 = g^a$, $g_2 = g^b$, and $\vec{g}_1 = (g_1, 1, g)^\top$ and $\vec{g}_2 = (1, g_2, g)^\top$. Let $\vec{g}_{3,0} = (\vec{g}_1)^{\xi_{0,1}} \cdot (\vec{g}_2)^{\xi_{0,2}} \cdot ((1, 1, g)^\top)^{\nu \cdot 2q_S - \rho_0}$ and $\vec{g}_{3,i} = (\vec{g}_1)^{\xi_{i,1}} \cdot (\vec{g}_2)^{\xi_{i,2}} \cdot ((1, 1, g)^\top)^{\rho_i}$ for $i \in [1, k]$. In the way that the tuple $(\vec{g}_1, \vec{g}_2, \{\vec{g}_{3,i}\}_{i \in [1,k]})$ is calculated, a successful adversary $\mathcal{A}$ against unforgeability will successfully help: i) either an algorithm to find collisions on the hash function H; ii) or an algorithm to solve the DLIN problem; iii) or a challenger $\mathcal{B}$ interacting with $\mathcal{A}$ to solve the (q, n, s^*, t^*)-aMSE-CDH problem.

For the item iii), let q be the cardinality of the attribute universe $\mathcal{P}$, n be the upper bound on the size of threshold policies, s^* be the cardinality of the target attribute set S^* and t^* be the target threshold value.

Initialization: $\mathcal{B}$ specifies the universe $\mathcal{P}$ of attributes while $\mathcal{A}$ chooses the signing policy $\Gamma^* = (t^*, S^*)$.

Setup: $\mathcal{B}$ receives the elements from the aMSE-CDH assumption such that it encodes each attribute from $\mathcal{P}$ with either the polynomial $f(x)$ for $at \notin S^*$ or $g(x)$ for $at \in S^*$.

Outsourcing Key and Private Key Queries: The exponent r is calculated with a random exponent $r' \in_R \mathbb{Z}_p$ and the polynomial $F_{\Omega_{S^*}}(\gamma)$ where $\Omega_{S^*} = S^* \cap \Omega$ is known. $x + \tau(at_i) | f(x) F_{\Omega_{S^*}}(x)$ since at_i is either in $\Omega \setminus \Omega_{S^*}$ or in Ω_{S^*}

Signature Queries: $\mathcal{B}$ computes $M = H(\mathcal{M}) = m_1 \cdots m_k$, evaluates the functions:

- $J(M) = \nu \cdot 2q_S + \rho_0 + \sum_{i=1}^k \rho_i \cdot m_i$,
- $K_1(M) = \xi_{0,1} + \sum_{i=1}^k \xi_{i,1} \cdot m_i$ and
- $K_2(M) = \xi_{0,2} + \sum_{i=1}^k \xi_{i,2} \cdot m_i$,

and sets $\vec{g}_{3,M} = (\vec{g}_1)^{K_1(M)} \cdot (\vec{g}_2)^{K_2(M)} \cdot ((1, 1, g)^\top)^{-J(M)}$. If $J(M) = 0$, then $\mathcal{B}$ aborts; otherwise, it generates σ by simulating NZIK proofs.

Forgery: $\mathcal{A}$ outputs $(\mathcal{M}^*, \sigma^* = (\vec{C}_{T_1}^*, \vec{C}_{T_2}^*, \vec{C}_\theta^*, \vec{\pi}_1^*, \vec{\pi}_2^*), \Gamma^*)$. $\mathcal{B}$ then evaluates $J(M^*), K_1(M^*), K_2(M^*)$ for $M^* = H(\mathcal{M}^*) = m_1^* \cdots m_k^*$. If there is a signature query on $\mathcal{M}$ such that $\mathcal{M} \neq \mathcal{M}^*$ and $H(\mathcal{M}) = H(\mathcal{M}^*)$, then $\mathcal{B}$ aborts. This breaks the resistance against collisions of the hash function H. If $J(M^*) \neq 0$, then $\mathcal{B}$ aborts. $\mathbf{g}_{\mathbf{M}^*} = (\vec{g}_1, \vec{g}_2, \vec{g}_{3,M^*})$ is a perfectly sound Groth-Sahai CRS. When $J(M^*) = 0$, the NIZK proofs must be valid regarding $\mathbf{g}_{\mathbf{M}^*}$. Hence, $\mathbf{g}_{\mathbf{M}^*}, \vec{C}_{T_1}^*, \vec{C}_{T_2}^*$ are extractable Groth-Sahai commitments. Therefore, $\mathcal{B}$ obtains T_1 and T_2, and finds a solution for the aMSE-CDH problem.

Privacy. The CHARIOT protocol can be proven computationally private assuming that the DLIN problem holds in $\mathbb{G}$ and τ is a collision-resistant HMAC, ensuring that the only leaked information is that attributes satisfy the signing policy. We give a sketch of the proof and let the reader refer to [6] for more details. Let the adversary $\mathcal{A}$ (IoT platform, cloud server) wish to break the privacy of the CHARIOT construction by interacting with a challenger $\mathcal{B}$ that tries to solve the DLIN problem.

Game_0 is defined as the real privacy game given in [6]. When Setup is run, $\mathcal{B}$ chooses the vectors $\vec{g}_1, \vec{g}_2, \{\vec{g}_{3,i}\}_{i\in[0,k]}$ such that $\vec{g}_{3,i}$ is linearly dependent on $\vec{g}_1$ and $\vec{g}_2$, for $i \in [0, k]$.
Game_1 is similar to Game_0, except that $\vec{g}_{3,i}$ is linearly independent of $\vec{g}_1$ and $\vec{g}_2$, for $i \in [0, k]$.
The modification between the two games is indistinguishable if the DLIN problem is hard to solve [10]. In Game_1, $\vec{C}_{T_1}, \vec{C}_{T_2}, \vec{\pi}_1$ could possibly leak information about the selected attributes, but commitments and proofs are perfectly hiding since they reveal no information about the committed values T_1 and T_2. Hence, attributes' privacy is preserved in Game_1. Moreover, the outsourcing key does not leak anything about the attributes of the IoT device except that a certain number of them satisfy the signing policy. Indeed, the HMAC τ ensure that, given the key K secretly shared between the device and the IoT platform, $at \in \mathcal{P}$ is sent onto $\tau(K, at) \in \mathbb{Z}_p^*$ such that all output values are different. The function τ protects the attributes for data integrity and authenticity, and securely hides the attributes from the cloud server's view.

III. Performance Analysis

A. Performance Comparison

We compare the computational and storage costs between our CHARIOT scheme and the original Threshold ABS scheme [6]. In the following tables, "$\text{Exp}_{\mathbb{G}}$", "$\text{Mult}_{\mathbb{G}}$" and "$\text{Pair}_{\mathbb{G}_T}$" denote exponentiation and multiplication in $\mathbb{G}$ and pairing operation in $\mathbb{G}_T$ respectively. Let "n.a." mean "non available". Let n be the upper bound on the size of the policies and k be the parameter used for the Groth-Sahai proofs. Let Ω be the device's attribute set, S be the signing attribute set such that $s = |S|$, t be the threshold value and $\Omega_S = \Omega \cap S$ such that $|\Omega_S| = t$.

TABLE I
Computational cost

	Threshold ABS [6]		CHARIOT	
	$\text{Mult}_{\mathbb{G}}$	$\text{Exp}_{\mathbb{G}}$	$\text{Mult}_{\mathbb{G}}$	$\text{Exp}_{\mathbb{G}}$
Setup	-	$2n + 3k + 5$	-	$2n + 3k + 6$
KeyGen	-	$\lvert\Omega\rvert + n$	-	$n + \lvert\Omega\rvert + 2$
Sign_{out}	n.a.	n.a.	$s - t + 17$	$2s + t^2 - t + 32$
Sign	$d + s$ $-t + 17$ $+3k + 5$	$d + 2s$ $+t^2 - t + 32$ $+3k + 4$	$3k + 5$	$3k + 4$
Verify	$3k$	$s + 3k + 1$	$3k$	$s + 3k + 1$

In Table I, let d be the number of dummy attributes such that $d = n + t - 1 - s \leq n - 1$ as in [6]. The sign $-$ means no computation of the given operation. Using dummy attributes in [6] increases the number of multiplications and exponentiations in $\mathbb{G}$ during the signature generation. By removing dummy attributes, up to $n-1$ modular multiplications and exponentiations are cut for the CHARIOT signing phase. Then, in CHARIOT, the computations done by a device are limited to the Groth-Sahai proofs' ones; all other calculations are delegated to the cloud server. Therefore, with the assistance of a cloud server, the device is relieved from most of the signing computations.

TABLE II
CHARIOT storage cost

$params$	msk	osk_Ω	sk_Ω	σ
$-d-1$	$+1$	$+\lvert\Omega\rvert + n + 3$	$-\lvert\Omega\rvert - n + 2$	$=$

In Table II, only the storage difference in CHARIOT compared to the ABS scheme in [6] is taken into account: an element $+1$ (resp. an element -1) in one cell means that there is one extra element (resp. there is one less element) in the CHARIOT protocol compared to Herranz et al.'s protocol. The sign $=$ in a given column denotes that storage is identical in the two protocols regarding the component linked to this column. The number d of dummy attributes is equal to $n - 1$ as in [6]. Hence, in [6], a set of $n - 1$ dummy attributes should be stored in $params$ in order to let the device generate its signature, while in CHARIOT, such storage cost is saved. An extra secret element is required in msk in CHARIOT to enable the outsourcing of signing computations. The outsourcing key osk_Ω in CHARIOT is mainly the private key sk_Ω in [6]. The device in CHARIOT simply receives an extra element from the shared secret and the key for the HMAC τ as its private key sk_Ω. Signatures are of equal size in Herranz et al.'s ABS [6] and CHARIOT schemes, and independent of the number of attributes.

B. Performance Benchmarking

We evaluate the performance of our CHARIOT protocol by calculating the timings of each algorithm. We implement CHARIOT in Charm [16] based on Python language. We set two different configurations, one for the trusted attribute authority, the cloud server and the IoT platform, and another one for the IoT device. We assume that the attribute authority, the cloud server and the platform have similar noticeable resources, while the device has much less resources than these three parties. The experiments are tested:

- *Configuration* 1 *(for the attribute authority, cloud server and platform):* on a processor Intel Core i5-2500 CPU @3.30GHz $\times 4$ with RAM 16GiB and OS Linux Ubuntu 14.04 LTS;
- *Configuration* 2 *(for the device):* on a processor Genuine Intel(R) U2300 CPU @1.20GHz with RAM 2GiB and OS Linux Ubuntu 17.10.

Policies are supposed to contain up to 30 attributes regarding real scenarios [17], [18]. Hence, we choose an upper bound n equal to 30, a policy set S with $s = 15$ attributes, a threshold t equal to 13 and a device's attribute set Ω with 20 attributes. The parameter k is chosen regarding the bit-size of the hashed message. Since the message to be signed is public, the hash function does not require to be cryptographic but rather collision-resistant. We also suppose that the dictionary of messages picked by IoT devices is of finite and moderated size. Therefore, we evaluate k with the values 10, 20 and 40

in order to avoid collisions with high probability. The given timings are an average from 10 rounds of the CHARIOT protocol.

TABLE III
TIMINGS IN MILLISECONDS

		$k = 10$	$k = 20$	$k = 40$
Configuration 1 (attribute authority, cloud server and platform)	Setup	143.77	190.67	284.54
	KeyGen	76.19	75.47	75.17
	Sign_{out}	265.07	272.69	271.55
	Verify	65.66	108.93	194.12
Configuration 2 (device)	Request	0.16	0.16	0.17
	Sign	182.55	322.01	601.6

In Table III, the setup phase is costly but should be executed only once. It largely depends on the upper bound n on the size of threshold policies and on the parameter k used for Groth-Sahai proofs. The key generation phase is also performed only once by the trusted attribute authority, and is relatively fast since it only relies on n that should not exceed 30 in an IoT environment.

By outsourcing the signature generation to a cloud server, we speed up by three times the computational timing. Indeed, the cloud server approximately needs 270 milliseconds to run the algorithm Sign_{out} while the device would require 870 milliseconds to do the same (when setting Configuration 2 as for the device). We recall that the algorithm Sign_{out} does not depend on the parameter k, and thus does not variate with it. Most of the signature generation is thus done by the cloud server running the algorithm Sign_{out}. The finalization of the signature generation is accomplished by the device by running the algorithm Sign and remains less than 870 milliseconds (resulting from running the algorithm Sign_{out} with Configuration 2). However, timings for signature finalization and verification increase with the parameter k as these two phases largely rely on such parameter. Hence, k should not exceed 40 in order to keep the advantage of outsourcing the signature generation to a cloud server.

In the CHARIOT protocol, the parameter k refers to the bit-size of the hashed message. This parameter k should be selected such that collisions are highly avoided. If the dictionary of messages to be signed is really small, then $k = 10$ is optimal. Otherwise, if the dictionary is slightly bigger, then $k = 20$ is still a reasonable choice. In addition, signature finalization and verification take similar timings to be performed since they are composed of the same kinds of computations. The difference comes from the constant number (equal to 27) of pairing operations carried out when checking the validity of the signature. Because of the number of pairing computations required for verification, the Groth-Sahai proof systems appear to be inefficient. Blazy et al. [19] propose a significant reduction of the cost of Groth-Sahai proof systems by using batch verification techniques. The authors improve the verification cost up to four times the number of pairings per proof verification. We can integrate their batch verification techniques into the CHARIOT protocol to improve the timings of the verification phase.

C. Related Work

ABS. Maji et al. [20] explicitly introduce the notion of ABS and suggest several schemes with very expressive signing policies. The most practical one is only proven secure in the generic model, while the one with security in the standard model is not efficient since the signatures have size linear in the number of group elements in the security parameter. Okamoto et Takashima [21] propose an ABS construction where the size of the signatures grows linearly in the size of the span program, which is greater than the number of selected attributes in the signing policies. Other works on ABS are given in [5], [22], [23]. Subsequently, Okamoto and Takashima [24] present a fully secure ABS scheme that supports general non-monotone policies. More recently, Herranz et al. [6] suggest an ABS construction with threshold policies and constant-size signatures, which requires the presence of dummy attributes following the technique from Dynamic Threshold Public-Key Encryption [14].

Outsourcing Computations. Several papers deal with the problem of securely outsourcing expensive computations such as matrix multiplications and quadrature computations, edit distance computations, linear algebraic computations and linear programming computations [25]–[29]. Nevertheless, the aforementioned schemes are not suitable to alleviate the computational burdens that signers meet when generating signatures. Server-aided signature schemes [30]–[32] aim to decrease the computational cost due to exponentiation calculations by outsourcing the latter to a server. Nevertheless, these schemes are not suitable for access control management based on credentials. Mediated cryptographic protocols [33]–[35] require a partially trusted on-line server and provide efficient revocation but cannot be used to extend ABS into an outsourced ABS. More recently, Chen et al. [7] present two outsourced ABS schemes. While the first solution achieves a constant number of exponentiation calculations at the signer's side, the size of the signature remains linear in the number of selected attributes. The second solution obtains constant-size signatures; however, the number of exponentiations depends on the number of dummy attributes in addition to the number of selected attributes. Yet, the cloud server must know the attributes of the signers and of the signing policies in plain in order to proceed, compromising privacy.

IV. CONCLUSION

In this paper, we propose a new protocol for server-aided access control in IoT, called CHARIOT. The protocol's features comprise cloud server assistance, constant-size signature and no dummy attributes. By outsourcing most of the calculations to a cloud server for the signature generation, we manage to relieve the computational and communication costs at the IoT device's side. By removing the presence of dummy attributes in the protocol, we achieve to obtain a more efficient and practical ABS scheme compared to existing ones. Moreover, our protocol guarantees privacy and secure identity management of the involved parties. These contributions make

our scheme suitable for securely authorizing devices with constrained resources to access an IoT platform.

REFERENCES

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] R. H. Weber, "Internet of things - new security and privacy challenges," *Computer Law & Security Review*, vol. 26, no. 1, pp. 23 – 30, 2010.

[3] C. M. Medaglia and A. Serbanati, "An overview of privacy and security issues in the internet of things," in *Proceedings of the 20th Tyrrhenian Workshop on Digital Communications*, ser. The Internet of Things'10, 2010, pp. 389–395.

[4] J. Su, D. Cao, B. Zhao, X. Wang, and I. You, "ePASS: An expressive attribute-based signature scheme with privacy and an unforgeability guarantee for the internet of things," *Future Generation Computer Systems*, vol. 33, no. Supplement C, pp. 11 – 18, 2014.

[5] J. Li, M. H. Au, W. Susilo, D. Xie, and K. Ren, "Attribute-based signature and its applications," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS'10, 2010, pp. 60–69.

[6] J. Herranz, F. Laguillaumie, B. Libert, and C. Ràfols, "Short attribute-based signatures for threshold predicates," in *Proceedings of the 12th Conference on Topics in Cryptology*, ser. CT-RSA'12, 2012, pp. 51–67.

[7] X. Chen, J. Li, X. Huang, J. Li, Y. Xiang, and D. S. Wong, "Secure outsourced attribute-based signatures," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3285–3294, 2014.

[8] W. Susilo, G. Yang, F. Guo, and Q. Huang, "Constant-size ciphertexts in threshold attribute-based encryption without dummy attributes," *Information Sciences*, vol. 429, no. C, pp. 349–360, Mar. 2018.

[9] J. Herranz, F. Laguillaumie, and C. Ráfols, "Constant size ciphertexts in threshold attribute-based encryption," in *Proceedings of the conference on Public Key Cryptography*, ser. PKC'10, 2010, pp. 19–34.

[10] J. Groth and A. Sahai, "Efficient non-interactive proof systems for bilinear groups," in *Proceedings of the 27th Annual International Conference on Theory and Applications of Cryptographic Techniques*, ser. EUROCRYPT'08, 2008, pp. 415–432.

[11] T. Malkin, I. Teranishi, Y. Vahlis, and M. Yung, "Signatures resilient to continual leakage on memory and computation," in *Proceedings of the 8th Conference on Theory of Cryptography*, ser. TCC'11, 2011, pp. 89–106.

[12] B. Waters, "Efficient identity-based encryption without random oracles," in *Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques*, ser. EUROCRYPT'05, 2005, pp. 114–127.

[13] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Proceedings of the 24th Annual International Cryptology Conference*, ser. CRYPTO'04, 2004, pp. 41–55.

[14] C. Delerablée and D. Pointcheval, "Dynamic threshold public-key encryption," in *Proceedings of the 28th Annual Conference on Cryptology*, ser. CRYPTO'08, 2008, pp. 317–334.

[15] J. Herranz, F. Laguillaumie, B. Libert, and C. Ràfols, "Short attribute-based signatures for threshold predicates," HAL archives ouvertes - Report 00611651, 2012, https://hal.archives-ouvertes.fr/hal-00611651.

[16] J. A. Akinyele, M. D. Green, and A. D. Rubin, "Charm: A framework for rapidly prototyping cryptosystems," Cryptology ePrint Archive, Report 2011/617, 2011, https://eprint.iacr.org/2011/617.

[17] X. Yao, Z. Chen, and Y. Tian, "A lightweight attribute-based encryption scheme for the internet of things," *Future Generation Computer Systems*, vol. 49, no. Supplement C, pp. 104 – 112, 2015.

[18] M. Ambrosin, A. Anzanpour, M. Conti, T. Dargahi, S. R. Moosavi, A. M. Rahmani, and P. Liljeberg, "On the feasibility of attribute-based encryption on internet of things devices," *IEEE Micro*, vol. 36, no. 6, pp. 25–35, Nov 2016.

[19] O. Blazy, G. Fuchsbauer, M. Izabachène, A. Jambert, H. Sibert, and D. Vergnaud, "Batch groth-sahai," in *Proceedings of the 8th International Conference on Applied Cryptography and Network Security*, ser. ACNS'10, 2010, pp. 218–235.

[20] H. K. Maji, M. Prabhakaran, and M. Rosulek, "Attribute-based signatures," in *Proceedings of the 11th International Conference on Topics in Cryptology*, ser. CT-RSA'11, 2011, pp. 376–392.

[21] T. Okamoto and K. Takashima, "Homomorphic encryption and signatures from vector decomposition," in *Proceedings of the 2nd International Conference on Pairing-Based Cryptography*, ser. PAIRING'08, 2008, pp. 57–74.

[22] S. F. Shahandashti and R. Safavi-Naini, "Threshold attribute-based signatures and their application to anonymous credential systems," in *Proceedings of the 2Nd International Conference on Cryptology in Africa*, ser. AFRICACRYPT'09, 2009, pp. 198–216.

[23] J. Li and K. Kim, "Hidden attribute-based signatures without anonymity revocation," *Information Science*, vol. 180, no. 9, pp. 1681–1689, 2010.

[24] T. Okamoto and K. Takashima, "Efficient attribute-based signatures for non-monotone predicates in the standard model," in *Proceedings of the 14th International Conference on Practice and Theory in Public Key Cryptography*, ser. PKC'11, 2011, pp. 35–52.

[25] M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E. H. Spafford, "Secure outsourcing of scientific computations," Advances in Computers, Tech. Rep., 1998.

[26] M. J. Atallah and J. Li, "Secure outsourcing of sequence comparisons," *Information Security*, vol. 4, no. 4, pp. 277–287, 2005.

[27] D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing of algebraic computations," in *Proceedings of the 2008 Sixth Annual Conference on Privacy, Security and Trust*, ser. PST'08, 2008, pp. 240–245.

[28] M. J. Atallah and K. B. Frikken, "Securely outsourcing linear algebra computations," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS'10, 2010, pp. 48–59.

[29] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in *Proceedings of the International IEEE Conference on Computer Communications*, ser. INFOCOM'11, 2011, pp. 820–828.

[30] M. Jakobsson and S. Wetzel, "Secure server-aided signature generation," in *Proccedings of the 4th International Workshop on Practice and Theory in Public Key Cryptosystems*, ser. PKC'01, 2001, pp. 383–401.

[31] K. Bicakci and N. Baykal, "Server assisted signatures revisited," in *Proceedings of the 2004 International Conference in Topics in Cryptology*, ser. CT-RSA'04, 2004, pp. 143–156.

[32] ——, "Improved server assisted signatures," *Computer Networks*, vol. 47, pp. 351–366, 2005.

[33] D. Boneh, X. Ding, G. Tsudik, and C. M. Wong, "A method for fast revocation of public key certificates and security capabilities," in *Proceedings of the 10th Conference on USENIX Security Symposium*, ser. SSYM'01, 2001.

[34] D. Boneh, X. Ding, and G. Tsudik, "Fine-grained control of security capabilities," *ACM Transactions on Internet Technology*, vol. 4, no. 1, pp. 60–82, 2004.

[35] X. Ding, G. Tsudik, and S. Xu, "Leak-free group signatures with immediate revocation," in *Proceedings of the 24th International Conference on Distributed Computing Systems*, ser. ICDCS'04, 2004, pp. 608–615.