Practical and Scalable Access Control Mechanism for the Internet of Things using Time-bound Attribute-based Encryption

Clémentine Gritti[*], Emanuel Regnath[†], Sebastian Steinhorst[‡]

April 15, 2024

Abstract

Internet of Things (IoT) promises a strong connection between digital and physical environments. Nevertheless, such framework comes with huge security vulnerabilities, due to the heterogeneous nature of devices and of the diversity of their provenance. Furthermore, the resource constraints of weaker devices, such as sensors, require a lightweight design of security protocols.

In 2018, Liu et al. presented a new system with access control key updates and direct user revocation, that are beneficial features in IoT. Access control is done using Ciphertext-Policy Attribute-Based Encryption where attributes represent roles of devices within their networks and time validity ranges. In this paper, we propose an extension of this system by enabling several authorities to allocate those role attributes, to mitigate the key escrow problem. Moreover, we devise a novel approach, based on a binary tree, to append the time credentials. This allows us to find an interesting trade-off between key update frequency and user revocation list length, for stressing time-sensitive data exchanged in IoT environments. We adapt the security model to follow the multi-authority setting and prove our scheme secure under the Decisional Bilinear Diffie-Hellman Exponent assumption. Finally, we implement and evaluate of our solution, in order to confirm that the latter is fully deployable in IoT networks.

Keywords. Ciphertext-Policy Attribute-Based Encryption, Time-Based Key Update, User Revocation List, Access Control, Internet of Things.

[*]Eurecom, France, clementine.gritti@eurecom.fr

[†]Siemens, Munich, Germany, emanuel.regnath@siemens.com

[‡]Technical University of Munich, Germany, sebastian.steinhorst@tum.de

1 Introduction

New possibilities from the Internet of Things (IoT) technology are explored every day around the world. Complex combinations of hardware, sensors, data storage, microprocessors, software and ubiquitous connectivity are now included, moving beyond mechanical and electrical components. The mechanisms and tools for IoT offer better efficiency and productivity, but expand cyber vulnerabilities and threats along with technical challenges [5]. Devices forming IoT networks are heterogeneous in their functionality [38], which is implemented by various manufacturing origins, not always well defined, and have constrained computing and communication resources [42]. Moreover, these networks are dynamic, yielding the management even more demanding. 75 billion devices will be in the IoT world by 2025, and 127 new devices are connected every second to the Internet [37]. All of these characteristics make IoT dependability, in particular reliability and availability, challenging [32].

Yet, other concerns come with the purposes of developing IoT, that is capitalizing fresh precious information. Indeed, IoT devices continuously collect and exchange a huge amount of data, that is combined and refined through data analytics, and the resulting information takes on real value[1]. Cisco believes that IoT has produced more than 500 zettabytes of data per year from 2020, and that number grows exponentially[2]. In addition, to improve the accuracy of IoT systems, efforts must be made on data sharing. The main drawback is the rise of security and privacy threats [2, 24, 22].

In this paper, we are interested in developing an efficient access control system for secure data exchanges in IoT networks. Access control with identity management and authentication ensures that only authorized users are able to reach data. We aim to design a solution that takes into account data sharing concerns while overcoming IoT dependability issues. The extremely large number of IoT devices and the dynamicity of IoT networks force us to go beyond basic identity assignment techniques as for Public Key Infrastructure (PKI) [3]. Another issue comes with trivial key management where each device either receives a public/private key pair or shares a secret key with another device; in both cases the device should maintain a substantial number of keys in order to interact with other devices. Moreover, such techniques imply a centralized architecture, raising single point failures with unpredictable threats. Due to their ubiquity combined with the high configuration vulnerability, IoT devices have been involved in many cyber attacks [36, 23]. Therefore, revocation must be an essential option when elaborating a system. Then, it has been very important to achieve

[1] https://blog.equinix.com/blog/2018/02/21/the-rise-of-iot-data-exchanges/

[2] https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

low latency and high reliability for many IoT use cases [41]. Devices collect time-sensitive data in various situations, where either data batch processing would produce results too late to be useful or any application where latency is a concern. For instance, some control decisions in autonomous vehicles require sub-microsecond response times [33]. Industrial control systems require response in tens of microseconds to avoid damage and ensure safety. Temperature sensors must collect data once every few minutes and respond within a second. Electric metering requires frequent communication, low latency and high data rate [33]. Hence, an access control system should consider time as an essential feature to meet the aforementioned requirements.

This work introduces a fine grained access control scheme based on Attribute-Based Encryption (ABE), which remains lightweight and hence deployable in IoT networks. We design our system with key updates for access control and device revocation to overcome the aforementioned IoT security vulnerabilities. First, an access control based on roles permits to share collected data securely following the dynamicity of IoT networks. Devices are seen as users in our system, either encrypting data (owners) or decrypting it (requesters). Second, we encourage the participation of multiple authorities in charge of distributing key material to users based on their roles within authorities' environments, averting the key escrow problem. Third, we enable direct user revocation, thus always protecting sensitive data even if a user secret key is compromised. Then, we append time credentials in addition to role credentials, emphasizing the ephemeral value of shared data while enabling an interesting trade-off between reasonable key update frequency and moderate user revocation list length. Thus, our solution does not require recurrent communication between users and authorities and deletion of components to expunge existing keys to produce new keys. Having shorter periods of access allows faster expiration of corrupt device keys and could avoid the need for creating new key pairs. Moreover, such mechanism allows to control key corruption in IoT networks where device management is difficult, similarly to temporary website certificates in PKI. Our approach efficiently integrates a role and time-based access control scheme with IoT technologies, such that the outcome is fully implementable in the real world. We carefully prove the security of our scheme under the Decisional Bilinear Diffie-Hellman Exponent (BDHE) assumption. Moreover, we observe from the implementation of our system that the computational and communication results make it adjustable in IoT environments.

Figure 1 illustrates an example of our access control system in a smart home. Several temperature sensors are scattered in a house. They collect temperature data once every few minutes. They encrypt their time-sensitive data according to an access policy, containing roles and time periods. There

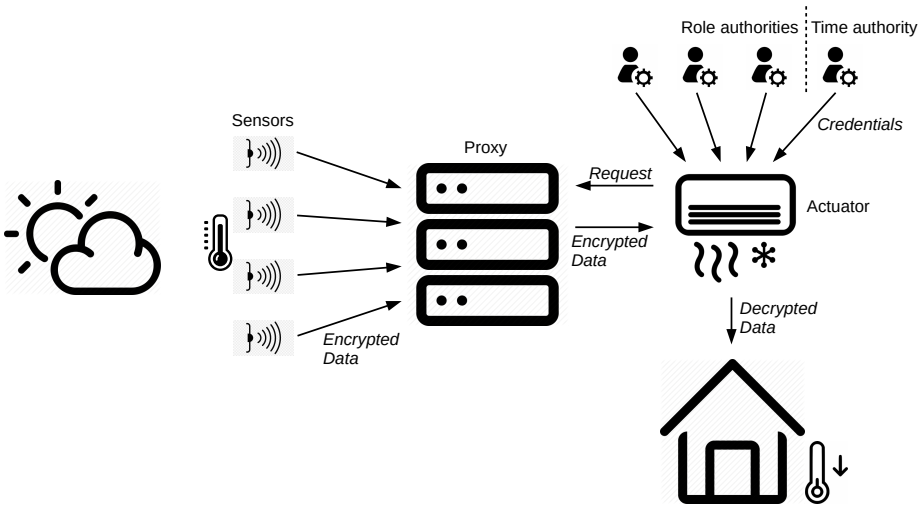

Figure 1: Our IoT scenario takes place in a smart home, where multiple temperature sensors are scattered and an actuator adjusts temperature in response to sensors' collected data.

is also an actuator (possibly indirectly, via a gateway for example) connected to these sensors. The actuator has received role and time credentials from multiple authorities. Since sensors have limited storage capacity, we suppose that they upload their encrypted data to a proxy (e.g. a cloud server). Within the rest of the paper, we assume that the proxy exists and is intimately linked to sensors, hence we omit to mention it explicitly. This proxy plays the role of an intermediary between the sensors and actuator. The actuator sends requests to the proxy for access to sensors' data every short time intervals, of the order of minutes. The proxy replies to the actuator's requests by forwarding the encrypted collected data. The actuator is able to recover the data in plain if and only if it has been granted with credentials satisfying sensors' assigned access policies. By having the plain data, the actuator adjusts the temperature accordingly.

1.1 Contributions

We propose an extension of the time-bound direct revocable ABE scheme presented in [27], with the following features:

1. User access control is made possible using user role attributes as well as time credentials, as in [27];

2. The participation of multiple role authorities, rather than one unique authority as in [27], alleviate the key escrow problem in the system;

3. A novel time-based access control using binary trees is used, instead of 31-ary trees as in [27], to better apply in IoT environments;

4. A direct approach based on a publicly available list for user revocation limits damages from compromised user secret keys, as in [27];

5. Asymmetric pairings are preferred, rather than symmetric pairings as in [27], to increase the system security and efficiency as proved in [20].

Those changes from the solution in [27] were not straightforward. First, shifting from symmetric pairings to asymmetric pairings incur less versatility in computing components. Symmetric pairings do not incur any order on inputs, while such an order appears with asymetric pairings. Therefore, it is essential to ensure that pairing calculations are still possible, e.g. for successful decryptions, while the components' order is correct. Second, allowing multiple authorities to manage the system brings its own challenges. We must ensure that authorities uniquely identify users based on specific attributes. Such an identification process guarantees that an unauthorised user does not get access to some data using an attribute from a role authority that incorrectly matches another attribute from another role authority. Last, changing the framework from 31-ary trees to 2-ary trees obliges to carefully define time periods. Indeed, using a 31-ary tree easily determines time periods, following our Gregorian calendar [15]. However, using a binary tree obliges us to carefully specify a process for setting time periods.

With such design, we manage to get a better trade-off between key update frequency and revocation list length compared to [27], to follow the constraints found in IoT, such as network dynamicity, heterogeneous origin and lack of governance [18, 19]. We adapt the security model in [27] to follow the multi-authority setting, and prove our scheme secure under the Decisional Bilinear Diffie-Hellman Exponent assumption (with asymmetric pairings). We also implement and evaluate our solution based on various parameters such as the maximum length of the revocation list and the number of role attributes per device. The observed results prove that our solution is fully deployable in IoT environments, where computing, communication and storage resources are highly limited [18, 19].

Since our scheme extends the one from [27], we choose to keep definitions and mathematical notations as close as possible to the ones found in [27], to facilitate the reading and comparison of the two schemes.

1.2 Related Work

Attribute-Based Encryption Identity-Based Encryption (IBE) [43, 12, 44] is a public-key cryptographic primitive that uses some unique information about the identity of a user (e.g. the email address) as the public key of that user. The corresponding secret key is generated by a trusted authority, based on the public key.

Attribute-Based Encryption (ABE) [9, 21, 45] is a variant of IBE (first called Fuzzy IBE [40]). Now, the secret key of a user and the ciphertext are dependent upon attributes (e.g. the country of living, the position within

the company, etc.). The decryption of a ciphertext is possible if and only if the attributes of the key match the attributes of the ciphertext.

There are two types of ABE schemes, that are closely related. Their difference comes from the access policy being linked either to the key or to the ciphertext. In the first one, called Key-Policy ABE (KP-ABE), the user secret key is linked to an access policy and the ciphertext is associated with an attribute set, such that the attributes in that set should satisfy the policy to get successful decryption. On the opposite, in a Ciphertext-Policy ABE (CP-ABE) scheme, the user secret key is associated with an attribute set while the ciphertext is linked to the access policy. Decryption works as above.

Extended Attribute-Based Encryption ABE has been subjected to many extensions, by including extra features while keeping security. Here, we review solutions with the features we are interested in, namely user revocation and multiple-authority setting. Yang and Jia [46] present a multi-authority CP-ABE scheme that embeds a revocation mechanism with forward and backward securities. In this scheme, each authority has its own attribute universe, and generates keys for users according to their attributes in that universe. However, one root authority is still required to generate the secret key material for each attribute authority, hence keeping the scheme prone to key escrow problem. Revoking a user is made possible by revoking one attribute granted to that user. Updating existing ciphertexts according to newly revoked users is delegated to a cloud server, thus alleviating the workload on the side of the user who generated them. Nevertheless, attribute revocation requires to update the secret keys of non-revoked users. Moreover, user revocation is decided by the authorities rather than the user who owns and has encrypted the data.

Liu et al. [28] propose to combine ABE and Time-based Proxy Re-Encryption to enable a fine-grained access control on encrypted data and scalable user revocation, while the data owner can remain offline. Revoking users is done by using time attributes. Users are given keys embedding role attributes as well as time attributes. The data owner encrypts the data according to an access policy; at this stage, time control is not enabled. Ciphertexts are uploaded to the cloud server (proxy), which updates the ciphertexts with the current time when users request data (like in a Proxy Re-Encryption scheme). If a user is not allowed to retrieve the data at the time of the request (by lack of adequate time attributes), then decryption fails. If a user has a key with the time attributes still available when requesting the data, then this user successfully decrypts. Unfortunately, the time control structure is cumbersome and not adaptable with time intervals but only with discrete timing. In addition, the data owner and cloud server must share a root secret key. While such key does not help the cloud server

to obtain information on the data, it permits to re-encrypt ciphertexts with the current time, implying strong trust assumptions on that cloud server. Also, the data owner is responsible for generating the secret keys of users, such that she decides time validity for them. Hence, the data owner should be fully trusted, while in general, data owners are also users requesting other data. Therefore, some misconducts can easily happen among colluding malicious users, making the system vulnerable. Practicality also suffers from such design: in a system with N users, all of them being owners of some data, each user needs $N-1$ keys generated by others. Li et al. [25] present a CP-ABE scheme with user revocation based on their attributes. However, such scheme requires the participation of a group manager to revoke attributes and thus users. Moreover, only one authority is responsible of allocating attributes to users, thus permitting key escrow to happen.

Liu et al. [27] combine CP-ABE with a direct revocation approach, that is, the most recent list of revoked users is always included in the ciphertext, and Hierarchical Identity-Based Encryption for time period control (i.e. a tree-based mechanism). In the rest of the paper, we refer to the scheme from [27] as the LYZL scheme. In order to avoid the revocation list growing too much as time goes by, each user obtains a key with an embedded validity time range. The users have then keys that expire on a date and one would only appear in the revocation list if she has been revoked before her key's expiration date (e.g. her key has been stolen before expiration). After key expiration, the name of the revoked user is discarded from the revocation list and a new user key is generated. Time periods are defined as a trade-off between a revocation list with reasonable length and a moderate frequency of key update. The key size depends on the validity time ranges assigned to users and on the maximum number of revoked users in the list, hence can dramatically grow. Moreover, there is a unique role authority in charge of generating user keys, promoting single point of failure and key escrow to happen. Symmetric pairings from cyclic groups of prime order are used, making the scheme less efficient and secure than using asymmetric pairings [20]. Our scheme keeps the positive features of the LYZL scheme, namely direct revocation with the list embedded into ciphertexts and time access control with a tree-based mechanism. However, we extend the solution by appending a robust multi-authority setting and by using asymmetric pairings and a suitable time framework, to improve the security and scalability of the system.

More recently, two CP-ABE schemes with direct revocation and time access control have been proposed in the literature [30, 49]. However, the drawbacks noticed in [27] are also found in [30, 49], namely single-authority setting and inefficient time-based tree hierarchy. Lastly, Li et al. [26] propose an ABE scheme managed by multiple authorities, to mitigate the key escrow problem. In addition to this feature, verifiability of data deletion has been added. However, this scheme does not consider the variety of users'

attributes (e.g. time and roles) and user revocation. Zhang et al. [50] propose a CP-ABE scheme that allows user revocation and prevents the key escrow problem. While the role authority is not responsible in generating the entire secret key of a user, there is still a single piont of failure in such design. Moreover, users' keys must be updated every time a user is revoked, by a specific authority. In case of frequent revocation events, this design may not be adequate.

Attribute-Based Encryption in IoT Yao et al. [47] present a new ABE scheme based on elliptic curve cryptography and without any pairing operation. Such features enable to obtain a lightweight and secure access control protocol in IoT networks. The authors also analyze the communication and computational costs induced by their solution and observe a significant gain in terms of practicality and efficiency over original schemes [9, 45]. While our scheme still requires pairing operations, we choose the asymmetric ones over elliptic curves (rather than symmetric ones), improving security and efficiency [20]. Moreover, contrary to us, no multi-authority and revocation mechanisms are implemented in Yao et al.'s scheme [47].

Oualha and Nguyen [35] propose an access control mechanism based on CP-ABE by considering the large number of devices in IoT networks and their constrained resources. Specifically, the authors apply pre-computations techniques [13] to Bethencourt et al.'s CP-ABE [9] to reduce the computational costs induced for data encryption, that is performed by IoT devices. However, the latter require more storage space since they must retain pre-computed tuples, that do not exist in the original scheme [9]. In addition, an extra trusted authority is required to generate these pre-computed values, and a secure channel is needed between this authority and the devices to transmit them. While encryption is made computationally easier for IoT devices, nothing is said about the rest of the access control protocol, namely secret key generation and storage, as well as decryption.

Meanwhile, Ambrosin et al. [6] study the feasibility of Bethencourt et al.'s CP-ABE [9] on widely used IoT-enabling devices. The authors focus on the evaluation of encryption and decryption steps, and test these cryptographic operations on four existing IoT platforms. Their results show that CP-ABE can be adopted in IoT environments without major flaw. The authors also present a successful use case application in smart healthcare using Bethencourt et al.'s CP-ABE [9]. While Oualha and Nguyen [35] show a technique to relieve computational workload on devices' side when implementing Bethencourt et al.'s CP-ABE [9], Ambrosin et al. [6] demonstrate that this CP-ABE scheme is fully adoptable in its original version. Results from Ambrosin et al. [6] suggest us that most of existing ABE schemes can be implemented in IoT systems.

Recently, few ABE-based IoT systems with access control based on time

and user revocation have been designed. A traceable and revocable time-based CP-ABE scheme has been proposed for smart cities [48]. Nevertheless, decryption needs to be outsourced to an external powerful entity in order to overcome devices' computational limitations. A time-bound access control for IoT in fog computing architecture using ABE technology is presented in [1]. However, only indirect user revocation is made possible. In the two aforementioned schemes, one unique authority is responsible of users' keys, thus the key escrow problem cannot be alleviated.

1.3 Road Map

In the following section, we define the building blocks and tools required for our solution. In Section 3, we present our CP-ABE scheme with multiple authority setting, direct revocation and time-based access control mechanism, along with its security. In Section 4, we implement and analyze our solution. Finally, in Section 5, we conclude the paper.

2 Preliminaries

2.1 Building Blocks

Multiple authorities We propose to enhance the LYZL scheme in [27] by involving multiple authorities. In [27], one authority, which is fully trusted, is in charge of setting up the system and generating the key material of users. Such configuration may be subject to key escrow and single point of failure. By enabling the generation of the user's public and secret parameters among several authorities, we reduce trust assumptions made on these authorities while enforcing the security of the scheme. More precisely, we only need to assume that one authority remains honest among all of them. For instance, if there are $N+1$ authorities in total, up to N authorities can be corrupted without breaking our scheme.

Revocation We follow the methodology proposed in [27], where revocation is done by making the secret key of a user unusable. The term "user" refers to a device in our system. The reasons can be diverse:

1. The user has left its IoT network, thus the key should no longer be usable. For instance, the owner of the temperature sensor has disconnected it from the smart home network.

2. The user has lost its key and been attributed a new one, hence the old key should no longer be usable. For example, a misuse of the IoT device by its owner has triggered some complications, such as key loss. When rebooting the device, a new key has been generated.

3. The user has one of its attributes changed and thus has received a new key with this new attribute, and the old key should no longer be usable. For

instance, one of the device's attributes has been modified from "everyone" to "adult" when some parental controls have been put in place.
Few approaches for revocation exist, such as key update for non-revoked users and cloud assistance. However, the former does not allow instant user revocation while the latter encounters practical issues when the number of users becomes huge.

A more interesting approach permits to revoke users by appending the user's identity in the revocation list. The list is public, instantly updated and included in each ciphertext in its latest version. Only the users not in the revocation list and with the attributes satisfying the access policy are able to decrypt the ciphertext. The main advantage is that key update is not necessary, avoiding extra communication and computational burdens. However, the number of users in that list grows with the time. If the number of users involved in the system is huge as in IoT networks, then this setting becomes a practical issue. An alternative is to create a non-revocation list that includes identities of non-revoked users. Hence, the length of this list will decrease over the time. We claim that the number of non-revoked devices is much larger than the number of revoked ones in an IoT system, making the non-revocation list difficult to handle.

The direct revocation mechanism presented in [27] is a trade-off between two techniques, namely appending the revocation list to ciphertexts and updating user keys based on time intervals. Users are given keys embedding their role attributes as well as their time validity ranges. The latter define a time period with an expiry date from which users are no longer authorized to access any data. Therefore, user keys are updated after the expiry date, such that the time interval between two updates should remain reasonable. Moreover, if a user is revoked before its key expires, then its identity is added into the revocation list and kept in it until the next key update. Then, a new key is generated according to role attributes and a new time validity range. We emphasize that key update is made possible but not mandatory (e.g. a user may be revoked definitely from the network). In addition, the generation of a key after its expiration may incur new attributes or discard used ones (e.g. a temperature sensor system has evolved and includes new functionalities, hence new attributes). We let the reader to refer to the exhaustive literature review on revocation in ABE in [27]. We keep this direct revocation mechanism proposed in [27] for our solution.

Role attributes In the LYZL scheme [27], an attribute universe is associated with the single authority, such that attributes are all different. In our solution, each role authority has its own attribute universe, such that the union of all the attribute universes forms the whole universe. We assume that attribute universes are all disjoint by defining attributes as follows: Let a role be "temperature" and two authorities refer to "Room A" and

"Room B" respectively. Hence, the two attributes are determined uniquely as "RoomA||temperature" and "RoomB||tempe- rature" respectively. Such appellation enables to obtain a whole universe with distinct attributes, as wished.

In the rest of the paper, we denote $\mathcal{U}_k$ the attribute universe associated with the role authority A_k. Let $\mathcal{U} = \cup_{A_k}\mathcal{U}_k$ be the disjoint union of all authorities' universes. Role attribution management is thus taken by multiple authorities in our system. They are responsible to define role attributes in their respective universes, and assign them to users when generating their keys. Role key updates do not require to be frequent, since roles such as "temperature" should remain forever for a temperature sensor. Key updates are rather required to refresh the revocation list once it reaches the maximum number of revoked users.

Time attributes The methodology proposed in [27] determines time intervals as days, months and years. The LYZL scheme supports both continuous and non-continuous time intervals; however, authors suggest that their method is only interesting in the case of continuous time intervals. The user encrypting the data defines a decryption time period such that only users with time credentials completely covering that time period can decrypt. For instance, if a user has time credential "15 January 2022" while the encryptor has set "January 2022" for decryption, then the user cannot decrypt since the credential does not completely cover the decryption time period. On the other side, if a user has time credential "January 2022" while the encryptor has set "from 01 to 15 January 2022" for decryption, then the former can successfully decrypt since it completely covers the decryption time period. Such properties are kept in mind when designing our time-based access control solution.

Authors in [27] suggest to use the Hierarchical Identity-Based Encryption scheme from [11] to create time validity control. Such tree-based approach allows improvement on the efficiency for continuous time intervals (claiming that user keys and ciphertexts are usually represented as time intervals). Then, a *set cover* approach is used to select the minimum number of nodes that represent all the valid time periods. Each node, except the root one, accounts for a time period such that leaves are days, leaves' parents are months and leaves' grand-parents are years. The root node is implicitly set as a starting time. Liu et al. [27] suggest that a 2-year interval between two key updates is reasonable, and thus the tree is constructed based on two consecutive years, for instance 2022 and 2023. Therefore, the starting time is "01 January 2022" and the tree represents time until "31 December 2023".

Let T be the depth of the tree and each node has z children. The time is thus represented as a z-ary string $\{1, 2, \cdots, z\}^{T-1}$ and a time period is

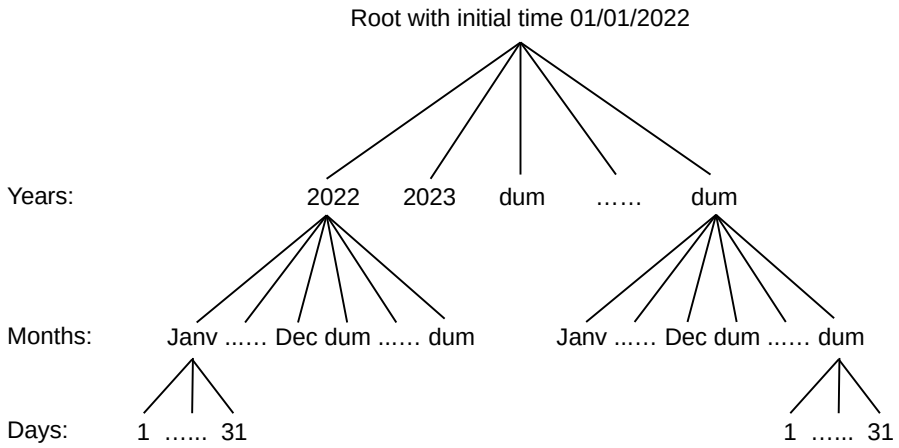

Figure 2: Time tree of the related LYZL scheme [27] for 2 years from "01 January 2022" until "31 December 2023" (included). "dum" denotes the dummy nodes. *Each* parent node has 31 child nodes regardless of the actual amount of children.

denoted with a z-ary element $(\tau_1, \tau_2, \cdots, \tau_\eta)$ for some $\eta < T$. No numerical value is given throughout the paper [27]; but we propose to make some assumptions from the reading. As mentioned above, the authors suggest that a 2-year interval between two key updates is reasonable and time periods based on year, month and day are enough for their purposes (but can be extended to minute and second). Moreover, in order to simplify the description of the tree structure, each node is supposed to have z children. From there, we infer that $T = 4$, and z is common to all non-leaf nodes and set to be equal to 31 (there are at most 31 days in a month). The latter assumption implies that the root has $z = 31$ children, and nodes representing years have also $z = 31$ nodes, even if 2-year intervals are examined and 12 months form a year. Such simplification approach causes the tree construction process to be more cumbersome with $31 - 2 = 29$ dummy nodes for years, $29 * (31 - 12) = 551$ dummy nodes for months and $29 * 19 * 31 = 17081$ dummy nodes for days. Figure 2 illustrates the above 2-year period example following the LYZL tree methodology.

Few ideas from the tree-based structure will be kept for our system. We also opt for a tree to represent the time framework with the root implicitly embedding a starting time and leaf nodes denoting days. We now explore the differences from the tree-based method in [27] and ours:

• In the LYZL scheme [27], the initialization algorithm solely generates

the parameters of the system and single authority. Since we involve multiple authorities, several algorithms are required to generate the common system parameters and the specific parameters of each authority. In particular, the authority responsible of time control creates the public and secret parameters required for the time tree.

• We choose a binary structure rather than a z-ary one (where $z = 31$ in [27]). Therefore, each node has two children. We hence avoid numerous dummy nodes from choosing the maximum value among number of years, number of months and number of days.

• We focus on shorter time periods according to our IoT-based time-sensitive data scenario. The number of leaf nodes (of the form 2^i for some integer i) defines the time interval between two key updates. Therefore, in order to keep the tree with a reasonable depth T, that number must be relatively small.

• Following our binary structure, a path from the root to a node is denoted as a string in $\{0,1\}^{T-1}$ where 0 denotes the left child and 1 denotes the right child of a given node.

• To construct a tree, one needs to choose a starting time (defining the root) and the number of days between two key updates. That number correspond to the number of leaf nodes. Then, from the bottom level, we build the tree up to the root.

Figure 3 illustrates a time tree following our methodology. The tree has depth $T = 5$, resulting into 16 leaf nodes, one for exactly one day. The root embeds the starting time "01 January 2022". Therefore, the time interval starts on "01 January 2022" and ends on "16 January 2022" (included). In our tree example, a user receives time key material for a time validity range of 7 days, starting 4 days after the starting time. This means that the user has been granted for a period from "04 January 2022" until "10 January 2022" (included). The user is given three key components as illustrated by blue circles in Figure 3: one for the leaf node representing day 4, one for the grand-parent of leaf nodes from day 5 until day 8 and for the parent of leaf nodes for days 9 and 10.

Following the 2-year period example given in [27], our tree would require 730 leaves, thus would have a depth equal to $T = 10$, making its generation and storage costs substantial on the system. A complete binary tree of depth $T = 10$ has $2^{10} - 1 = 1023$ nodes. However, following the tree construction in [27] as shown in Figure 2, we obtain a tree with $31 * 31 * 31 * 31 = 29791$ nodes, thus making the tree generation and storage costs noticeably worse than ours.

Time management is taken by a dedicated time authority. The latter is responsible to define time trees for the system and assign time validity ranges to users when generating their keys. Time key updates are frequent, of the order of several days, due to our IoT-based time-sensitive data scenario.

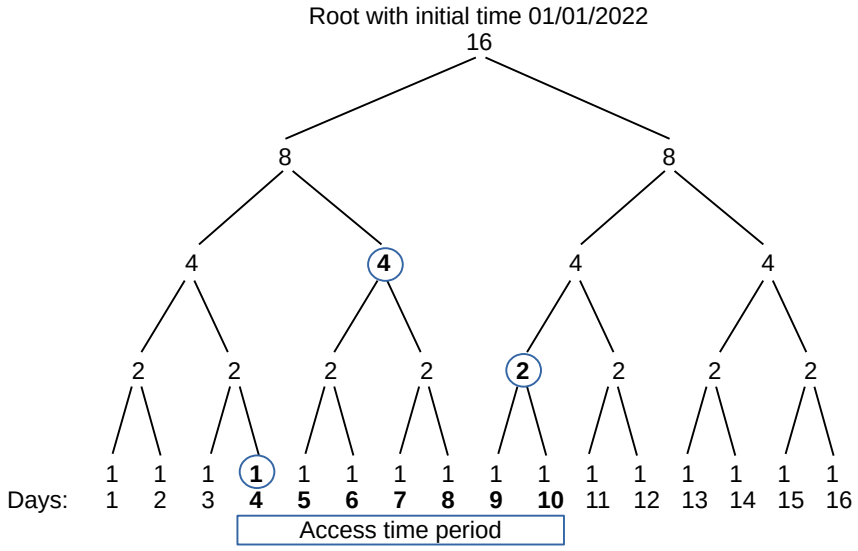

Figure 3: Time tree from "01 January 2022" until "16 January 2022" (included). Access time period is given for 7 days, from "04 January 2022" until "10 January 2022" (included): three keys corresponding to nodes with blue-line circles are then generated.

Asymmetric bilinear pairings Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be three multiplicative cyclic groups of prime order p. A pairing e is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ which satisfies three properties that are carefully specified in the next section. If the same group is used for the first two groups, meaning that $\mathbb{G}_1 = \mathbb{G}_2$, the pairing is called *symmetric* and is a mapping from two elements of one group to an element from a second group. Such setting is used in [27]. Otherwise, meaning that $\mathbb{G}_1 \neq \mathbb{G}_2$, the pairing is called *asymmetric*. In this case, either there is an efficiently computable homomorphism $\phi : \mathbb{G}_1 \rightarrow \mathbb{G}_2$ or there are no efficiently computable homomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$ [17]. It has been shown that designing a scheme in an asymmetric bilinear pairing setting rather than a symmetric one enables a better efficiency as well as an improved security level [20, 10]. Therefore, our solution extends the LYZL scheme [27], originally set with symmetric bilinear pairing, to permit an asymmetric pairing setting.

2.2 Miscellaneous.

In this section, we define the mathematical tools required to develop our Ciphertext-Policy Attribute-Based Encryption scheme for access control in the Internet of Things.

Notations For $n \in \mathbb{N}$, we define $[1, n] = \{1, 2, \cdots, n\}$. We use $x \in_R S$ to denote the process of uniformly sampling an element from set S and assigning it to variable x.

Vector Let $\vec{v} = (v_1, \cdots, v_R)$ be a vector in $\mathbb{Z}_p^R$ for an integer R. Let $g_1^{\vec{v}} = (g_1^{v_1}, \cdots, g_1^{v_R})^\top$ be a column vector in $\mathbb{G}_1$. Given $\vec{v}, \vec{w}$, let the product $\langle \vec{v}, \vec{w} \rangle$ be $\vec{v}^\top \vec{w} = \sum_{i=1}^{R} v_i w_i$ and $(g_1^{\vec{v}})^{\vec{w}}$ be $g_1^{\langle \vec{v}, \vec{w} \rangle}$.

Bilinear pairing Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be three multiplicative cyclic groups of prime order p. A pairing e is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ which satisfies the following properties:

• Bilinearity: Given $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$;

• Non-degeneracy: There exist $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ such that $e(g_1, g_2) \neq 1_{\mathbb{G}_T}$;

• Computability: There exists an efficient algorithm to compute $e(g_1, g_2)$ for all $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$.

Bilinear group Given as input a security parameter 1^λ, the algorithm Gen outputs the tuple $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ where $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are multiplicative cyclic groups of prime order p and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a pairing.

Access structure [8] Let $\mathcal{P} = \{P_1, P_2, \cdots, P_n\}$ be a set of parties. A collection $\mathbb{C} \subseteq 2^{\mathcal{P}}$ is said to be monotone if for all A, B, such that $A \in \mathbb{C}$ and $A \subseteq B$ then $B \in \mathbb{C}$. An access structure is a collection $\mathbb{C} \subseteq 2^{\mathcal{P}} \backslash \{\emptyset\}$. The sets in (resp. not in) $\mathbb{C}$ are said to be *authorized* (resp. *unauthorized*). For our solution, such a structure will be needed for user access based on their roles. Specifically, attributes will correspond to the roles of the user and authorized attribute sets will be contained in a monotone access structure $\mathbb{C}$.

Linear secret sharing scheme [8] A Secret Sharing Scheme (SSS) Π over a set of parties $\mathcal{P}$ is called Linear (and denoted as LSSS) if the following conditions hold:

• The shares of the parties form a vector over $\mathbb{Z}_p$;

• There are a $l \times \nu$ matrix M and a function ρ that maps the i-th row, for $i \in [1, l]$, to an associated party $\rho(i)$. Let $s \in \mathbb{Z}_p$ be a secret to be shared, and $\gamma_2, \cdots, \gamma_\nu$ be random exponents from $\mathbb{Z}_p$. Let $\vec{v} = (s, \gamma_2, \cdots, \gamma_\nu)$ be a column vector and $M\vec{v}$ be the vector of l shares of the secret s according to Π such that the share $(M\vec{v})_i$ belongs to party $\rho(i)$.

We now define the linear reconstruction property: Let Π be an LSSS for an access structure $\mathbb{C}$, $S \in \mathbb{C}$ be an authorized set and $I = \{i; \rho(i) \in S\} \subset [1, l]$. There exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that, if $\{\lambda_i\}$ are valid shares

of any secret s according to Π, then $\sum_{i \in I} \omega_i \lambda_i = s$. The constants ω_i can be found in time polynomial in the size of M. Moreover, for any unauthorized set $S \notin \mathbb{C}$, the secret s should be information theoretically hidden from the parties in S.

Let the vector $(1, 0, 0, \cdots, 0)$ be the *target* vector for LSSS [45]. Given an autorized set of rows I in the matrix M, the target vector is in the span of I. On the other side, given an unauthorized set of rows I, the target vector is not in the span of the rows of I. There also is a vector $\vec{w}$ such that $\vec{w}(1, 0, 0, \cdots, 0) = -1$ and $\vec{w}M_i = 0$ for all $i \in I$.

In our solution, we will define an user access policy based on an access structure $\mathbb{C}$ [29]. We do not specify any weight for a given access condition. Rather, we use `AND` and `OR` gates to emphasize whether an access condition is either required or optional. For instance, we consider the following access policy: A user can access the system if she satisfies both conditions: 1) she has attribute E; 2) she has at least 2 attributes among A, B, C and D. Such access policy can be expressed with the following boolean formula:

$$E \wedge ((A \wedge B) \vee (A \wedge C) \vee (A \wedge D) \vee (B \wedge C) \vee (B \wedge D) \vee (C \wedge D))$$

which can be reduced as follows:

$$E \wedge (((A \wedge B) \vee (C \wedge D)) \vee ((A \vee B) \wedge (C \vee D)))$$

The represention method with boolean formula intuitively responds to the relationships of `AND` and `OR`.

Access tree Let a tree represent an access structure. Each non-leaf node of the tree represents a threshold gate, described by its children and a threshold value. Considering the above example of access policy, we can build the corresponding access tree with `AND` and `OR` gates generated from the compressed boolean formula. The tree is depicted in Figure 4. A (t, n)-threshold gate tree is another kind of access trees. Here, the non-leaf nodes define a (t, n)-threshold where n is the number of leaves connected to this node and t is the number of leaves that must be covered to satisfy the access condition. Again, using the previous example, we build a threshold-gate access tree as $(E, (A, B, C, D, 2), 2)$.

An access tree is used in [27] to represent a time interval for access. In order to find the minimum number of nodes representing the time validity range, a set cover mechanism is used. Let the validity time range be from "30 November 2022" until "31 December 2023". Following this range, the selected nodes from Figure 2 are the node "30" with parent "November" and grand-parent "2022", the node "December" with parent "2022", and the node "2023". Let η_τ be the number of selected nodes and T be the depth of the tree such that $\eta_\tau < T$. Then, we define $\mathbb{T} = (\tau_1, \tau_2, \cdots, \tau_{\eta_\tau})$ as the set cover representing the time validity range. Following the above

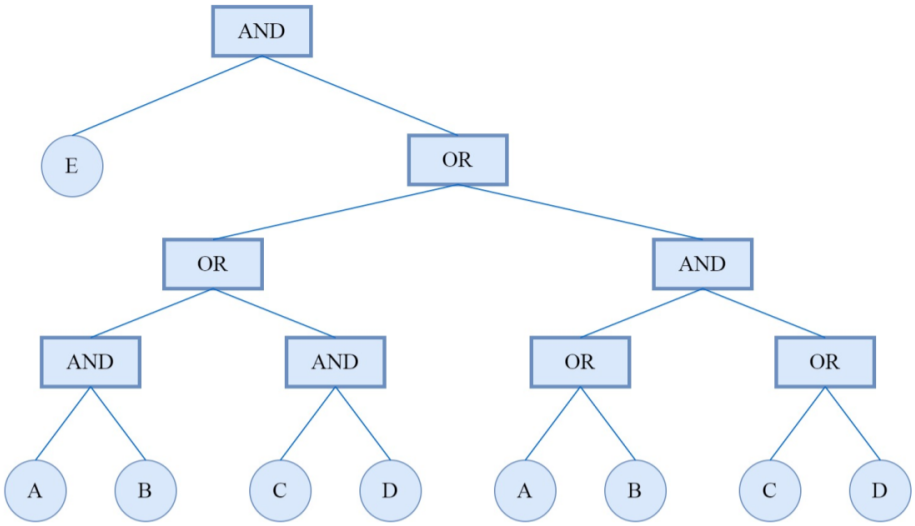

Figure 4: Access tree for $E \wedge (((A \wedge B) \vee (C \wedge D)) \vee ((A \vee B) \wedge (C \vee D)))$.

example, we get $\eta_\tau = 3$ such that $\tau_1 =$ "30 November 2022", $\tau_2 =$ "December 2022", and $\tau_3 =$ "2023". Note that in this example, we choose to not explicitly mention the presence of the dummy nodes to not overload the understanding. However, they should also be included in the set cover, thus $\eta_\tau >> 3$. Similarly to [27], we use an access tree to represent time periods for user access. However, rather than using z-ary strings $\{1, 2, \cdots, z\}^{T-1}$ with $z = 31$, we consider binary strings $\{0, 1\}^{T-1}$ such that a node represented as a bit 0 means going to the left child while as a bit 1 means going to the right child. Let the validity time range be from "04 January 2022" until "10 January 2022". The selected nodes are the ones circled in blue in Figure 3. Here, $\mathbb{T} = (\tau_1, \tau_2, \tau_3)$ where $\tau_1 = (0, 0, 1, 1)$, $\tau_2 = (0, 1)$ and $\tau_3 = (1, 0, 0)$.

Decisional q-BDHE assumption The security of our Ciphertext-Policy Attribute-Based Encryption scheme relies on the Decisional q-BDHE assumption. Given $\vec{P} = (g_1, g_1^s, g_1^a, \cdots, g_1^{a^q}, g_1^{a^{q+2}}, \cdots, g_1^{a^{2q}}, g_2, g_2^s, g_2^a, \cdots, g_2^{a^q}, g_2^{a^{q+2}}, \cdots, g_2^{a^{2q}}) \in \mathbb{G}_1^{2q+1} \times \mathbb{G}_2^{2q+1}$ and $Q \in \mathbb{G}_T$, where $s, a \in \mathbb{Z}_p$, $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, the Decisional q-Bilinear Diffie-Hellman Exponent (BDHE) problem is defined as to decide whether $Q = e(g_1, g_2)^{sa^{q+1}}$ or a random element in $\mathbb{G}_T$.

Indexing and implementing role attributes With a correct index assignment, we ensure that one index exactly corresponds to one attribute. All role attributes are unique since they are defined according to a specific role authority representing an IoT environment, and determine a role

within that environment. Then, we assign the indices for role attributes as follows: Let N be the number of role authorities and A_k be the role authority with universe $\mathcal{U}_k$ containing U_k attributes, for $k \in [1, N]$. Then, indices for attributes in the universe $\mathcal{U}_k$ associated with authority A_k are $(\sum_{j=1}^{k-1} U_j + 1), \cdots, (\sum_{j=1}^{k-1} U_j + U_k)$. To simplify the reading with indices, let $k||i = (\sum_{j=1}^{k-1} U_j + i)$ for $i \in [1, U_k]$. In addition, let $I = \{i; \rho(i) \in S\} \subseteq [1, l]$ be defined as above, and $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ be the set of constants such that if the set $\{\lambda_i\}$ contains valid shares of a value s according to the matrix M, then $\sum_{i \in I} \omega_i \lambda_i = s$. Let $\mathcal{A}$ be the set of role authorities whose attributes are in the access policy (i.e. the access structure). Let $\pi : k \rightarrow \pi(i)$ be defined as $\exists!(A_k \in \mathcal{A}, j \in [1, U_k])$ such that $\rho(i) = k||j$. Such surjective function exists since each attribute is defined uniquely in the whole universe $\mathcal{U} = \cup_{A_k} \mathcal{U}_k$. As mentioned above, an attribute in the whole universe $\mathcal{U}$ is uniquely controlled by one authority A_k. In order to explain the functionality of the function π and to make it implementable, let us assume that there exists a publicly computable function $F_\pi : \mathcal{U} \rightarrow \mathcal{A}_k$ that maps one attribute to a specific role authority [39]. From this mapping, let a second labeling of rows be defined in the access structure $((M, \rho), \rho')$ such that it maps rows to attributes via the function $\rho(\cdot) = F_\pi(\rho'(\cdot))$.

3 A new Multi-Authority Time-Based Revocable Ciphertext-Policy Attribute-Based Encryption

3.1 Overview

In Table 1, we provide the notation and definition of the elements involved in the construction of our ABE solution. An overview of our construction is given in Figure 5. Our solution contains seven algorithms, defining three phases:

Phase 1. An initialization phase sets up the system. Public parameters PP are generated and made available to authorities and users. Then, the role and time authorities, denoted as A_k and B respectively, generate their public and secret key material. That phase is run only once.

In more details, this phase comprises three algorithms. The first algorithm Setup is run as follows: $\mathsf{Setup}(1^\zeta, R) \rightarrow PP$, where 1^ζ is the security parameter, $R-1$ is the maximum number of revoked users at any time and PP are the public parameters. The second algorithm $\mathsf{RAKeyGen}$ is run as follows: $\mathsf{RAKeyGen}(PP, U_k) \rightarrow (PK_k, SK_k)$, where PP are the public parameters, U_k is the number of role attributes in the universe $\mathcal{U}_k$ associated with the role authority A_k, and PK_k and SK_k are the public and secret keys of the role authority A_k respectively. The third algorithm $\mathsf{TAKeyGen}$ is run as follows: $\mathsf{TAKeyGen}(PP, T) \rightarrow (PK, SK)$, where PP are the public parameters, T is the depth of the time binary tree associated with the time

Notation	Definition
1^{ζ}	Security parameter
$R-1$	Maximum number of revoked users
PP	Public parameters
A_k	Role authority
$\mathcal{U}_k$	Universe associated with A_k
U_k	Number of role attributes in $\mathcal{U}_k$
PK_k	Public key of A_k
SK_k	Secret key of A_k
B	Time authority
T	Depth of the time-based binary tree associated with B
$\{0,1\}^{T-1}$	Binary string representing the time
PK	Public key of B
SK	Secret key of B
ID	Identity of the user
$S_{ID,k}$	Role attribute set of user with ID and ass. with A_k
$RSK_{ID,k}$	Secret key of the user with ID and associated with A_k
T_{ID}	Time validity range of user with ID and ass. with B
TSK_{ID}	Secret key of user with ID and associated with B
$\mathcal{A}$	Set of role attributes in the access policy
m	Message to be encrypted
$\mathcal{R}$	Revocation list
(M,ρ)	LSSS access structure for the role access policy
T_{dec}	Decryption time period
$\mathcal{F}_{\mathcal{R}}(\cdot)$	Polynomial defining $\mathcal{R}$
CT	Ciphertext from m
S_{ID}	Disjoint union of all role attribute sets $S_{ID,k}$ for user with ID and ass. with A_k
τ_{dec}	Binary representation of decryption time period T_{dec}
$\mathbb{T}$	Set cover representing the time validity range T_{ID}

Table 1: Notation and definition of parameters.

authority B (such that the time is represented as a binary string $\{0,1\}^{T-1}$), and PK and SK are the public and secret keys of the time authority B respectively.

Phase 2. The authorities create the key material for users. There are N role authorities A_k (for $1 \leq k \leq N$), who are responsible of creating keys based on user roles (defined in role attribute sets) within their respective environments (i.e. universes $\mathcal{U}_k$). Role key updates for non-revoked users are run occasionally, say every two years. One time authority B generates user keys based on time validity ranges, denoted as T_{ID} where ID is the user identity. This authority frequently updates such key material based on new time validity ranges, e.g. each month.

In more details, this phase comprises two algorithms. The first algorithm $\mathsf{RUKeyGen}$ is run as follows: $\mathsf{RUKeyGen}(PP,\ (PK_k, SK_k), ID, S_{ID,k}) \rightarrow RSK_{ID,k}$, where PP are the public parameters PP, PK_k and SK_k are the public and secret keys of the role authority A_k, ID is the identity of the user and $S_{ID,k}$ is role attribute set that contains the roles of this user in A_k's environment. The output is the secret key $RSK_{ID,k}$ of the user associated with A_k. The second algorithm $\mathsf{TUKeyGen}$ is run as follows: $\mathsf{TUKeyGen}(PP, (PK, SK), ID, T_{ID}) \rightarrow TSK_{ID}$, where PP are the public parameters, PK and SK are the public and secret keys of the time authority B, ID is the identity of the user and T_{ID} is the time validity range of this user specified by B. The output is the secret key TSK_{ID} of the user associated with B.

Phase 3. An encryptor chooses an access policy, denoted as (M, ρ) where M is a $l \times \nu$ matrix and the function ρ associates rows of the matrix M to role attributes. She also selects a decryption time period T_{dec}. She then encrypts some data m according to that policy and time period, along with the current user revocation list $\mathcal{R}$, resulting in a ciphertext CT. A user which has been granted with role and time credentials satisfying the access policy and decryption time period can successfully decrypt the ciphertext CT and recover the data m.

In more details, this phase comprises two algorithms. The first algorithm $\mathsf{Encrypt}$ is run as follows: $\mathsf{Encrypt}(PP, \{PK_k\}_{A_k \in \mathcal{A}}, PK, m, \mathcal{R}, (M, \rho), T_{dec}) \rightarrow CT$, where PP are the public parameters, PK_k are the public keys of the role authorities A_k belonging to the role authority set $\mathcal{A}$ defined by the access policy, PK is the public key of the time authority B, m is the data to be encrypted, $\mathcal{R}$ is the current revocation list (with up to $R-1$ revoked users at any time), (M, ρ) is the access policy and T_{dec} is the allowed decryption time period. The output is the ciphertext CT of m. The second algorithm $\mathsf{Decrypt}$ is run as follows: $\mathsf{Decrypt}(PP, CT, \mathcal{R}, \{RSK_{ID,k}\}_{A_k \in \mathcal{A}}, TSK_{ID}) \rightarrow m$, where PP are the public parameters, CT is the ciphertext, $\mathcal{R}$ is the current revocation list, $RSK_{ID,k}$ is the role secret key of the user with identity ID and associated with $A_k \in \mathcal{A}$, and TSK_{ID} is the time secret key of this user associated with B. If the user has the required credentials, in terms

of role attributes and time validity range, and her identity ID is not in the revocation list $\mathcal{R}$, then this user should be able to get the output as the message m. If the user fails to get the required credentials and/or has been revoked, then she gets a null sign $\perp$.

Phase	Algorithm	Role Authority	Time Authority	User (sensor)	User (actuator)
1	Setup	PP	PP	PP	PP
	RAKeyGen	PK_k, SK_k			
	TAKeyGen		PK, SK		
2	RUKeyGen		$\xrightarrow{RSK_{ID,k}}$		
	TUKeyGen		$\xrightarrow{TSK_{ID}}$		
3	Encrypt			$\xrightarrow{CT}$	
	Decrypt				m

Figure 5: Overview of our proposed solution. The column 'Algorithm' indicates which algorithm is run to obtain the output mentioned on the same row, under the corresponding recipient. Arrows link, from left to right, a source entity which runs the algorithm, and a destination entity which receives the output.

3.2 Construction

The Multi-Authority Time-Based Revocable Ciphertext-Po- licy Attribute-Based Encryption construction is as follows:

Phase 1.

• $\mathsf{Setup}(1^\zeta, R)$. The algorithm Setup generates the public parameters made available to all participating entities, namely authorities and users.

Let $R-1$ be the maximum number of revoked users. On inputs the security parameter 1^ζ and R, the algorithm Setup outputs the public parameters PP. First, run the algorithm Gen and obtain two bilinear groups $\mathbb{G}_1, \mathbb{G}_2$ of prime order p with generators g_1 and g_2 respectively, along with a third group $\mathbb{G}_T$ of prime order p and a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Pick at random $\delta, \alpha_1, \cdots, \alpha_R \in_R \mathbb{Z}_p$ Set $\vec{\alpha} = (\alpha_1, \cdots, \alpha_R)^\top$ and $\vec{F} = g_1^{\vec{\alpha}} = (g_1^{\alpha_1}, \cdots, g_1^{\alpha_R})^\top = (f_1, \cdots, f_R)^\top$. The public parameters are $PP = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_1^\delta, g_2, \vec{F})$.

• $\mathsf{RAKeyGen}(PP, U_k)$. The algorithm $\mathsf{RAKeyGen}$ generates the public and secret keys of the role authority.

Let U_k be the number of role attributes in the universe $\mathcal{U}_k$ associated with role authority A_k. On inputs the public parameters PP and U_k, the algorithm $\mathsf{RAKeyGen}$ outputs the public key PK_k and the secret key SK_k of the role authority A_k. Pick at random $\kappa_k \in_R \mathbb{Z}_p$ and $h_{k||1}, \cdots, h_{k||U_k} \in_R \mathbb{G}_1$

(these elements $h_{k||i}$ will be used for role access control with relation to the authority A_k). The public key is $PK_k = (e(g_1, g_2)^{\kappa_k}, h_{k||1}, \cdots, h_{k||U_k})$ and the secret key is $SK_k = \kappa_k$.

• $\mathsf{TAKeyGen}(PP, T)$. The algorithm $\mathsf{TAKeyGen}$ generates the public and secret keys of the time authority.

Let T be the depth of the time binary tree associated with time authority B. The time is represented as a binary string $\{0,1\}^{T-1}$. On inputs the public parameters PP and T, the algorithm $\mathsf{TAKeyGen}$ outputs the public key PK and the secret key SK of the time authority B. Pick at random $\sigma \in_R \mathbb{Z}_p$ and $V_0, V_1, \cdots, V_T \in_R \mathbb{G}_1$ (the elements V_j will be used for time access control w.r.t. the time authority B). The public key is $PK = (e(g_1, g_2)^{\sigma}, V_0, V_1, \cdots, V_T)$ and the secret key is $SK = \sigma$.

Phase 2.

• $\mathsf{RUKeyGen}(PP, (PK_k, SK_k), ID, S_{ID,k})$. The algorithm $\mathsf{RUKeyGen}$ generates the secret key of the user w.r.t. her roles.

Let $S_{ID,k}$ be the role attribute set of a user with identity ID and associated with role authority A_k. Let $k||x \in S_{ID,k}$ denote the attribute uniquely defined in the whole universe $\mathcal{U} = \cup_{A_k} \mathcal{U}_k$ by determining the associated authority A_k and the role x within $\mathcal{U}_k$. On inputs the public parameters PP, the public and secret keys PK_k and SK_k of the role authority A_k, ID and $S_{ID,k}$, the algorithm $\mathsf{RUKeyGen}$ outputs the secret key $RSK_{ID,k}$ of the user with identity ID, role attribute set $S_{ID,k}$ and associated with authority A_k. First, pick at random $u_k, t_k \in_R \mathbb{Z}_p$. Then, compute the following:

$$\begin{aligned}
D_{k,0} &= g_2^{t_k} \\
D'_{k,0} &= g_2^{u_k} \\
D_{k,1} &= g_1^{\kappa_k} g_1^{\delta t_k} f_1^{u_k} = g_1^{\kappa_k} g_1^{\delta t_k} g_1^{\alpha_1 u_k} \\
K_{k,x} &= h_{k||x}^{t_k} \text{ for } k||x \in S_{ID,k} \\
F_{k,i} &= (f_1^{-ID^{i-1}} f_i)^{u_k} \text{ for } i \in [2, R]
\end{aligned}$$

The secret key is $RSK_{ID,k} = (D_{k,0}, D'_{k,0}, D_{k,1}, \{K_{k,x}\}_{k||x \in S_{ID,k}}, \{F_{k,i}\}_{i \in [2,R]})$ and includes a description of $S_{ID,k}$.

• $\mathsf{TUKeyGen}(PP, (PK, SK), ID, T_{ID})$. The algorithm $\mathsf{TUKeyGen}$ generates the secret key of the user w.r.t. her access time period.

Let T_{ID} be the time validity range of the user with identity ID and associated with time authority B. On inputs the public parameters PP, the public and secret keys PK and SK of the time authority B, ID and T_{ID}, the algorithm $\mathsf{TUKeyGen}$ outputs the secret key TSK_{ID} of the user with identity ID, time validity range T_{ID} and associated with authority B. Let $\mathbb{T}$ be the set cover representing T_{ID} which consists of time elements $\tau = (\tau_1, \cdots, \tau_{\eta_\tau}) \in \{0,1\}^{\eta_\tau}$ where $\eta_\tau < T$ for any $\tau \in \mathbb{T}$. First, pick at

random $\beta, v_\tau \in_R \mathbb{Z}_p$ for $\tau \in \mathbb{T}$. Then, compute the following:

$$\begin{aligned}
D_{0,\tau} &= g_2^{v_\tau} \text{ for } \tau \in \mathbb{T} \\
D_{1,\tau} &= g_1^{\sigma} f_1^{\beta} (V_0 \prod_{j=1}^{\eta_\tau} V_j^{\tau_j})^{v_\tau} \text{ for } \tau \in \mathbb{T} \\
D_2 &= g_2^{\beta} \\
L_{j,\tau} &= V_j^{v_\tau} \text{ for } j \in [\eta_\tau + 1, T] \text{ and } \tau \in \mathbb{T} \\
E_i &= (f_1^{-ID^{i-1}} f_i)^{\beta} \text{ for } i \in [2, R]
\end{aligned}$$

The secret key is $TSK_{ID} = (\{D_{0,\tau}, D_{1,\tau}\}_{\tau \in \mathbb{T}}, D_2, \{L_{j,\tau}\}_{j \in [\eta_\tau + 1, T], \tau \in \mathbb{T}}, \{E_i\}_{i \in [2,R]})$ and includes a description of T_{ID}.

Phase 3.

• $\mathsf{Encrypt}(PP, \{PK_k\}_{A_k \in \mathcal{A}}, PK, m, \mathcal{R}, (M, \rho), T_{dec})$. The algorithm Encrypt generates a ciphertext of the message m.

Let $\mathcal{A}$ be the set of role authorities whose role attributes are in the access policy. Let m be the message to be encrypted. Let $\mathcal{R} = (ID_1, \cdots, ID_r)$ be the revocation list containing $r < R$ revoked users. Let (M, ρ) be an LSSS access structure, defining the role access policy, where M is a $l \times \nu$ matrix and the function ρ associates rows of the matrix M to role attributes. Let T_{dec} be the decryption time period of the ciphertext. On inputs the public parameters PP, the public keys PK_k of the role authorities $A_k \in \mathcal{A}$, the public key PK of the time authority B, m, $\mathcal{R}$, (M, ρ) and T_{dec}, the algorithm Encrypt outputs a ciphertext CT. Let $\tau_{dec} = (\tau_1, \cdots, \tau_{\eta_{dec}}) \in \{0,1\}^{\eta_{dec}}$ be the binary representation of T_{dec}, where $\eta_{dec} < T$. First, choose a secret s from $\mathbb{Z}_p$ and pick at random $\gamma_2, \cdots, \gamma_\nu \in_R \mathbb{Z}_p$. Set the vector $\vec{v} = (s, \gamma_2, \cdots, \gamma_\nu)$. Then, for $i \in [1, l]$, compute $\lambda_i = \langle \vec{v}, M_i \rangle$, where M_i is the i-th row of M. Let $\mathcal{F}_{\mathcal{R}}(Z) = (Z - ID_1) \cdot (Z - ID_2) \cdots (Z - ID_r) = y_1 + y_2 Z + \cdots + y_r Z^{r-1} + y_{r+1} Z^r$ be a polynomial defining the revocation list. If $r + 1 < R$, then set the coefficients $y_{r+2}, \cdots, y_R$ equal to 0. Then, compute the following:

$$\begin{aligned}
C_0 &= m \cdot e(g_1, g_2)^{\sigma s} \cdot \prod_{A_k \in \mathcal{A}} e(g_1, g_2)^{\kappa_k s} \\
C_0' &= g_2^s \\
C_0'' &= (f_1^{y_1} \cdots f_R^{y_R})^s \\
C_0''' &= (V_0 \prod_{j=1}^{\eta_{dec}} V_j^{\tau_j})^s \\
C_i &= g_1^{\delta \lambda_i} h_{\rho(i)}^{-s} \text{ for } i \in [1, l]
\end{aligned}$$

The ciphertext is $CT = (C_0, C_0', C_0'', C_0''', \{C_i\}_{i \in [1,l]}, (M, \rho))$ and includes descriptions of T_{dec} and $\mathcal{A}$.

• $\mathsf{Decrypt}(PP, CT, \mathcal{R}, \{RSK_{ID,k}\}_{A_k \in \mathcal{A}}, TSK_{ID})$. The algorithm $\mathsf{Decrypt}$ attempts to recover the message m from the ciphertext using appropriate secret parameters. On inputs the public parameters PP, the ciphertext CT, the revocation list $\mathcal{R}$, the role secret keys $RSK_{ID,k}$ of user with identity ID and associated with $A_k \in \mathcal{A}$ and the time secret key TSK_{ID} of user with identity ID and associated with B, the algorithm $\mathsf{Decrypt}$ outputs either the message m or a null sign $\perp$.

Let $\vec{X} = (1, ID, \cdots, ID^{R-1})$ for the identity ID and $\vec{Y} = (y_1, \cdots, y_R)$, where the exponents y_i have been defined during the encryption phase. Hence, $\langle \vec{X}, \vec{Y} \rangle = y_1 + y_2 ID + \cdots + y_r ID^{r-1} + y_{r+1} ID^r = \mathcal{F}_{\mathcal{R}}(ID)$. If $r + 1 < R$, then the coefficients $y_{r+2}, \cdots, y_R$ are equal to 0. Let $S_{ID} = \cup_{A_k \in \mathcal{A}} S_{ID,k}$ be the disjoint union of all the role attribute sets $S_{ID,k}$ of the user with identity ID and associated with $A_k \in \mathcal{A}$. Let τ_{dec} be the binary representation for the decryption time period T_{dec} and $\mathbb{T}$ be the set cover representing the time validity range T_{ID}. Let us define the following conditions:

• *Insufficient roles attributes:* S_{ID} does not satisfy the access structure (M, ρ);

• *Revoked user:* $ID \in \mathcal{R}$, that is $\langle \vec{X}, \vec{Y} \rangle = \mathcal{F}_{\mathcal{R}}(ID) = 0$;

• *Invalid access time period:* T_{dec} is not completely covered in T_{ID}, that is τ_{dec} and all its prefixes are not in $\mathbb{T}$.

If any of the above conditions occurs, then output $\perp$ and abort. Otherwise, since $\langle \vec{X}, \vec{Y} \rangle \neq 0$, compute the following:

$$
\begin{aligned}
F_k &= \prod_{i=2}^{R} F_{k,i}^{y_i} = (f_1^{-\langle \vec{X}, \vec{Y} \rangle} \prod_{i=1}^{R} f_i^{y_i})^{u_k} \\
\xi_{k,1} &= \left(\frac{e(F_k, C_0')}{e(C_0'', D_{k,0}')} \right)^{\frac{-1}{\langle \vec{X}, \vec{Y} \rangle}} = e(g_1, g_2)^{\alpha_1 s u_k} \\
E &= \prod_{i=2}^{R} E_i^{y_i} = (f_1^{-\langle \vec{X}, \vec{Y} \rangle} \prod_{i=1}^{R} f_i^{y_i})^{\beta} \\
\xi_1' &= \left(\frac{e(E, C_0')}{e(C_0'', D_2)} \right)^{\frac{-1}{\langle \vec{X}, \vec{Y} \rangle}} = e(g_1, g_2)^{\alpha_1 s \beta}
\end{aligned}
$$

Let $I \subseteq [1, l]$ be defined as $\{i; \rho(i) \in S_{ID}\}$ and $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ be the set of constants such that if the set $\{\lambda_i\}$ contains valid shares of a value s according to the matrix M, then $\sum_{i \in I} \omega_i \lambda_i = s$. In addition, there is a surjective function from I to $\mathcal{A}$ determined as follows: Let $\pi : k \to \pi(i)$ be defined as $\exists!(A_k \in \mathcal{A}, j \in [1, U_k])$ such that $\rho(i) = k||j$. Such function exists since each attribute is defined uniquely in the whole universe $\mathcal{U} = \cup_{A_k} \mathcal{U}_k$. Then, compute:

$$
\xi_2 = \prod_{i \in I} \left(e(C_i, D_{\pi(i),0}) \cdot e(K_{\rho(i)}, C_0') \right)^{\omega_i} = \prod_{A_k \in \mathcal{A}} e(g_1, g_2)^{\delta s t_k}
$$

If $\tau_{dec} = (\tau_1, \cdots, \tau_{\eta_{dec}}) \in \mathbb{T}$, then $D_{1,\tau_{dec}}$ should be one component of the secret key TSK_{ID}. Otherwise, let $\tau'_{dec} = (\tau_1, \cdots, \tau_{\eta'_{dec}})$ denote the prefix such that $\eta'_{dec} < \eta_{dec}$ and $\tau'_{dec} \in \mathbb{T}$. Then, derive a key component $D_{1,\tau_{dec}}$ from TSK_{ID} with respect to τ'_{dec} by calculating $D_{1,\tau_{dec}} = D_{1,\tau'_{dec}} \prod_{j=\eta'_{dec}+1}^{\eta_{dec}} L_{j,\tau'_{dec}}^{\tau_j}$ and set $\tau_{dec} = \tau'_{dec}$. Finally, recover the message m as follows:

$$m = C_0 \cdot \xi_2 \cdot \frac{e(D_{0,\tau_{dec}}, C_0''') \cdot \xi'_1}{e(D_{1,\tau_{dec}}, C'_0)} \cdot \prod_{A_k \in \mathcal{A}} \frac{\xi_{k,1}}{e(D_{k,1}, C'_0)}$$

Correctness We first calculate the element F_k that is needed for calculating $\xi_{k,1}$. Implicitly, the user must not have been revoked in order to get a correct result for F_k and proceed for successful decryption.

$$\begin{aligned} F_k &= \prod_{i=2}^{R} F_{k,i}^{y_i} \\ &= \prod_{i=2}^{R} (f_1^{-ID^{i-1}} f_i)^{y_i u_k} = (f_1^{-(IDy_2 + ID^2 y_3 + \cdots + ID^{R-1} y_R)} \\ &\quad \cdot g_1^{\sum_{i=2}^{R} \alpha_i y_i})^{u_k} \\ &= (f_1^{-\langle \vec{X}, \vec{Y} \rangle + y_1} \prod_{i=2}^{R} f_i^{y_i})^{u_k} = (f_1^{-\langle \vec{X}, \vec{Y} \rangle} \prod_{i=1}^{R} f_i^{y_i})^{u_k} \end{aligned}$$

Using the result from calculating F_k, we can now calculate the element $\xi_{k,1}$ that participates in checking the role credentials of the user, associated with the role authority A_k, against the access policy. Note that if the user has been revoked, then we would not be able to calculate such element (we need $\langle \vec{X}, \vec{Y} \rangle \neq 0$).

$$\begin{aligned} \xi_{k,1} &= \left(\frac{e(F_k, C'_0)}{e(C''_0, D'_{k,0})} \right)^{\frac{-1}{\langle \vec{X}, \vec{Y} \rangle}} \\ &= \left(\frac{e((f_1^{-\langle \vec{X}, \vec{Y} \rangle} \prod_{i=1}^{R} f_i^{y_i})^{u_k}, g_2^s)}{e((f_1^{y_1} \cdots f_R^{y_R})^s, g_2^{u_k})} \right)^{\frac{-1}{\langle \vec{X}, \vec{Y} \rangle}} = e(g_1, g_2)^{\alpha_1 s u_k} \end{aligned}$$

We then calculate the element E that takes again into account the revocation status of the user.

$$\begin{aligned} E &= \prod_{i=2}^{R} E_i^{y_i} = \prod_{i=2}^{R} (f_1^{-ID^{i-1}} f_i)^{y_i \beta} \\ &= (f_1^{-(IDy_2 + ID^2 y_3 + \cdots + ID^{R-1} y_R)} \cdot g_1^{\sum_{i=2}^{R} \alpha_i y_i})^{\beta} \\ &= (f_1^{-\langle \vec{X}, \vec{Y} \rangle + y_1} \prod_{i=2}^{R} f_i^{y_i})^{\beta} = (f_1^{-\langle \vec{X}, \vec{Y} \rangle} \prod_{i=1}^{R} f_i^{y_i})^{\beta} \end{aligned}$$

Using the result from calculating E, we can now calculate the element ξ_1' that participates in checking the time credentials of the user, associated with the time authority B, against the access time period. Note that if the user has been revoked, then we would not be able to calculate such element (we need $\langle \vec{X}, \vec{Y} \rangle \neq 0$).

$$\begin{aligned}
\xi_1' &= \left(\frac{e(E, C_0')}{e(C_0'', D_2)} \right)^{\frac{-1}{\langle \vec{X}, \vec{Y} \rangle}} \\
&= \left(\frac{e((f_1^{-\langle \vec{X}, \vec{Y} \rangle} \prod_{i=1}^{R} f_i^{y_i})^{\beta}, g_2^s)}{e((f_1^{y_1} \cdots f_R^{y_R})^s, g_2^{\beta})} \right)^{\frac{-1}{\langle \vec{X}, \vec{Y} \rangle}} = e(g_1, g_2)^{\alpha_1 s \beta}
\end{aligned}$$

We then calculate ξ_2 that involves the linear reconstruction property of the LSSS access structure, meaning that user role credentials are prepared for verification.

$$\begin{aligned}
\xi_2 &= \prod_{i \in I} \left(e(C_i, D_{\pi(i),0}) \cdot e(K_{\rho(i)}, C_0') \right)^{\omega_i} \\
&= \prod_{i \in I} \left(e(g_1^{\delta \lambda_i} h_{\rho(i)}^{-s}, g_2^{t_{\pi(i)}}) \cdot e(h_{\rho(i)}^{t_{\pi(i)}}, g_2^s) \right)^{\omega_i} \\
&= \prod_{i \in I} \left(e(g_1^{\delta \lambda_i}, g_2^{t_{\pi(i)}}) \cdot e(h_{\rho(i)}^{-s}, g_2^{t_{\pi(i)}}) \cdot e(h_{\rho(i)}^{t_{\pi(i)}}, g_2^s) \right)^{\omega_i} \\
&= \prod_{i \in I} e(g_1, g_2)^{t_{\pi(i)} \delta (\lambda_i \omega_i)} = \prod_{A_k \in \mathcal{A}} e(g_1, g_2)^{s t_k \delta}
\end{aligned}$$

Finally, we recover the message m as follows:

$$\begin{aligned}
& C_0 \cdot \xi_2 \cdot \frac{e(D_{0,\tau_{dec}}, C_0''') \cdot \xi_1'}{e(D_{1,\tau_{dec}}, C_0')} \cdot \prod_{A_k \in \mathcal{A}} \frac{\xi_{k,1}}{e(D_{k,1}, C_{k,0}')} \\
= & \left(m \cdot e(g_1, g_2)^{\sigma s} \cdot \prod_{A_k \in \mathcal{A}} e(g_1, g_2)^{\kappa_k s} \right) \cdot \prod_{A_k \in \mathcal{A}} e(g_1, g_2)^{s t_k \delta} \\
& \cdot \frac{e((V_0 \prod_{j=1}^{\eta_{dec}} V_j^{\tau_j})^s, g_2^{v_{\tau_{dec}}}) \cdot e(g_1, g_2)^{\alpha_1 s \beta}}{e(g_1^{\sigma} g_1^{\alpha_1 \beta} (V_0 \prod_{j=1}^{\eta_{dec}} V_j^{\tau_j})^{v_{\tau_{dec}}}, g_2^s)} \\
& \cdot \prod_{A_k \in \mathcal{A}} \frac{e(g_1, g_2)^{\alpha_1 s u_k}}{e(g_1^{\kappa_k} g_1^{\delta t_k} g_1^{\alpha_1 u_k}, g_2^s)}
\end{aligned}$$

$$\begin{aligned}
&= m \cdot e(g_1, g_2)^{\sigma s} \cdot \frac{1}{e(g_1, g_2)^{\sigma s}} \\
&\quad \cdot \left(\prod_{A_k \in \mathcal{A}} e(g_1, g_2)^{\kappa_k s} \cdot \frac{1}{e(g_1, g_2)^{\kappa_k s}} \right) \\
&= m
\end{aligned}$$

where the role attributes of the user cancel out with the ones embedded in the access policy (based on the linear reconstruction property of the LSSS access structure), while the time validity range of the user fits in the decryption time period (based on the set cover mechanism and binary tree representation).

User collusion The user may have multiple role secret keys $RSK_{ID,k}$, issued by different role authorities A_k, along with a time secret key TSK_{ID}, issued by B. Each key $RSK_{ID,k}$ embeds her identity ID in the component $F_{k,i}$ for authority $A_k \in \mathcal{A}$, and the key TSK_{ID} contains ID in the component E_i, for index $i \in [2, R]$. These elements are required to check whether the user belongs to the revoked list $\mathcal{R}$. If so, then some decrypting elements do not cancel out, and then decryption fails.

Moreover, the elements $F_{k,i}$ and E_i avoid user collusion. Let us suppose that a user with identity ID has the appropriate role attributes to fulfill the decryption requirements, but does not have the suitable time validity range T_{ID}. Hence, this user should not be able to successfully decrypt the ciphertext. We now assume that this user with identity ID colludes with the user with identity ID', that has the suitable time validity range $T_{ID'}$. Thus, the user with identity ID attempts to decrypt the ciphertext using her own role keys $RSK_{ID,k}$ and the time key $TSK_{ID'}$ from the user with identity ID'. We observe that the identity ID is required to generate the vector $\vec{X}$, while the element E contains the identity ID', thus the element ξ_1' is not correctly calculated (some factors are not correctly deleted), and the user with identity ID fails decrypting.

Revocation list In the current version, the revocation list $\mathcal{R}$ is given in plain in the ciphertext CT, following [27]. However, we could improve the efficiency and security by providing the hash value of $\mathcal{R}$ in CT. The revocation list $\mathcal{R}$ is then stored in plain on a publicly accessible server from which the decryptor retrieves the list $\mathcal{R}$ and verifies its integrity with the hash value given in CT.

3.3 Security

In order to prove our scheme secure, we suppose that either there is at least one honest role authority whose some attributes are included in the access policy or the time authority is honest. Indeed, if all the (role and time) authorities are malicious and collude, then the key generation may be altered to the advantage of these authorities. Our Multi-Authority Time-Based Revocable Ciphertext-Policy Attribute-Based Encryption scheme is selectively secure as long as the Decisional q-BDHE assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$ [27].

3.3.1 Security Model

Similarly to Identity-Based Encryption (IBE) schemes, the adversary $\mathcal{E}$ is allowed to query users' secret keys such that they cannot be used to decrypt the challenged ciphertext CT^* [45]. Moreover, from the ABE framework, access structures identify the ciphertexts and attributes identify users' secret keys. Hence, the adversary $\mathcal{E}$ commits to be challenged on an encryption to an access structure (M^*, ρ^*) along with a revocation list $\mathcal{R}^*$ and a decryption time period T^*_{dec}. The adversary $\mathcal{E}$ is allowed to request any user's secret key with attribute set $S_{ID,k}$ as long as the authority A_k remains honest and $S_{ID,k}$ does not satisfy (M^*, ρ^*), i.e. the user with identity ID has not enough attributes from this authority A_k to decrypt [14].

As in [27], we consider a *selective* security model for our solution where the adversary commits to the challenged access structure (M^*, ρ^*) before setup. The following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{E}$ describes the selective security model. In that game, $\mathcal{E}$ first submits a challenged access structure (M^*, ρ^*), a challenged revocation list $\mathcal{R}^*$, a challenged set $\mathcal{A}^*$ of role authorities whose attributes are in the challenged access policy (i.e. the access structure) and a challenged decryption time period T^*_{dec} to $\mathcal{C}$ and then receives the public parameters and authorities' public keys. The adversary is permitted to query users' secret keys that cannot be used to decrypt the challenged ciphertext CT^*. In addition, following [14, 31], $\mathcal{E}$ selects an honest authority $A_{k^*} \in \mathcal{A}^*$ for some index k^*. Therefore, the adversary is allowed to request secret keys for a given user with identity ID and attribute set S_{ID} as long as there remains one honest authority $A_{k^*} \in \mathcal{A}^*$ such that the user has insufficient attributes from this authority to decrypt. Note that we focus on the case where the honest authority is a role authority; similarly, one can design the proof with the honest authority being the time authority.

Initialization. The adversary submits the challenged access structure (M^*, ρ^*), challenged revocation list $\mathcal{R}^*$ and challenged decryption time period T^*_{dec} to the challenger. It must also provide the challenged set $\mathcal{A}^*$ of role authorities whose attributes are in the challenged access policy and at least one honest authority $A_{k^*} \in \mathcal{A}^*$.

Setup. $\mathcal{C}$ runs the Setup, RAKeyGen and TAKeyGen algorithms and gives to $\mathcal{E}$ the public parameters PP, the public keys PK_k for all A_k and the public key PK for B.

Query Phase 1. The adversary can make secret key queries corresponding to the user with identity ID and secret keys $RSK_{ID,k}$ and TSK_{ID} such that:

• The secret keys $RSK_{ID,k}$ of the user with identity ID and associated with A_k result from the role attribute sets $S_{ID,k}$.

• The secret key TSK_{ID} results from the range T_{ID}.

Then, at least one of the following conditions must hold:

• Let $S_{ID} = \cup_{A_k \in \mathcal{A}^*} S_{ID,k}$ be the disjoint union of all the role attribute sets $S_{ID,k}$ of the user with identity ID and associated with $A_k \in \mathcal{A}^*$. S_{ID} does not satisfy (M^*, ρ^*), meaning that for each user with identity ID, there must be at least one honest authority $A_k^* \in \mathcal{A}^*$ from which the adversary never requests enough attributes to decrypt the challenged ciphertext. The honest authority A_{k^*} replies such that the corresponding role attribute set S_{ID,k^*} does not satisfy (M^*, ρ^*), meaning that the access structure (M^*, ρ^*) cannot only contain attributes from A_{k^*}. In addition, the adversary never queries the same authority twice with the same identity ID [14].

• $ID \in \mathcal{R}^*$, meaning that the user has been revoked.

• T_{dec}^* is not completely covered in T_{ID}, meaning that τ_{dec}^* and all its prefixes are not in $\mathbb{T}$, the set cover of T.

Challenge. The adversary submits two messages m_0 and m_1 of equal length. $\mathcal{C}$ picks a random bit $b \in \{0, 1\}$ and encrypts m_b using the challenged access structure (M^*, ρ^*), challenged revocation list $\mathcal{R}^*$, challenged decryption time period T_{dec}^* and challenged authority $A_{k^*} \in \mathcal{A}^*$. The resulting challenged ciphertext CT^* is given to $\mathcal{E}$.

Query Phase 2. This phase is similar to the first one.

Guess. The adversary outputs a bit $b' \in \{0, 1\}$ and wins if $b' = b$.

The advantage of the adversary in the game is defined as $Adv_{\mathcal{E}} = Pr[b' = b] - 1/2$. The revocable CP-ABE scheme is said to be *secure* if no probabilistic polynomial-time adversary has non-negligible advantage in the above game.

3.3.2 Security Proof

We give the security proof of our construction in full details below. We define a *reduction* as follows: if one can break our scheme, then one can break the Decisional q-BDHE assumption. When proving our solution secure, we need the reduction to *program* the challenged ciphertext CT^* into the public parameters PP [45]. An attribute may be associated with multiple rows in the challenged matrix M^*, meaning that the function ρ^* is not injective. This is similar to a value appearing in different leaves in a tree. For instance, let $\rho^*(i) = z$ for f_z based on the i-th row of the matrix M^*. In the reduction, we must program the parameters accordingly. Nevertheless, if $z = \rho^*(i) = \rho^*(j)$ for some i, j such that $i \neq j$, then this is a problem since we have to program both rows i and j. This implies that there may be a conflict in how we program the parameters. In the reduction, this conflict is solved by using different elements in the Decisional q-BDHE assumption. We can thus program different rows of the matrix M^* into one element corresponding to an attribute.

Assuming that the Decisional q-BDHE assumption holds, then there is no probabilistic polynomial-time adversary that can selectively break our Multi-Authority Time-Based Revocable Ciphertext-Policy Attribute-Based

Encryption scheme with a challenged matrix M^* of size $l^* \times \nu^*$ for $l^*, \nu^* < q$, a challenged revocation list $\mathcal{R}^*$ for $|\mathcal{R}^*| < q-2$ and a challenged decryption time period T^*_{dec} with binary representation $\tau^*_{dec} \in \{0,1\}^{\eta^*_{dec}}$ for $\eta^*_{dec} < T < q$, along with a set $\mathcal{A}^*$ of role authorities whose attributes are in the challenged access structure and an authority $A_{k^*} \in \mathcal{A}^*$. Let $\mathcal{E}$ be an adversary with non-negligible advantage against our solution. Let $\mathcal{C}$ be a challenger that interacts with $\mathcal{E}$ and solves the Decisional q-BDHE problem with non-negligible probability.

Initialization. The challenger is given the tuple $\vec{P} = (g_1, g_1^s, g_1^a, \cdots, g_1^{a^q}, g_1^{a^{q+2}}, \cdots, g_1^{a^{2q}}, g_2, g_2^s, g_2^a, \cdots, g_2^{a^q}, g_2^{a^{q+2}}, \cdots, g_2^{a^{2q}}) \in \mathbb{G}_1^{2q+1} \times \mathbb{G}_2^{2q+1}$ and $Q \in \mathbb{G}_T$, and should decide whether $Q = e(g_1, g_2)^{sa^{q+1}}$ by interacting with the adversary. The latter first submits the challenged access structure (M^*, ρ^*), revocation list $\mathcal{R}^*$ and decryption time period T^*_{dec}, a challenged set $\mathcal{A}^*$ of role authorities whose attributes are in (M^*, ρ^*) and a challenged honest authority $A_{k^*} \in \mathcal{A}^*$ to the challenger, such that the matrix M^* has $\nu^* \leq q$ columns, the time period has binary representation $\tau^*_{dec} = (\tau^*_1, \cdots, \tau^*_{\eta^*_{dec}}) \in \{0,1\}^{\eta^*_{dec}}$ for $\eta^*_{dec} < T_{ID} < q$ and $|\mathcal{R}^*| < q-2$, meaning that the maximum number of revoked users $R-1$ is set to $q-2$.

Setup. The challenger chooses random exponents $\theta_0, \vartheta_0, \vartheta_1, \cdots, \vartheta_T \in \mathbb{Z}_p$, $\kappa'_{k^*} \in \mathbb{Z}_p$ for $A_{k^*} \in \mathcal{A}^*$, and $\kappa_k \in \mathbb{Z}_p$ for $A_k \neq A_{k^*}$. Let $g_1^\delta = g_1^a$. It implicitly sets $\kappa_{k^*} = \kappa'_{k^*} + \theta_0 a^{q+1}$ by letting:

$$e(g_1, g_2)^{\kappa_{k^*}} = e(g_1, g_2)^{\kappa'_{k^*}} e(g_1^a, g_2^{a^q})^{\theta_0}$$

Given an authority A_k, for $k||x \in [1, U_k]$, pick at random $z_{k||x} \in \mathbb{Z}_p$. Let I be the set of indices i such that $\rho^*(i) = k||x$ (and where k is such that $A_k \in \mathcal{A}^*$). The challenger computes $h_{k||x}$ as follows:

$$h_{k||x} = g_1^{z_{k||x}} \cdot g_1^{aM^*_{i,1}} \cdot g_1^{a^2 M^*_{i,2}} \cdots g_1^{a^{\nu^*} M^*_{i,\nu^*}}$$

We observe that if $I = \emptyset$ then $h_{k||x} = g_1^{z_{k||x}}$. All the parameters are randomly distributed thanks to the value $g_1^{z_{k||x}}$.

Let $|\mathcal{R}^*| = r \leq q-2$. Let $\vec{X}_1, \cdots, \vec{X}_r$ be the corresponding vectors for the revoked list $\mathcal{R} = (ID_1, \cdots, ID_r)$, meaning that $\vec{X}_i = (1, ID_i, \cdots, ID_i^{q-2})$ for $i \in [1, r]$. Then, for each $i \in [1, r]$, let the matrix be:

$$M_{\vec{X}_i} = \begin{pmatrix} -ID_i & \cdots & -ID_i^{q-2} \\ & \mathcal{I}_{q-2} & \end{pmatrix}$$

where the $\mathcal{I}_{q-2}$ is the $(q-2) \times (q-2)$ identity matrix. Then, $\mathcal{C}$ chooses $\vec{B}_i \in \mathbb{Z}_p^{q-1}$ such that $\vec{B}_i \cdot M_{\vec{X}_i} = \vec{0}$. The vector $\vec{B}_i = (1, ID_i, \cdots, ID_i^{q-2}) = \vec{X}_i$ is the simplest candidate. In addition, for $i \in [r+1, q-1]$, let $\vec{B}_i = \vec{0}$. Now, let $\mathbf{B} = (\vec{B}_1 | \cdots \vec{B}_r | \vec{0} | \cdots | \vec{0})$ be a $(q-1) \times (q-1)$ matrix where the i-th column consists of $\vec{B}_i$ for $i \in [1, r]$ and of $\vec{0}$ for $i \in [r+1, q-1]$. Note

that the vector $(1, ID_i, \cdots, ID_i^{q-2})$ has $q-1$ elements. Therefore, we must define an $(q-2) \times (q-1)$ matrix $M_{\vec{X}_i}$ so that $(1, ID_i, \cdots, ID_i^{q-2}) M_{\vec{X}_i} = \vec{0}$ as required.

The challenger also defines $V_j = g_1^{\vartheta_j a^{q-j+1}}$ for $j \in [1, T]$ and $V_0 = \prod_{j=1}^{\eta^*} V_j^{-\tau_j^*} g_1^{\vartheta_0}$. It then defines the vector $\vec{\varpi} = (\varpi_1, \cdots, \varpi_{q-1})^\top = (a^q, \cdots, a^2)^\top$ where $\varpi_i = a^{q+1-i}$ and sets $g_1^{\vec{\varpi}} = (g_1^{\varpi_1}, \cdots, g_1^{\varpi_{q-1}})^\top$. It implicitly sets $\vec{\alpha} = \mathbf{B} \cdot \vec{\varpi} + \vec{\theta}$ by randomly choosing $\vec{\theta} = (\theta_1, \cdots, \theta_{q-1})^\top \in \mathbb{Z}_p^{q-1}$. The challenger finally sets $\vec{F} = g_1^{\mathbf{B} \cdot \vec{\varpi}} \cdot g^{\vec{\theta}} = (g_1^{\alpha_1}, \cdots, g_1^{\alpha_R})^\top = (f_1, \cdots, f_R)^\top$.

Query Phase 1. Let $S_{ID} = \cup_{A_k \in \mathcal{A}^*} S_{ID,k}$ be the disjoint union of all the role attribute sets $S_{ID,k}$ of the user with identity ID and associated with $A_k \in \mathcal{A}^*$. We observe that one could define $S_{ID} = \cup_{A_k} S_{ID,k}$ for all A_k (and not necessarily in $\mathcal{A}^*$); however, sets $S_{ID,k}$ for $A_k \notin \mathcal{A}^*$ do not satisfy (M^*, ρ^*) by design.

The adversary makes secret key queries for the user with identity ID and secret keys $RSK_{ID,k}$, TSK_{ID} such that:

- The secret keys $RSK_{ID,k}$ result from the role attribute sets $S_{ID,k}$.
- The secret key TSK_{ID} results from the range T_{ID}.

Then, at least one of the following conditions must hold:

- S_{ID} does not satisfy (M^*, ρ^*) (Case 1).
- $ID \in \mathcal{R}^*$ (Case 2).
- T_{dec}^* is not completely covered in T_{ID} (Case 3).

Case 1: S_{ID} does not satisfy (M^*, ρ^*). The challenger randomly picks $\varphi \in \mathbb{Z}_p$ and finds a vector $\vec{w} = (w_1, \cdots, w_{\nu^*}) \in \mathbb{Z}_p^{\nu^*}$ such that $w_1 = -1$ and for all i where $\rho^*(i) \in S_{ID}$, $\vec{w} \cdot M_i^* = 0$ [27]. By the definition of an LSSS, such a vector must exist since S_{ID} does not satisfy the access structure (M^*, ρ^*).

Then, $\mathcal{C}$ implicitly sets $t_{k^*} = \varphi + \theta_0(w_1 a^q + w_2 a^{q-1} + \cdots + w_{\nu^*} a^{q-\nu^*+1})$ and picks at random $t_k \in \mathbb{Z}_p$ for $A_k \neq A_{k^*}$. The challenger also chooses u_k at random for A_k.

It first computes $D'_{k,0} = g_2^{u_k}$ for all A_k. It also calculates $D_{k,0} = g_2^{t_k}$ and $D_{k,1} = g_1^{\kappa_k} g_1^{\delta t_k} g_1^{\alpha_1 u_k}$ for $A_k \neq A_{k^*}$, and $D_{k^*,0} = g_2^{\varphi} \prod_{i=1}^{\nu^*} (g_2^{a^{q+1-i}})^{w_i \theta_0} = g_2^{t_{k^*}}$

along with:

$$
\begin{aligned}
D_{k^*,1} &= g_1^{\kappa'_{k^*}} g_1^{a\varphi_{k^*}} \prod_{i=2}^{\nu^*} (g_1^{a^{q+2-i}})^{w_i\theta_0} g_1^{\alpha_1 u_{k^*}} \\
&= g_1^{\kappa'_{k^*}} g_1^{\theta_0 a^{q+1}} g_1^{a\varphi_{k^*}} g_1^{-\theta_0 a^{q+1}} \prod_{i=2}^{\nu^*} (g_1^{a^{q+2-i}})^{w_i\theta_0} g_1^{\alpha_1 u_{k^*}} \\
&= g_1^{\kappa_{k^*}} g_1^{a\varphi_{k^*}} g_1^{w_1\theta_0 a^{q+1}} \prod_{i=2}^{\nu^*} (g_1^{a^{q+2-i}})^{w_i\theta_0} g_1^{\alpha_1 u_{k^*}} \\
&\quad \text{where } w_1 = -1 \\
&= g_1^{\kappa_{k^*}} g_1^{a\varphi_{k^*}} \prod_{i=1}^{\nu^*} (g_1^{a^{q+2-i}})^{w_i\theta_0} g_1^{\alpha_1 u_{k^*}} \\
&= g_1^{\kappa_{k^*}} \Big(g_1^{\varphi_{k^*}} \prod_{i=1}^{\nu^*} (g_1^{a^{q+1-i}})^{w_i\theta_0} \Big)^a g_1^{\alpha_1 u_{k^*}} \\
&= g_1^{\kappa_{k^*}} g_1^{at_{k^*}} g_1^{\alpha_1 u_{k^*}} = g_1^{\kappa_{k^*}} g_1^{\delta t_{k^*}} g_1^{\alpha_1 u_{k^*}}
\end{aligned}
$$

For all $\tau = (\tau_1, \cdots, \tau_{\eta_\tau}) \in \mathbb{T}$, it randomly picks $\beta, v_\tau \in \mathbb{Z}_p$ and sets $D_2 = g_2^\beta$, $D_{0,\tau} = g^{v_\tau}$ and $D_{1,\tau} = g_1^{\sigma} g_1^{\alpha_1\beta} (V_0 \prod_{j=1}^{\eta_\tau} V_j^{\tau_j})^{v_\tau}$.

If $k||x \in S_{ID,k}$ for which there is no index i such that $\rho^*(i) = k||x$ (and where k is such that $A_k \notin \mathcal{A}^*$), then the challenger sets $K_{k||x} = D_{k,0}^{z_{k||x}}$. Otherwise (i.e. $k||x \in S_{ID,k}$ for which there is an index i such that $\rho^*(i) = k||x$ and where k is such that $A_k \in \mathcal{A}^*$) [45], then $\mathcal{C}$ computes:

$$
K_{k||x} = D_{k,0}^{z_{k||x}} \cdot \prod_{j=1}^{\nu^*} g_1^{a^j \varphi_k} \Big(\prod_{l=1, l\neq j}^{\nu^*} (g^{a^{q+1+j-l}})^{w_l\theta_0} \Big)^{M^*_{i,j}}
$$

Finally, $\mathcal{C}$ sets $F_{k,i} = (f_1^{-ID^{i-1}} \cdot f_i)^{u_k}$ and $E_i = (f_1^{-ID^{i-1}} \cdot f_i)^\beta$ for all A_k and $i \in [2, R]$, and $L_{j,\tau} = V_j^{v_\tau}$ for $j \in [\eta_\tau + 1, T]$ and $\tau \in \mathbb{T}$.

Case 2: $ID \in \mathcal{R}^*$. For $j \in [1, r]$, let $ID_j \in \mathcal{R}^*$ be the identity of the secret key that the adversary queries [27, 7]. The challenger defines $\tilde{\beta}_j = \beta_j - \theta_0 a^j$ and $\tilde{u}_{k^*,j} = u_{k^*,j} - \theta_0 a^j$ for a random exponent $\beta_j, u_{k^*,j} \in \mathbb{Z}_p$. It also chooses at random $u_{k,j} \in \mathbb{Z}_p$ for $A_k \neq A_{k^*}$. From the equation $\vec{\alpha} = \mathbf{B} \cdot \vec{\varpi} + \vec{\theta}$, the first coordinate of the vector $\vec{\alpha}$ is the following:

$$
\alpha_1 = \sum_{i=1}^{r} \varpi_i + \theta_1 = \sum_{i=1}^{r} a^{q+1-i} + \theta_1
$$

Then, the challenger computes $D_2 = g_2^{\beta_j} (g_2^{a^j})^{-\theta_0} = g_2^{\tilde{\beta}_j}$, $D'_{k^*,0} = g_2^{u_{k^*,j}} (g_2^{a^j})^{-\theta_0} =$

$g_2^{\tilde{u}_{k^*,j}}$ and $D'_{k,0} = g_2^{u_{k,j}}$ for $A_k \neq A_{k^*}$. For all $\tau = (\tau_1, \cdots, t_{\eta_\tau}) \in \mathbb{T}$, it randomly chooses $v_\tau \in \mathbb{Z}_p$, and computes $D_{0,\tau} = g_2^{v_\tau}$ along with:

$$D_{1,\tau} = g_1^{\sigma} g_1^{\alpha_1\beta_j - \alpha_1\theta_0 a^j} (V_0 \prod_{j=1}^{\eta_\tau} V_j^{\tau_j})^{v_\tau} = g_1^{\sigma} g_1^{\alpha_1\tilde{\beta}_j} (V_0 \prod_{j=1}^{\eta_\tau} V_j^{\tau_j})^{v_\tau}$$

In addition, it picks at random $t_k \in \mathbb{Z}_p$ for all A_k, and calculates $D_{k,1} = g_1^{\kappa_k} g_1^{\delta t_k} g_1^{\alpha_1 u_{k,j}}$ for $A_k \neq A_{k^*}$, along with:

$$\begin{aligned}
D_{k^*,1} &= g_1^{\kappa'_{k^*}} f_1^{u_{k^*,j}} (g_1^{a^j\theta_1} \prod_{i=1, i\neq j}^{r} g_1^{a^{q+1-i+j}})^{-\theta_0} g_1^{at_{k^*}} \\
&= g_1^{\kappa'_{k^*}} g_1^{\alpha_1 u_{k^*,j}} g_1^{\theta_0 a^{q+1}} (g_1^{\theta_1} \prod_{i=1}^{r} g_1^{a^{q+1-i}})^{-\theta_0 a^j} g_1^{\delta t_{k^*}} \\
&= g_1^{\kappa_{k^*}} g_1^{\delta t_{k^*}} g_1^{\alpha_1 u_{k^*,j}} (g_1^{\alpha_1})^{-\theta_0 a^j} \\
&= g_1^{\kappa_{k^*}} g_1^{\delta t_{k^*}} g_1^{\alpha_1 u_{k^*,j} - \alpha_1\theta_0 a^j} = g_1^{\kappa_{k^*}} g_1^{\delta t_{k^*}} g_1^{\alpha_1 \tilde{u}_{k^*,j}}
\end{aligned}$$

Let $\mathbb{F}_{k,j} = (F_2, \cdots, F_R)^\top$ be the secret key component for the identity ID_j. We recall that $\vec{\varpi} = (\varpi_1, \cdots, \varpi_{q-1})^\top = (a^q, \cdots, a^2)^\top$ with $\varpi_i = a^{q+1-i}$ and $g_1^{\vec{\varpi}} = (g_1^{\varpi_1}, \cdots, g_1^{\varpi_{q-1}})^\top$. First, we observe that $\mathcal{C}$ can compute $g_1^{a^j M_{\vec{X}_j}^\top \mathbf{B}\vec{\varpi}}$ because the j-th column of $M_{\vec{X}_j}^\top \mathbf{B}$ is equal to $\vec{0}$. The challenger computes $\mathbb{F}_{k,j} = g_1^{u_{k,j} M_{\vec{X}_j}^\top \vec{\alpha}}$ for $A_k \neq A_{k^*}$, as well as:

$$\begin{aligned}
\mathbb{F}_{k^*,j} &= g_1^{u_{k^*,j} M_{\vec{X}_j}^\top \vec{\alpha}} \cdot g_1^{-\theta_0 a^j M_{\vec{X}_j}^\top \mathbf{B}\vec{\varpi}} \cdot g_1^{-\theta_0 a^j M_{\vec{X}_j}^\top \vec{\theta}} \\
&= g_1^{u_{k^*,j} M_{\vec{X}_j}^\top \vec{\alpha}} \cdot g_1^{-\theta_0 a^j M_{\vec{X}_j}^\top \vec{\alpha}} \\
&= g_1^{(u_{k^*,j} - \theta_0 a^j) M_{\vec{X}_j}^\top \vec{\alpha}} = g_1^{\tilde{u}_{k^*,j} M_{\vec{X}_j}^\top \vec{\alpha}}
\end{aligned}$$

It also calculates $\mathbb{E}_j$ as follows:

$$\begin{aligned}
\mathbb{E}_j &= g_1^{\beta_j M_{\vec{X}_j}^\top \vec{\alpha}} \cdot g_1^{-\theta_0 a^j M_{\vec{X}_j}^\top \mathbf{B}\vec{\varpi}} \cdot g_1^{-\theta_0 a^j M_{\vec{X}_j}^\top \vec{\theta}} \\
&= g_1^{\beta_j M_{\vec{X}_j}^\top \vec{\alpha}} \cdot g_1^{-\theta_0 a^j M_{\vec{X}_j}^\top \vec{\alpha}} \\
&= g_1^{(\beta_j - \theta_0 a^j) M_{\vec{X}_j}^\top \vec{\alpha}} = g_1^{\tilde{\beta}_j M_{\vec{X}_j}^\top \vec{\alpha}}
\end{aligned}$$

We recall that $R = q - 1$ and we denote $M_{\vec{X}_j, i-1}^\top$ as the $(i-1)$-th row of $M_{\vec{X}_j}^\top$, then for $i \in [2, R]$, we have $F_{k,i} = (f_1^{-ID_j^{i-1}} \cdot f_i)^{u_{k,j}}$ for $A_k \neq A_{k^*}$, and

the following:

$$\begin{aligned} F_{k^*,i} &= g_1^{\tilde{u}_{k^*,j} M^{\top}_{\vec{X}_j,i-1} \vec{\alpha}} = g_1^{\tilde{u}_{k^*,j}(-ID_j^{i-1}\alpha_1+\alpha_i)} = (f_1^{-ID_j^{i-1}} \cdot f_i)^{\tilde{u}_{k^*,j}} \\ E_i &= g_1^{\tilde{\beta}_j M^{\top}_{\vec{X}_j,i-1} \vec{\alpha}} = g_1^{\tilde{\beta}_j(-ID_j^{i-1}\alpha_1+\alpha_i)} = (f_1^{-ID_j^{i-1}} \cdot f_i)^{\tilde{\beta}_j} \end{aligned}$$

Finally, the challenger computes $K_{k||x} = h_{k||x}^{t_k}$ for $k||x \in S_{ID,k}$ and $L_{j,\tau} = V_j^{v_\tau}$ for $j \in [\eta_\tau + 1, T]$ and $\tau \in \mathbb{T}$.

Case 3: τ^*_{dec} and all its prefixes are not in $\mathbb{T}$. For all $\tau = (\tau_1, \cdots, \tau_{\eta_\tau}) \in \mathbb{T}$, let $\tau_{\eta_\tau+1}, \cdots, \tau_q = 0$ and $\tau^*_{\eta^*_{dec}+1}, \cdots, \tau^*_q = 0$. Let $\eta' \leq \eta^*_{dec}$ be the smallest index such that $\tau_{\eta'} \neq \tau^*_{\eta'}$.

$\mathcal{C}$ randomly chooses $t_k, u_k \in \mathbb{Z}_p$ for $A_k \neq A_{k^*}$ along with $t_{k^*}, u_{k^*} \in \mathbb{Z}_p$, and sets $u_{k^*} = u'_{k^*} - \frac{\theta_0}{\alpha_1} a^{\eta'}$. It then computes $D_{k,0} = g_2^{t_k}$ for all A_k. It also calculates $D'_{k^*,0} = g_2^{u'_{k^*} - \frac{\theta_0}{\alpha_1} a^{\eta'}} = g_2^{u_{k^*}}$ and:

$$\begin{aligned} D_{k^*,1} &= g_1^{\kappa'_{k^*}} g_1^{\theta_0 a^{q+1-\eta'}} g_1^{\delta t_{k^*}} g_1^{\alpha_1 u'_{k^*}} \\ &= g_1^{\kappa'_{k^*}} g_1^{\theta_0 a^{q+1}} g_1^{\delta t_{k^*}} g_1^{\alpha_1 u'_{k^*}} g_1^{-\theta_0 a^{\eta'}} = g_1^{\kappa_{k^*}} g_1^{\delta t_{k^*}} g_1^{\alpha_1 u_{k^*}} \end{aligned}$$

It sets $D'_{k,0} = g_2^{u_k}$ and $D_{k,1} = g_1^{\kappa_k} g_1^{\delta t_k} g_1^{\alpha_1 u_k}$ for $A_k \in \mathcal{A} \setminus \{A_{k^*}\}$. In addition, the challenger picks at random $\beta, v_\tau \in \mathbb{Z}_p$ and calculates $D_2 = g_2^{\beta}$ and $D_{0,\tau} = g_2^{v_\tau}$.

We recall that $V_j = g_1^{\vartheta_j a^{q-j+1}}$ for $j \in [1, T]$ and $V_0 = \prod_{j=1}^{\eta^*} V_j^{-\tau^*_j} \; g_1^{\vartheta_0}$.

For all τ, it sets:

$$\begin{aligned}
D_{1,\tau} &= g_1^{\sigma} g_1^{\alpha_1 \beta} g_1^{\vartheta_0 v_\tau} g_1^{\vartheta_{\eta'} a^{q-\eta'+1}(\tau_{\eta'}-\tau^*_{\eta'}))v_\tau} \prod_{j=\eta'+1}^{\eta_\tau+1} g_1^{\vartheta_j a^{q-j+1}\tau_j^* v_\tau} \\
&= g_1^{\sigma} g_1^{\alpha_1 \beta} g_1^{(\vartheta_0+\vartheta_{\eta'} a^{q-\eta'+1}(\tau_{\eta'}-\tau^*_{\eta'}))v_\tau} \prod_{j=1}^{\eta'-1} V_j^{-\tau_j^* v_\tau} \prod_{j=1}^{\eta'-1} V_j^{\tau_j v_\tau} \\
&\quad \prod_{j=\eta'+1}^{\eta_\tau+1} g_1^{\vartheta_j a^{q-j+1}\tau_j^* v_\tau} \text{ since } \tau_j = \tau_j^* \text{ if } j < \eta' \\
&= g_1^{\sigma} g_1^{\alpha_1 \beta} (\prod_{j=1}^{\eta'} V_j^{-\tau_j^*} g_1^{\vartheta_0})^{v_\tau} \prod_{j=1}^{\eta'} V_j^{\tau_j v_\tau} \prod_{j=\eta'+1}^{\eta_\tau+1} g_1^{\vartheta_j a^{q-j+1}\tau_j^* v_\tau} \\
&= g_1^{\sigma} g_1^{\alpha_1 \beta} (V_0 \prod_{j=1}^{\eta'} V_j^{\tau_j})^{v_\tau} \prod_{j=\eta'+1}^{\eta_\tau+1} V_j^{\tau_j v_\tau} \\
&= g_1^{\sigma} g_1^{\alpha_1 \beta} (V_0 \prod_{j=1}^{\eta_\tau+1} V_j^{\tau_j})^{v_\tau} \\
&= g_1^{\sigma} g_1^{\alpha_1 \beta} (V_0 \prod_{j=1}^{\eta_\tau} V_j^{\tau_j})^{v_\tau} \text{ since } \tau_{\eta_\tau+1} = 0
\end{aligned}$$

The challenger also sets $K_{k||x} = h_{k||x}^{t_k}$ for $k||x \in S_{ID,k}$. For $i \in [2, R]$, it computes $E_i = (f_1^{-ID^{i-1}} \cdot f_i)^{\beta}$, $F_{k^*,i} = (f_1^{-ID^{i-1}} \cdot f_i)^{u_{k^*}} = (f_1^{-ID^{i-1}} \cdot f_i)^{u'_{k^*} - \frac{\theta_0}{\alpha_1} a^{\eta'}}$ and $F_{k,i} = (f_1^{-ID^{i-1}} \cdot f_i)^{u_k}$ for $A_k \neq A_{k^*}$. For $j \in [\eta_\tau + 1, T]$ and $\tau \in \mathbb{T}$, the challenger computes $L_{j,\tau} = g_1^{\vartheta_j a^{q-j+1} v_\tau} = V_j^{v_\tau}$.

Challenge. The adversary submits two messages m_0 and m_1 of equal length. $\mathcal{C}$ picks a random bit $b \in \{0, 1\}$ and encrypts m_b under the access structure (M^*, ρ^*), the revocation list $\mathcal{R}^*$, the authority set $\mathcal{A}^*$, the authority $A_{k^*} \in \mathcal{A}^*$ and decryption time period T^*_{dec} with binary representation τ^* as follows. The challenger first computes $C'_0 = g_2^s$ along with:

$$C_0 = m_b \cdot Q^{\theta_0} \cdot e(g_1^s, g_2^{\kappa'_{k^*}}) \cdot e(g_1^s, g_2^{\sigma}) \cdot \prod_{A_k \in \mathcal{A}^* \setminus \{A_{k^*}\}} e(g_1^s, g_2^{\kappa_k})$$

It then creates C''_0 as follows [7]. Let $\mathcal{R}^* = (ID_1, \cdots, ID_r)$ and $\mathcal{F}_{\mathcal{R}^*}(Z) = (Z - ID_1) \cdots (Z - ID_r) = y_1 + y_2 Z + \cdots + y_r Z^{r-1} + y_{r+1} Z^r$. If $r+1 < R$, then the coefficients $y_{r+2}, \cdots, y_R$ are set to be equal to 0. Let $\vec{Y} = (y_1, \cdots, y_R)^\top$ satisfy $\langle \vec{X}_j, \vec{Y} \rangle = 0$ for $j \in [1, r]$. We claim that $\vec{Y}^\top \cdot \mathbf{B} \cdot \vec{\varpi} = 0$ [27]. Hence, we obtain that $\langle \vec{Y}, \vec{\alpha} \rangle = \langle \vec{Y}, \vec{\theta} \rangle$. It then sets $C''_0 = (g_1^s)^{\langle \vec{Y}, \vec{\theta} \rangle}$.

We also observe that the terms g^{a^i} in V_i are canceled out since the challenged time is $\tau^* = (\tau_1^*, \cdots, \tau_{\eta^*}^*)$. The challenger computes $C'''_0 = (g_1^s)^{\vartheta_0}$.

The terms C_i are computed as follows [45]. Since the terms $h^s_{\rho^*(i)}$ contain terms of the form $g^{a^i s}$ that cannot be simulated, the secret splitting technique is thus required, and the latter terms can be canceled out. The challenger chooses exponents $\gamma'_2, \cdots, \gamma'_{\nu^*} \in \mathbb{Z}_p$ and then shares the secret s using the vector:

$$\vec{v^*} = (s, sa + \gamma'_2, \cdots, sa^{\nu^*-1} + \gamma'_{\nu^*}) \in \mathbb{Z}_p^{\nu^*}$$

This permits the terms $h^s_{-\rho^*(i)}$ cancel out with the terms $g_1^{a\lambda_i}$. Then, for $i \in [1, \nu^*]$, the challenger computes $C_i = (g_1^s)^{-z_{\rho^*(i)}} \cdot \prod_{j=2}^{\nu^*} (g^a)^{M^*_{i,j}\gamma'_j}$. In order to see the correct simulation of the terms C_i, we first define:

$$\lambda_i^* = \langle \vec{v^*}, \vec{M_i^*} \rangle = sM^*_{i,1} + (sa + \gamma'_2)M^*_{i,2} + \cdots + (sa^{\nu^*-1} + \gamma'_{\nu^*})M^*_{i,\nu^*}$$

Thus, the correct distribution of C_i should be as follows:

$$\begin{aligned} C_i &= g_1^{a\lambda_i^*} h^{-s}_{\rho^*(i)} \\ &= g_1^{a(sM^*_{i,1} + (sa+\gamma'_2)M^*_{i,2} + \cdots + (sa^{\nu^*-1}+\gamma'_{\nu^*})M^*_{i,\nu^*})} \cdot g_1^{-sz_{\rho^*(i)}} \\ &\quad \cdot g_1^{-(saM^*_{i,1} + \cdots + sa^{\nu^*}M^*_{i,\nu^*})} \\ &= (g_1^s)^{-z_{\rho^*(i)}} \cdot g_1^{aM^*_{i,2} + \cdots + aM^*_{i,\nu^*}} = (g_1^s)^{-z_{\rho^*(i)}} \cdot \prod_{j=2}^{\nu^*} (g^a)^{M^*_{i,j}\gamma'_j} \end{aligned}$$

Query Phase 2. The same as in Phase 1.

Guess. The adversary outputs a guess $b' \in \{0, 1\}$ for b. If $b = b'$, then the challenger outputs 0 to guess that $Q = e(g_1, g_2)^{sa^{q+1}}$. Otherwise, $\mathcal{C}$ outputs 1 to guess that Q is a random element of the group $\mathbb{G}_T$.

When Q is equal to $e(g_1, g_2)^{sa^{q+1}}$, then $\mathcal{C}$ gives a perfect simulation, and its advantage is the same as the adversary's one. When Q is a random element of $\mathbb{G}_T$, then the message m_b is completely hidden from $\mathcal{E}$, and thus $Pr[\mathcal{C}(\vec{P}, Q \in_R \mathbb{G}_T) = 0] = 1/2$. Hence, the challenger can solve the Decisional q-BDHE problem with non-negligible advantage.

From selective security to static security We have proved our scheme selectively secure in the standard model, meaning that the adversary submits at the really beginning of the game a challenged access structure (M^*, ρ^*), a challenged authority set $\mathcal{A}^*$, a challenged honest authority $A_{k^*} \in \mathcal{A}^*$, a challenged revocation list $\mathcal{R}^*$ and a challenged decryption time period T^*_{dec} to $\mathcal{C}$, before receiving the public parameters and authorities' public keys from the challenger. A possible improvement is to consider static security, where all challenged items and queries submitted by $\mathcal{E}$ are sent to the challenger directly after seeing the public parameters. Such improvement would be done

following Rouselakis and Waters' technique [39]. Their technique enables the challenger of the security reduction to separate an unauthorized set of the matrix rows and pass over this set for the remaining of the reduction. $\mathcal{C}$ then ignores the contributions of these rows even in the construction of the challenged ciphertext.

4 Evaluation

Since we have extended the solution in [27] to obtain our solution, we choose to compare both of them. The advantage is that we can check what our extension has gained and/or lost compared to the original solution, from a technical perspective. However, shifting from one authority to multiple authorities brings substantial overheads. Hence, the multi-authority aspect cannot be compared with the original scheme accurately. Instead, we compare our solution with the Multi-Authority ABE scheme proposed in [39] to evaluate our multi-authority setting.

4.1 Theoretical Analysis

Scheme	LYZL [27]	Ours
Public key material	$(U+R+T+3)\mathbb{G}$ $+\mathbb{G}_T$	$(U+R+T+3)\mathbb{G}_1$ $+\mathbb{G}_2+(N+1)\mathbb{G}_T$
User secret key material	$(R+1+S$ $+\frac{1}{2}T(T+3))\mathbb{G}$	$(N(R+1)+S$ $+\frac{1}{2}T(T+3))\mathbb{G}_1$ $+2(N+1)\mathbb{G}_2$
Ciphertext	$(l+3)\mathbb{G}+\mathbb{G}_T$	$(l+2)\mathbb{G}_1$ $+\mathbb{G}_2+\mathbb{G}_T$
Decryption time (# of pairings)	$4+2l$	$3N+2l$ $+2$

Table 2: Size of keys and ciphertexts, and number of pairing operations during decryption.

Table 2 compares the efficiency of our scheme and of the LYZL scheme [27]. Let $N+1$ be the number of authorities in our system (i.e. N role authorities and 1 time authority). Let $R-1$ be the maximum number of revoked users and T be the depth of the tree. Let U be the number of attributes in the whole universe $\mathcal{U}$, and S be the total number of role attributes of the user. Let l denote the number of attributes used in the decryption. The public key material contains the public parameters along with the authorities' public keys. The user secret key material contains the user keys issues by all the involved authorities.

At first sight, the scheme in [27] seems to be more efficient than ours. We easily observe that extra elements in our case are due to the multi-authority setting. We recall that a single authority is responsible of generating user key material in [27], while $N + 1$ authorities are involved in our system. By setting $N = 1$, the performance of our system is equivalent to the one of the LYZL scheme. Hence, our attempt to reduce the key escrow problem is at the expense of practicality.

Yet, the tree storage costs do not appear in Table 2. We recall that Liu et al. [27] suggest one common value z as the number of children per node. By setting $z = 31$, many dummy nodes are created, and storage costs get cumbersome. Let us compare the two tree-based methods with an IoT-related example. Suppose that an actuator is granted to request data from its connected sensors on a period of 7 days, starting on "04 January 2022". In both schemes, we suppose that the starting time (root of the tree) is "01 January 2022". Following the tree-based method used in [27] with leaf nodes as days (and $T = 4$ representing year, month and day levels), the actuator obtains 7 keys, one for each day. Following our method with a time interval of 16 days as in Figure 3 (i.e. $T = 5$), the actuator receives 3 keys. Therefore, our technique is more efficient when dealing with data valuable over short time periods, say on a daily basis. Moreover, our solution is not limited to 16-day time periods; we can define longer periods as long as the tree size remains reasonable.

While the LYZL solution [27] is interesting in some cases, e.g. within a company where the system is setup in a narrow, private environment and time periods are of the order of months or years, it does not fit our IoT scenario which involves time-sensitive data and numerous heterogeneous devices. On the other hand, our solution is attractive in IoT environments, with a profitable framework for short time periods and an advantageous security level meeting IoT requirements.

4.2 Implementation and Practical Analysis

In this section, we aim to check the practicality of the scheme in an IoT environment while considering the limited capacities of such framework, in terms of computation, communication and storage. We examine timing at setup, encryption and decryption phases. More precisely, we are interested in verifying the practicality of our Multi-Authority Time-Based Revocable Ciphertext-Policy Attribute-Based Encryption scheme by testing various elliptic curves, and varying the revocation list length, role attribute sets' size and binary tree depth.

Testing environment We test our solution on a Raspberry Pi 4B with a Quad Core ARM64 Cortex-A72 CPU running at 1.5 GHz with 8 GB of 3200 MHz SDRAM. The programming language is `Python`3.6. We use the

Python-based Charm Crypto library [4], an open source framework developed for rapid prototyping of cryptographic systems. For all our benchmarks, we execute $1,000$ tests and calculate the average time out of those $1,000$ outputs. Time metric is millisecond.

Parameter selection In an IoT environment, access policies contain up to 30 attributes, and devices are allocated around 10 attributes [6, 47]. Roles can be related to the devices' functionalities, locations and permissions to specific operations such as Read and Write. For example, let the number of roles authorities be $N = 3$. Therefore, given a role authority A_k, we set its universe $U_k = 10$ and the user attribute set $S_{ID,k} = 4$. We consider the number l of attributes used in the decryption be equal to $\lceil \frac{S_k}{2} \rceil$, that is 2. Our evaluation considers a short period of time, that aims to represent a realistic feature in IoT networks as illustrated in Figure 3. Of course, longer periods of time are possible.

We assume that role authorities' algorithms RAKeyGen and RUKeyGen will be run in parallel. W.l.o.g., we only consider one authority when testing our solution. Unless specified, we consider the following parameters for our testing:

- The role authority A_k has 4 attributes.
- The user has 2 attributes w.r.t. the role authority A_k.
- The time period consists of 16 days, which means that the depth of the binary tree is $T = 5$.
- There are 4 attributes in the access policy when encrypting a message, which means that the matrix M has 4 rows.
- There are 2 attributes from the user's attribute set that are needed for successful decryption, which means that 2 rows will be used in M.

We choose to not test bigger numbers of attributes for two reasons: first, we aim to implement our solution in a realistic context, hence following the aforementioned numerical suggestions from the literature [6, 47]; second, we believe that the efficiency of our solution will fail with large attribute numbers since many components (such as key material, ciphertext and decryption process) depend on those numbers.

Elliptic curve selection Implementing our solution requires to generate cyclic groups of prime orders built from an elliptic curve. The Charm Crypto library [4] proposes several elliptic curves, offering different levels of security.

We run the algorithms of our Multi-Authority Time-Based Revocable Ciphertext-Policy Attribute-Based Encryption scheme based on the following elliptic curves: SS512 and SS1024 with 512-bit and 1024-bit base fields respectively [16]; MNT curves with 159-bit, 201-bit and 224-bit base fields respectively [34]. The results are shown in Tables 4 and 5. We also provide the security level offered by each tested elliptic curve in Table 3.

Curve	Security level
SS512	80
SS1024	112
MNT159	70
MNT201	90
MNT224	100

Table 3: Security levels (in bits) of the tested elliptic curves [39].

Curve/Algo.	Setup	RAKeyGen	TAKeyGen	RUKeyGen
SS512	8.1	4.0	9.4	21.3
SS1024	92.7	4.3	5.8	240.6
MNT159	7.5	2.3	3.5	12.9
MNT201	10.1	3.0	3.9	16.6
MNT224	13.0	3.8	5.0	21.5

Table 4: Average running times (in milliseconds) of algorithms Setup, RAKeyGen, TAKeyGen and RUKeyGen with different elliptic curves.

Curve/Algo.	TUKeyGen	Encrypt	Decrypt
SS512	31.7	26.8	18.7
SS1024	387.9	193.7	300.4
MNT159	21.8	21.0	37.8
MNT201	27.9	23.5	41.9
MNT224	36.7	28.1	61.2

Table 5: Average running times (in milliseconds) of algorithms TUKeyGen, Encrypt and Decrypt with different elliptic curves.

Curve /Algo.	Global Setup	Auth Setup	KeyGen	Encrypt	Decrypt
SS512	2.7	1.2	15.5	25.5	10.1
MNT159	5.4	1.1	47.9	44.5	30.0
MNT201	7.0	1.4	56.4	68.7	37.3
MNT224	8.8	1.8	74.1	102.7	52.4

Table 6: Average running times (in milliseconds) of all algorithms of the RW scheme [39] with different elliptic curves.

Algorithms RAKeyGen and TAKeyGen have similar time results for all elliptic curves. Except with SS1024, algorithm Setup and Encrypt also get comparable time outputs. While SS1024 offers the highest security level among all the elliptic curves, it noticeably impacts the running times of most of the algorithms, namely Setup, RUKeyGen, TUKeyGen, Encrypt and Decrypt. Decryption with curves MNT159 and MNT201 requires around twice the time needed for decryption with the curve SS512, for a similar security level. While MNT224 guarantees a higher security level than SS512, the decrypting algorithm takes three times longer for the former.

For a realistic trade-off between efficiency and security, the curve SS512 is the most appropriate choice. Therefore, for subsequent testings, the curve SS512 is used to build our cyclic groups of prime order during the setup phase.

Scheme comparison based on elliptic curves Rouselakis and Waters (RW) scheme [39] is a Multi-Authority ABE scheme similar to ours. Indeed, the authors proposed a CP-ABE scheme where users' keys are generated by several authorities, based on their attributes. However, we note that the time-based tree framework is missing in [39]: authorities only deliver role attributes to users, and no time-related parameters are used. Therefore, w.l.o.g., we only compare our solution with RW scheme based on role authorities' algorithms, and omit time authorities' algorithms, namely TAKeyGen and TUKeyGen. We choose to evaluate RW scheme based on the same elliptic curves that we tested for our solution. However, we excluded SS1024 since running times were high and thus such curve is not realistic for an IoT application.

Results for the RW scheme are given in Table 6. We compare those results with the ones given in Tables 4 and 5. RW algorithm GlobalSetup corresponds to our algorithm Setup, AuthSetup to RAKeyGen and KeyGen to RUKeyGen. Our algorithms Setup and RAKeyGen are slightly slower than RW algorithms GlobalSetup and AuthSetup respectively, based on the curve SS512. Nevertheless, this time difference does not impact much the applica-

R	Setup	RUKeyGen	TUKeyGen	Encrypt	Decrypt
5	8.1	21.3	31.7	26.8	18.7
10	13.0	49.0	57.7	26.8	18.0
15	17.3	80.1	91.2	27.0	17.6
20	22.8	129.5	135.9	26.8	17.8
25	26.7	176.3	187.2	27.2	17.8
30	31.0	238.9	243.6	27.5	18.2

Table 7: Average running times (in milliseconds) of algorithms Setup, RUKeyGen, TUKeyGen, Encrypt and Decrypt with different values for R.

bility of our scheme in an IoT environment since those algorithm are likely run by powerful entities and only once. Encryption process times are similar in both schemes. RW decryption process is slightly faster than ours. Nevertheless, as above, we argue that this process is likely managed by a more powerful entity, such as a gateway connected to the actuator.

Our Multi-Authority CP-ABE scheme outperforms RW scheme built on MNT curves, regarding key generation, encryption and decryption. Using those MNT curves allows our solution to be faster and more efficient, such that those qualifying adjectives are important in IoT environments. Hence, selecting those elliptic curves for our scheme is still a reasonable choice for IoT applications.

Revocation list Let $R-1$ be the maximum number of revoked users. Let $\mathcal{R} = (ID_1, \cdots, ID_r)$ be the revocation list containing $r < R$ revoked users at a given time. We are interested in observing how those parameters R and r affect the execution time of our algorithms. We text all algorithms except RAKeyGen and TAKeyGen since they do not take as input R or r. We conduct a first experiment with $R \in \{5, 10, 15, 20, 25, 30\}$. Result are shown in Table 7 and Figure 6.

We observe that the value for R has low impact on the encryption and decryption processes as expected. Indeed, those processes rather depend on the value given to r that is fixed in this experiment. The running time of the algorithm Setup depends on R, with a light increase with larger R values. The running times of the algorithms RUKeyGen and TUKeyGen are linear with R, such that those key generation processes are noticeably affected by the value given to the parameter R. As we suggested, the maximum length R of the revocation list $\mathcal{R}$ must remain reasonable to permit an interesting trade-off between key update frequency and user revocation list length. We suggest that R can be set up to 15 in order to keep the running time of the algorithm RUKeyGen below 100 milliseconds A larger value would negatively impact the applicability of our solution in IoT environments by

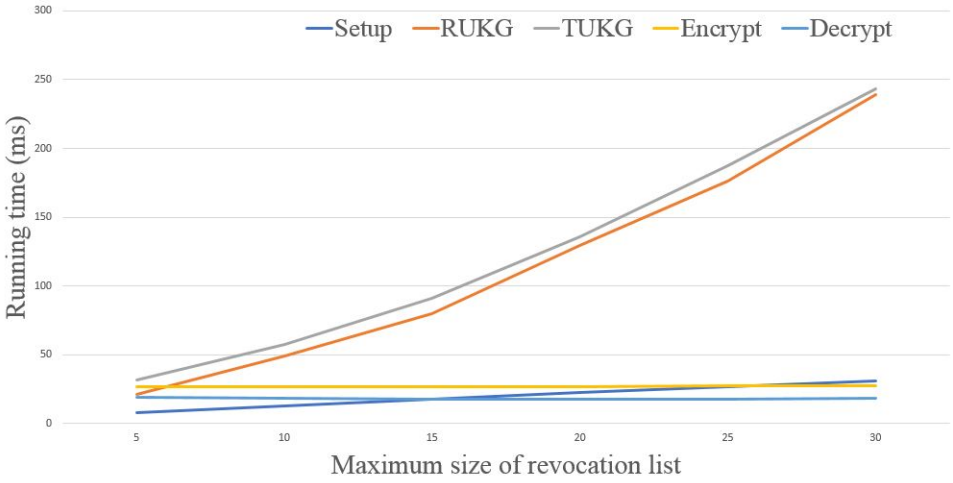

Figure 6: Average running times (in milliseconds) of algorithms Setup, RUKeyGen, TUKeyGen, Encrypt and Decrypt with different values for R.

noticeably slowing down the management of the keys of numerous devices in the network.

We then prepare a second experiment with $R = 10$ and $r \in [1, 9]$. The revocation list $\mathcal{R}$, and thus the parameter r are inputs of the algorithms Encrypt and Decrypt only. We are interested in seeing how those two algorithms are affected by the number r of revoked users in $\mathcal{R}$. Let $R = 10$, thus the maximum number of revoked users is 9. We run the algorithms Encrypt and Decrypt by varying the number r of revoked users from 1 for 9. The results are shown in Figure 7. Encryption and decryption timings linearly increase with the number of revoked users in $\mathcal{R}$. The effect is stronger for encryption than for decryption. Indeed, the value $\mathcal{F}_{\mathcal{R}}(Z)$ based on $\mathcal{R}$ must be computed and a set of its coefficients y must be defined, requiring more computing resources with the number of revoked users being larger.

Role authorities' attribute set The number of attributes controlled by a role authority A_k has an influence on the time needed to generate the keys PK_k and SK_k of this role authority A_k. Let this number vary between 1 and 20. The results are shown in Figure 8. Each attribute adds around 1 millisecond in computing the key.

By having multiple authorities, we manage to dispatch the role attributes among them such that each role authority has only a small subset of them, enabling a faster key generation while mitigating key escrow and single point of failure.

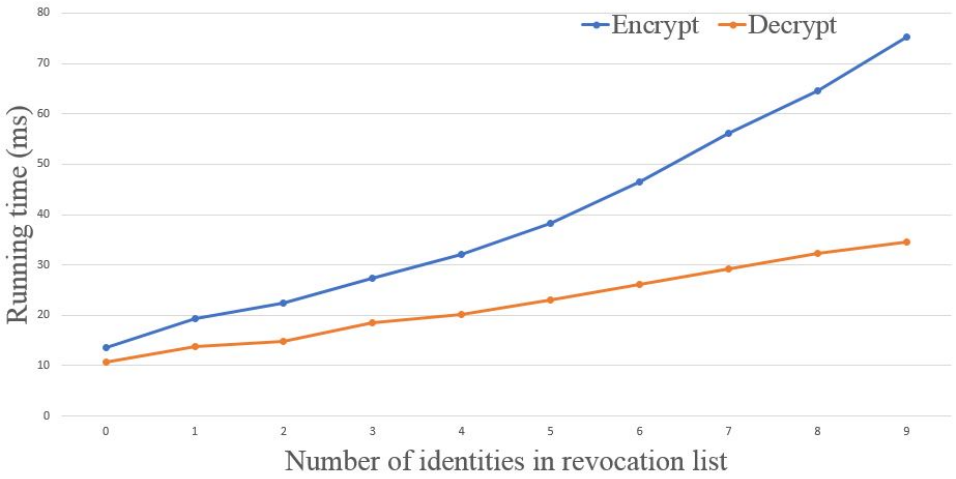

Figure 7: Average running times (in milliseconds) of algorithms Encrypt and Decrypt based on the number of revoked users in the revocation list $\mathcal{R}$ such that $R = 10$.

Users' role attribute set The number of users' role attributes has an impact of the time required to generate the keys $RSK_{ID,k}$ of those users. Let the number of attributes controlled by a role authority A_k be 15. Let the number of role attributes given to a user vary between 2 and 15. The results are shown in Figure 9. We observe that the time needed to generate the user key $RSK_{ID,k}$ is linear in the number of role attributes given to the user, reaching around 34ms for 15 attributes. Each attribute adds around 1 millisecond in computing the key.

Therefore, by having several role authorities, we manage to share the computing resources needed to compute the users' keys, enabling a faster key generation. Moreover, we mitigate the key escrow problem that one-authority systems may suffer.

Time binary tree In Section 2, we presented a binary tree of depth $T = 5$, with 16 leaves. In this experiment, we compare such depth value with higher depth values, i.e. up to $T = 12$. Both the algorithms TAKeyGen and TUKeyGen depend on the parameter T. More precisely, those algorithms should require a noticeable amount of time to construct the tree and to run the set cover process that finds the minimum number of nodes to represent the time period. The results are shown in Table 8. When the value of the depth T is set between 5 and 8, the time required to build the binary tree and to find the minimum cover set is strictly less than 1 millisecond. When $T = 9$ and above, the time increases exponentially. Indeed, the number of leaves scales with 2^T.

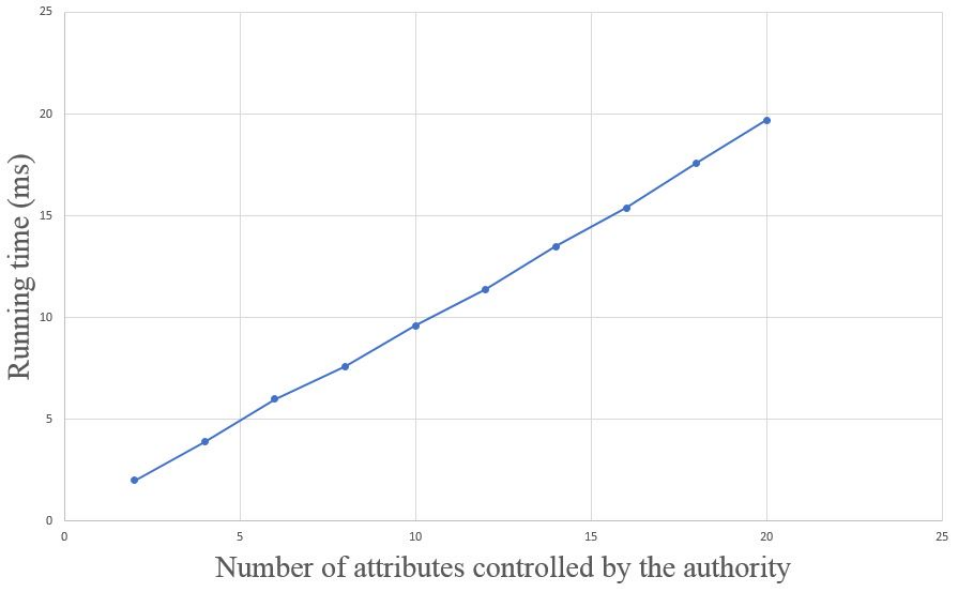

Figure 8: Average running times (in milliseconds) of algorithm RAKeyGen based on the number of role attributes.

T	5	6	7	8	9	10	11	12
Time	< 1	< 1	< 1	< 1	1	2	3	6

Table 8: Average running times (in milliseconds) of combined algorithms TAKeyGen and TUKeyGen based on different values T.

In an IoT context, it is important to keep the time required to build the binary tree below 1 millisecond. For instance, temperature sensors collect data once every few minutes, thus may require very short access time periods such as few days, and may not need big trees to represent those short periods. In addition, defining trees of reasonable depth moderates storage costs of the system. Hence, setting a depth $T = 5$ is a judicious choice considering our IoT context.

Access policy during encryption The access policy is represented by the LSSS matrix M and the function ρ. The number of rows in M has an effect on the running time of the algorithm Encrypt. We recall that the number of rows in M is exactly the number of attributes in the access policy. In this experiment, let the number of attributes in the access policy vary from 1 to 12. We are interested in evaluating the algorithm Encrypt based on the number of attributes in the access policy, and thus based on the number of rows in M. The results are shown in Table 4.2. The running

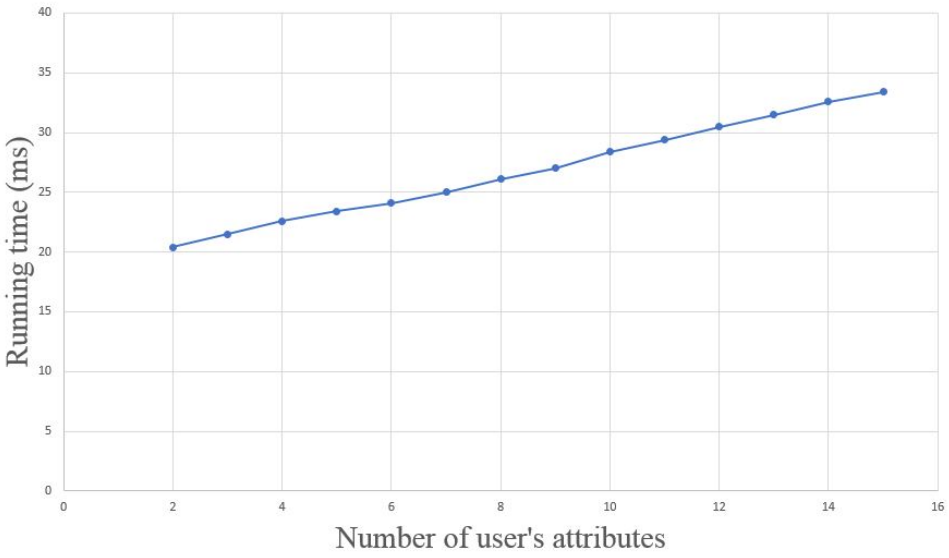

Figure 9: Average running times (in milliseconds) of algorithm RUKeyGen based on the number of role attributes.

time of Encrypt is linear in the number of attributes in the access policy, being slightly larger than 40ms for 12 attributes/rows. 2 extra milliseconds are added for each extra attribute/row.

We previously suggested that there could be up to 30 attributes in the access policy [6, 47]. Given such number, we could expect a message being encrypted in 76ms, that is a reasonable time for most IoT applications.

Access policy during decryption The algorithm Decrypt depends on the number of matrix rows utilized in order to recover the message successfully. We assume `AND` operations in the access policy only. In this experiment, we vary the number of attributes required for a successful decryption, and thus the number of rows, between 1 and 12. We are interested in evaluating the algorithm Decrypt based on the number of attributes in the access policy needed to recover the message in its entirety, and thus based on the number of rows in M that are used for decryption. The results are shown in Table 4.2. The running time of Decrypt is linear in the number of attributes in the access policy that are required for successful decryption, being almost 35ms for 12 attributes/rows. 1.5 extra milliseconds are added for each extra attribute/row.

We suggested that there are up to 10 attributes needed to decrypt a ciphertext [6, 47]. Hence, considering such number, the time required to decrypt a message would be around 30ms, that is acceptable in IoT environments.

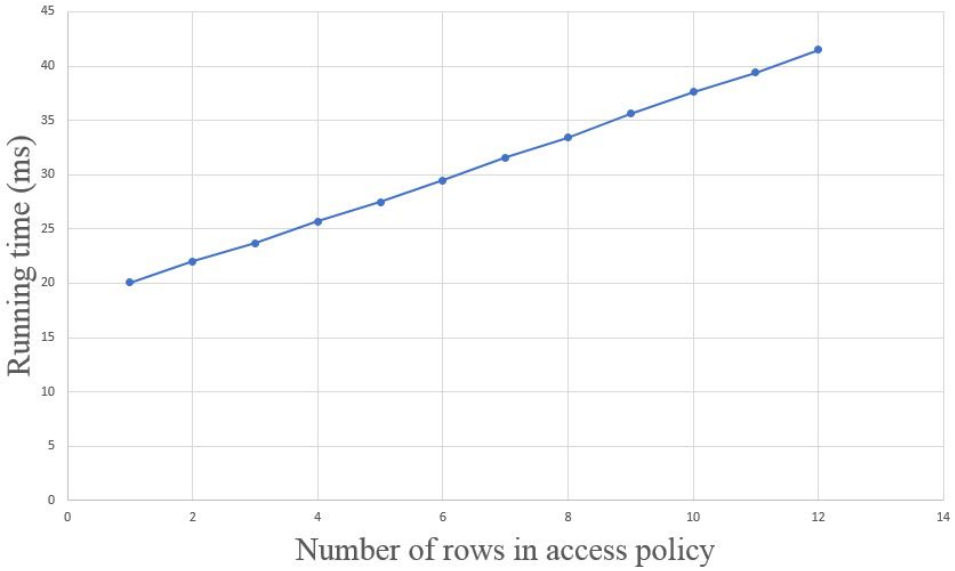

Figure 10: Average running times (in milliseconds) of algorithm Encrypt based on different numbers of attributes in the access policy/of rows in the LSSS matrix.

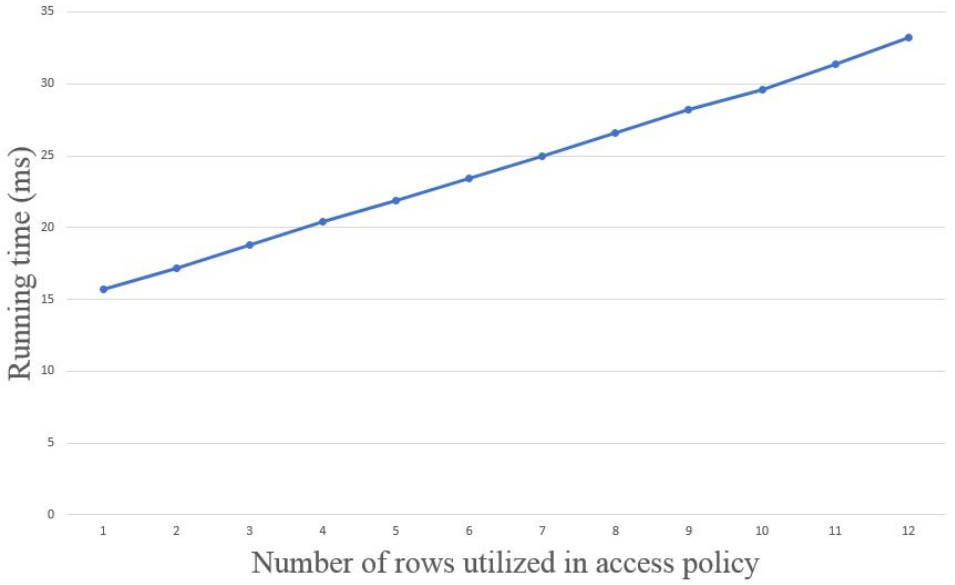

Figure 11: Average running times (in milliseconds) of algorithm Decrypt based on different numbers of attributes in the access policy/of rows in the LSSS matrix that are needed for successful decryption.

Summary Timing for encryption and decryption must stay low since the latter are executed by resource-constrained devices, while timing for system setup and key generation can be significantly higher since they are conducted by powerful entities.

Implementation and evaluation show that our scheme can be realistically deployed in an IoT environment. Indeed, each algorithm is run below 100 milliseconds in most cases.

While setup and key generation algorithms take a longer time, the applicability of our scheme is not impacted since those algorithms are run by powerful entities and only once. On the other side, timings for encryption and decryption reach low results, that is important since their algorithms are supposed to be run frequently by constrained-resource devices.

5 Conclusion

In this paper, we designed an IoT system with access control key updates and direct user revocation, which are beneficial features in IoT. Access control is done using Ciphertext-Policy Attribute-Based Encryption where attributes represent roles of devices within their networks and time validity ranges. We allowed the participation of multiple role authorities to alleviate the key escrow problem. Moreover, we devised a novel approach, based on a binary tree, to append time credentials. This allows us to find an interesting trade-off between key update frequency and user revocation list length, for stressing time-sensitive data exchanged in IoT environments. We adapted the security model to follow the multi-authority setting and proved our scheme secure under the Decisional Bilinear Diffie-Hellman Exponent assumption. The implementation and evaluation results showed that our solution is fully deployable in IoT networks.

References

[1] N. AboDoma, E. Shaaban, and A. Mostafa. Adaptive time-bound access control for internet of things in fog computing architecture. *International Journal of Computers and Applications*, pages 1–12, 2021.

[2] M. Abomhara and G. M. Køien. Security and privacy in the internet of things: Current status and open issues. In *2014 International Conference on Privacy and Security in Mobile Systems (PRISMS)*, pages 1–8, May 2014.

[3] C. Adams and S. Lloyd. *Understanding Public-Key Infrastructure: Concepts, Standards, and Deployment Considerations*. Macmillan Technical Publishing, 1999.

[4] J. A. Akinyele, M. D. Green, and A. D. Rubin. Charm: A framework for rapidly prototyping cryptosystems. Cryptology ePrint Archive, Report 2011/617, 2011.

[5] Z. H. Ali, H. A. Ali, and M. M. Badawy. Article: Internet of things (iot): Definitions, challenges and recent research directions. *International Journal of Computer Applications*, 128(1):37–47, October 2015. Published by Foundation of Computer Science (FCS), NY, USA.

[6] 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*, 36(6):25–35, Nov 2016.

[7] N. Attrapadung, B. Libert, and E. De Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In *Proceedings of the 14th International Conference on Practice and Theory in Public Key Cryptography Conference on Public Key Cryptography*, PKC'11, pages 90–108, Berlin, Heidelberg, 2011. Springer-Verlag.

[8] A. Beimel. *Secure Schemes for Secret Sharing and Key Distribution.* PhD thesis, Israel, 1996.

[9] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, SP '07, pages 321–334, Washington, DC, USA, 2007. IEEE Computer Society.

[10] O. Blazy and C. Chevalier. Spreading alerts quietly: New insights from theory and practice. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*, ARES 2018, pages 30:1–30:6, New York, NY, USA, 2018. ACM.

[11] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In R. Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer Berlin Heidelberg, 2005.

[12] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *Proceedings of the 21th Annual International Conference on Advances in Cryptology*, CRYPTO'01, pages 213–229. Springer Berlin Heidelberg, 2001.

[13] V. Boyko, M. Peinado, and R. Venkatesan. Speeding up discrete log and factoring based schemes via precomputations. In K. Nyberg, editor, *Advances in Cryptology — EUROCRYPT'98*, pages 221–235, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[14] M. Chase. Multi-authority attribute based encryption. In *Proceedings of the 4th Conference on Theory of Cryptography*, TCC'07, pages 515–534, Berlin, Heidelberg, 2007. Springer-Verlag.

[15] N. Dershowitz and E. Reingold. *Calendrical Calculations*. Number 3. Cambridge University Press, 2008.

[16] S. D. Galbraith. Supersingular curves in cryptography. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 495–513. Springer, 2001.

[17] S. D. Galbraith, K. G. Paterson, and N. P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113 – 3121, 2008. Applications of Algebra to Cryptography.

[18] C. Gritti, R. Molva, and M. Önen. Lightweight secure bootstrap and message attestation in the Internet of Things. In *Proceedgins of the 33rd ACM/SIGAPP Symposium On Applied Computing*, SAC'18, 2018.

[19] C. Gritti, M. Önen, R. Molva, W. Susilo, and T. Plantard. Device identification and personal data attestation in networks. *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, 9(4):1–25, 2018.

[20] A. Guillevic. Comparing the pairing efficiency over composite-order and prime-order elliptic curves. In *Proc. of the 11th International Conference on Applied Cryptography and Network Security*, ACNS'13, pages 357–372, Berlin, Heidelberg, 2013. Springer-Verlag.

[21] J. Herranz, F. Laguillaumie, and C. Ráfols. Constant size ciphertexts in threshold attribute-based encryption. In P. Nguyen and D. Pointcheval, editors, *Public Key Cryptography – PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 19–34. Springer Berlin Heidelberg, 2010.

[22] Y. H. Hwang. Iot security & privacy: Threats and challenges. In *Proceedings of the 1st ACM Workshop on IoT Privacy, Trust, and Security*, IoTPTS '15, pages 1–1, New York, NY, USA, 2015. ACM.

[23] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas. Ddos in the iot: Mirai and other botnets. *Computer*, 50(7):80–84, 2017.

[24] D. Kozlov, J. Veijalainen, and Y. Ali. Security and privacy threats in iot architectures. In *Proceedings of the 7th International Conference on Body Area Networks*, BodyNets '12, pages 256–262, ICST, Brussels, Belgium, Belgium, 2012. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[25] J. Li, W. Yao, J. Han, Y. Zhang, and J. Shen. User collusion avoidance cp-abe with efficient attribute revocation for cloud storage. *IEEE Systems Journal*, 12(2):1767–1777, 2018.

[26] J. Li, R. Zhang, Y. Lu, J. Han, Y. Zhang, W. Zhang, and X. Dong. Multiauthority attribute-based encryption for assuring data deletion. *IEEE Systems Journal*, pages 1–10, 2022.

[27] J. K. Liu, T. H. Yuen, P. Zhang, and K. Liang. Time-based direct revocable ciphertext-policy attribute-based encryption with short revocation list. In B. Preneel and F. Vercauteren, editors, *Applied Cryptography and Network Security*, pages 516–534, Cham, 2018. Springer International Publishing.

[28] Q. Liu, G. Wang, and J. Wu. Time-based proxy re-encryption scheme for secure data sharing in a cloud environment. *Information Sciences*, 258:355 – 370, 2014.

[29] Z. Liu, Z. Cao, and D. S. Wong. Efficient generation of linear secret sharing scheme matrices from threshold access trees. 2010.

[30] Z. Liu, F. Wang, K. Chen, and F. Tang. A new user revocable ciphertext-policy attribute-based encryption with ciphertext update. *Secur. Commun. Networks*, 2020:8856592:1–8856592:11, 2020.

[31] R. Longo, C. Marcolla, and M. Sala. Key-policy multi-authority attribute-based encryption. In A. Maletti, editor, *Algebraic Informatics*, pages 152–164, Cham, 2015. Springer International Publishing.

[32] D. Macedo, L. A. Guedes, and I. Silva. A dependability evaluation for internet of things incorporating redundancy aspects. In *Proceedings of the 11th IEEE International Conference on Networking, Sensing and Control*, pages 417–422, April 2014.

[33] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer. A comparative study of lpwan technologies for large-scale iot deployment. *ICT Express*, 5(1):1 – 7, 2019.

[34] A. Miyaji, M. Nakabayashi, and S. Takano. Characterization of elliptic curve traces under FR-reduction. In *International Conference on Information Security and Cryptology*, pages 90–108. Springer, 2000.

[35] N. Oualha and K. T. Nguyen. Lightweight attribute-based encryption for the internet of things. In *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6, Aug 2016.

[36] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow. Iotpot: Analysing the rise of iot compromises. In *Proceedings of the 9th USENIX Conference on Offensive Technologies*, WOOT'15, pages 9–9, Berkeley, CA, USA, 2015. USENIX Association.

[37] M. Patel, J. Shangkuan, and C. Thomas. What's new with the internet of things? Technical report, McKinsey, 5 2017.

[38] C. Pham, Y. Lim, and Y. Tan. Management architecture for heterogeneous iot devices in home network. In *2016 IEEE 5th Global Conference on Consumer Electronics*, pages 1–5, Oct 2016.

[39] Y. Rouselakis and B. Waters. Efficient statically-secure large-universe multi-authority attribute-based encryption. In R. Böhme and T. Okamoto, editors, *Financial Cryptography and Data Security*, pages 315–332, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[40] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'05, pages 457–473, Berlin, Heidelberg, 2005. Springer-Verlag.

[41] P. Schulz, M. Matthe, H. Klessig, M. Simsek, G. Fettweis, J. Ansari, S. A. Ashraf, B. Almeroth, J. Voigt, I. Riedel, A. Puschmann, A. Mitschele-Thiel, M. Muller, T. Elste, and M. Windisch. Latency critical iot applications in 5g: Perspective on the design of radio interface and network architecture. *IEEE Communications Magazine*, 55(2):70–78, February 2017.

[42] A. Sehgal, V. Perelman, S. Kuryla, and J. Schonwalder. Management of resource constrained devices in the internet of things. *IEEE Communications Magazine*, 50(12):144–149, December 2012.

[43] A. Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of the Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO'84, pages 47–53, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[44] B. Waters. Efficient identity-based encryption without random oracles. In *Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'05, pages 114–127, Berlin, Heidelberg, 2005. Springer-Verlag.

[45] B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Proceedings of the 14th International Conference on Practice and Theory in Public Key Cryptography Conference on Public Key Cryptography*, PKC'11, pages 53–70, Berlin, Heidelberg, 2011. Springer-Verlag.

[46] K. Yang and X. Jia. Expressive, efficient, and revocable data access control for multi-authority cloud storage. *IEEE Transactions on Parallel and Distributed Systems*, 25(7):1735–1744, July 2014.

[47] X. Yao, Z. Chen, and Y. Tian. A lightweight attribute-based encryption scheme for the internet of things. *Future Gener. Comput. Syst.*, 49(C):104–112, Aug. 2015.

[48] J. Zhang, T. Li, Q. Jiang, and J. Ma. Enabling efficient traceable and revocable time-based data sharing in smart city. *Eurasip Journal on Wireless Communications and Networking*, 2022, 2022.

[49] Q. Zhang, S. Wang, D. Zhang, J. Wang, and Y. Zhang. Time and attribute based dual access control and data integrity verifiable scheme in cloud computing applications. *IEEE Access*, 7:137594–137607, 2019.

[50] R. Zhang, J. Li, Y. Lu, J. Han, and Y. Zhang. Key escrow-free attribute based encryption with user revocation. *Information Sciences*, 600:59–72, 2022.