Secure and Robust Cyber Security Threat Information Sharing

Anis Bkakria, Reda Yaich, and Walid Arabi

Institut de Recherche Technologique SystemX, 91120–Palaiseau,
{ `Anis.Bkakria, reda.yaich, walid.arabi` }`@irt-systemx.fr`

Abstract. In recent years, several laws have been decreed, at both national and European levels, to mandate private and public organizations to share their Cyber Security related information. However, existing threat sharing platforms implement "classical" access control mechanisms or at most centralized attribute-based encryption (ABE) to prevent data leakage and preserve data confidentiality. These schemes are well-known to be suffering from a single point of failure on security aspects. That is, if the central authority is compromised, the confidentiality of the shared sensitive information is no longer ensured. To address this challenge, we propose a new ABE scheme combining both the advantages of centralized and decentralized ABE while overcoming their weaknesses. It overcomes the centralized ABE's single point of failure on security by requiring the collaboration of several entities for decryption key issuing. In addition, in contrast to existing decentralized ABE schemes, our construction does not require the data providers to fully trust all attributes authorities, only a single authority should be trusted. Finally, we formally prove the security of our ABE construction in the generic group model.

Keywords: Information Sharing · Fine-grained Access Control · Attribute Based Encryption · Cyber Security.

1 Introduction

Several studies and experts have recognised Cyber Security Threat Information Sharing among European operators of critical infrastructures (and essential services) as a mandatory step for continuous improvement of the national security posture [7]. It enhances the pro-activeness of security practitioners through the exchange of actionable information related to network and information security, such as threats, incidents, vulnerabilities, mitigating measures and best practices. Therefore, the European Council resolution 68/01 of 2007 *"encourage, where appropriate in cooperation with The European Network and Information Security Agency (ENISA), effective exchanges of information and cooperation between the relevant organisations and agencies at national level; to commit to fighting spam, spyware and malware"*[21].

More recently, the European legislation such as NIS Directive or the Cyber Security Act advocated and incentivised the creation of sectoral Information

Sharing and Analysis Centers (ISACs) both at national and Europe Levels. ISACs are non-profit structures that aim to provide a federated structure to gather, analyses and share threat information among sectorial communities of private and public stakeholders. However, setting up and running ISACs is facing several technical, financial and legal barriers[1, 14]. Alongside cost-saving and poor management, the lack of trust and potential reputational risks are the most critical barriers to effective information sharing [1].

In the next subsection, we use an illustrative case-study to further motivate the need for a fine-grained and secure threat information sharing mechanism as a mean to mitigate the risk of sensitive information leakage and to incentive critical operators and legal authorities to share valuable information.

1.1 Illustrative Case-study

France was the first European country to have gone through the regulatory process to implement an adequate and mandatory Cyber Security for "Critical Infrastructures Information Protection" (CIIP). As such, a dedicated CIIP regulatory framework was established in 2013 under the name of "CIIP law". As the national authority for Cyber Security and cyber defence, the ANSSI is in charge of coordinating the Cyber Security aspect of the framework and accompanying the critical operators (called "operators of vital importance") in implementing the new measures. In this law, a critical operator is defined as "operator[s] whose unavailability could strongly threaten the economical or military potential, the security or the resilience of the Nation". As part of the CIIP law, critical operators must notify the national authority ANSSI of any cyber incident targeting their critical information systems. The type of incidents to be notified have also been specified by sectorial orders.

However, setting up and running ISACs is facing several technical, financial and legal barriers among which

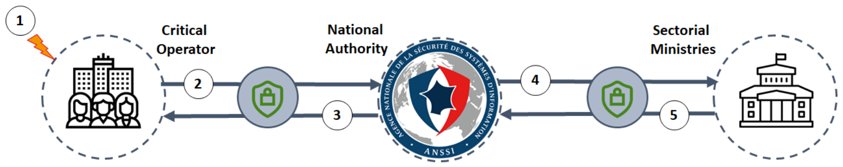

Fig. 1. Illustrative Use-Case (Adapted from[2])

As illustrated in the figure above, the cyber incident handling process is coordinated by the French Authority ANSSI. The process starts when a cyber incident occurs in a critical operator (1). If the incident concerns an Information System of Vital Importance (SIIV), the critical operator sends an incident form to the national authority ANSSI (2). The critical operator will receive ANSSI support that can take the form of remote recommendations or onsite technical assistance to handle the incident (3). The completed incident declaration form is a "confidential" document as it may contain sensitive information that disclosure can lead to criminal prosecution. Thus only secure communication means or post can be used to send the document in order to preserve its integrity and confidentiality. CIIP law imposes ANSSI to preserve incidents information at State level (4-5). However, ANSSI is allowed to use technical information to analyse and anticipate cyber-crisis. These analyse, after being anonymised, can be shared further with other critical operators at the national or European level to strengthen their capacity to detect sophisticated attacks.

Nowadays, the above incident management and threat information sharing process is for the most important part manual. The critical operator need to download a form, complete it and send it by post or secure communication channel. The advent of cyber threat management platforms such as OpenCTI[1] will bring progressively higher degree of automation and accelerate detection and response of new security threats. Figure 2, illustrates how such platforms can be used to automate the process described in the case-study.

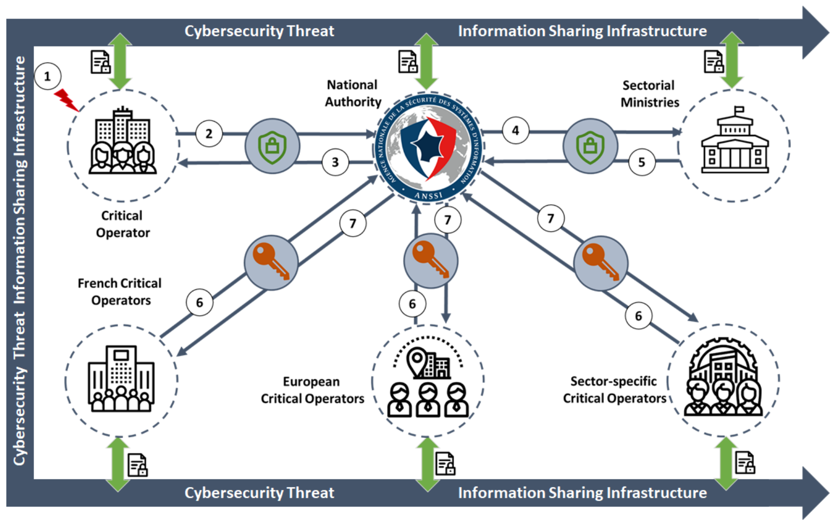
Cybersecurity Threat
Information Sharing Infrastructure
Cybersecurity Threat Information Sharing Infrastructure
National Authority
Sectorial Ministries
Critical Operator
French Critical Operators
European Critical Operators
Sector-specific Critical Operators
ANSSI
1
2
3
4
5
6
7

Fig. 2. Automation of Cyber Threat Information Sharing

However, using such platforms to share confidentials data as presented above requires a *Secure and Robust Cyber Security Threat Information Sharing* mechanism. The next section presents the main contributions of the article to address security and robustness (e.g. Single Point of Failure) challenges raised by the use of such platforms.

[1] https://www.opencti.io/en/

1.2 Our Contributions

In this paper, three main contributions are proposed. First, we propose a secure and robust cyber threat intelligent (CTI) sharing solution. That is, we use a novel ABE based construction to ensure fine grained access control on shared data items. Compared to existing CTI sharing solutions, ours provides higher robustness level as it does not involve any single point failure on security. Second, we propose a novel ABE construction combining both the advantages of centralized and decentralized ABE while overcoming their weaknesses. Our ABE construction overcomes the centralized ABE's single point failure on security by requiring the collaboration of several entities for decryption key issuing. Moreover, our construction does not require the data providers to fully trust all attributes authorities, only a single authority should be trusted by data providers. Finally, we formally prove the security and the robustness of our solution in the generic group model.

1.3 Paper Organization

Our paper has the following structure. We begin in Section 2 by reviewing related work and details the main contributions of our work. In Section 3, we introduce some basic concepts that will be used to build our construction as well as the assumptions under which our schemes achieve provable security. In section 4, we give the overview of the considered system model, ABE scheme definition, threat model, and security requirements. Then, Section 5 details our proposed decentralized ABE scheme. In Section 6, we provide the security results of our proposed construction. We conclude in Section 7. Finally, the appendix reports the formal proofs of the security properties ensured by our ABE construction.

2 Related Work

2.1 Privacy-preserving Cyber Security Information Sharing

In the context of Cyber Security information sharing automation, various protocols and standards have been proposed, such as TAXII, STIX, CybOX, VERIS, MAEC, SCAP and IODEF [9, 19]. Security information sharing in competitive environments with the game theory approach has been studied in [10]. The study of privacy issues in Cyber Security information sharing in [22, 23]. Several information sharing programs, such as CISSP, NCCIC, ISAC, have also been developed in [9].

The recent studies [24, 26] review the current state of the art on cyber threat intelligence (CTI) sharing, identifying associated benefits and barriers. These works highlight that issues of security, trustworthiness, provenance, and privacy remain open research challenges in cyber threat intelligence sharing in that they have not been comprehensively addressed.

Therefore, in this paper, we focus at the most recent and elaborate work on CTI sharing, in order to make it more secure, especially against attackers who target the sharing mechanism itself. One of the most recent Framework using Ciphertext-Policy Attribute Based Encryption (CP-ABE) scheme is [16]. It allows to address several issues related to CTI sharing not resolved by previous works, namely: the confidentiality of personal information, fine-grained access control, reliability, auditability. The authors propose to combine the TATIS security framework, which provides fine-grained protection for the threat intelligence platform API, with the capabilities of the distributed registry to enable trusted and reliable sharing of threat intelligence, with the ability to verify the provenance of the threat intelligence.

Nevertheless, After reviewing the current state of the art in cyber threat intelligence (CTI) sharing, we came to the conclusion that all ABE based CTI sharing approaches use a single authority. In a single-authority-system, all trust rests on the single authority, so if the authority is compromised, the entire system and there is an overhead on the Central Authority (CA) for key management.

Hence, to deal with a single point of failure (SPOF), we propose in our work to use the decentralized systems approach which distribute the responsibility among several entities. Moreover, to address the SPOF flaw, our ABE approach, allows to keep the data contrability, namely, among the chosen attribute authorities that ensure the responbility of sharing, one of them will be identified as a trusted attribute authority (TAA) by the data provider and at each new access to its data it will have the control to generate a key decryption or not. More technical details to address these failures will be given in the following section.

2.2 ABE Access Control

Attribute-based encryption (ABE) was introduced in 2005 by Sahai and Waters [17]. It is a one-to-many public key encryption scheme, i.e. we encrypt with a single key and we have the possibility to generate several keys to decrypt. ABE provides highly granular access control, scalable key management and flexible data distribution [11]. It allows data to be encrypted and shared on the basis of descriptive attributes, without any prior knowledge of the identity of recipients. Only entities with attributes that satisfy a data access policy can decrypt a text. ABE has been widely studied in the literature resulting in many ABE constructions [11]. These constructions can be classified into single-authority [4] and multi-authority [6] ABE. In single-authority setting, the attributes as well as the access key (decryption key) issuing are managed by a single authority, while in the multi-authority setting, the attributes are generated by multiple authorities, yet each authority is responsible of issuing access keys for the data that has been encrypted using its public key. As pointed out in [12], both single and multi-authority ABE constructions suffer from a single-point of failure on security. That is, once an authority is compromised, an adversary can easily obtain the the authority's master key that can be used to generate private keys of any subset of the attributes managed by the compromised authority to access (decrypt) the encrypted data. To deal with the previous security weakness, Li et al [12] proposed a new ABE multi-authority ABE construction called TMACS, where the set of authorities collaboratively manage the whole set of attributes and no one of the authorities has full control of any specific attribute. Thanks to the usage of the (t, n) threshold secret sharing proposed in [15], TMACS is proved to be secured when less that t authorities are compromised by an adversary.

The robustness of the TMACS construction comes at the expense of data access controllability. That is, regardless the access policy that will be enforced by the data owner on the encrypted data, any set of t authorities can issue a decryption key for the encrypted data. Hence, the data owner needs to fully trust all the involved authorities in the system, which is seldom satisfied in real world secure data sharing use cases, including our secure CTI sharing use case.

Compared to existing ABE constructions, the ABE scheme we are proposing in this paper achieves a high level of robustness while providing a better data access controllability to data provider. That is, instead of requiring all attribute authorities to be trusted by all data providers, each data provider needs only to trust a single attribute authority.

3 Preliminary

In this section, we give background information on bilinear maps and the security assumption we are considering. Then, we give a brief description of the trusted third party free secret sharing method proposed by Pedersen in [15].

3.1 Bilinear Maps

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic group of prime order p. Let g be a generator of $\mathbb{G}$ and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map having the following properties:

- Symmetric bilinearity: for all $g_1, g_2 \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(g_1^a, g_2^b) = e(g_1^b, g_2^a) = e(g_1, g_2)^{a \cdot b}$.
- Non-degeneracy: e(g,g) $\neq 1$.
- The group operations in $\mathbb{G}$ and $e(\cdot, \cdot)$ are efficiently computable.

In the sequel, the we refer to the tuple $(\mathbb{G}, \mathbb{G}_T, p, e(\cdot, \cdot))$ as a bilinear environment.

Definition 1 (independence [5]). *Let $P, Q \in \mathbb{F}_p[X_1, \cdots, X_n]$ be two s-tuples of n-variate polynomials over $\mathbb{F}_p$. Write $P = (p_1, \cdots, p_s)$ and $Q = (q_1, \cdots, q_s)$. We say that polynomial $f \in \mathbb{F}_p[X_1, \cdots, X_n]$ is dependant on the sets (P,Q) if there exists $s^2 + s$ constant $\{\vartheta_{i,j}^{(a)}\}_{i,j=1}^{s}$, $\{\vartheta_k^{(b)}\}_{k=1}^{s}$ such that*

$$f = \sum_{i,j} \vartheta_{i,j}^{(a)} \cdot p_i \cdot p_j + \sum_k \vartheta_k^{(b)} \cdot q_k$$

We say that f is independent of (P, Q) if f is not dependent on (P, Q).

Definition 2 (GDHE assumption [5]). *Let $(\mathbb{G}, \mathbb{G}_T, p, e(\cdot, \cdot))$ be a bilinear environment and s,n be positive integers. Let $P, Q \in \mathbb{F}_p[X_1, \cdots, X_n]$ be two s-tuples of n-variate polynomials over $\mathbb{F}_p$ and let $f \in \mathbb{F}_p[X_1, \cdots, X_n]$. Let g be a* *generator of $\mathbb{G}$ and $g_t = e(g, g) \in \mathbb{G}_T$. The GDHE assumption states that, given the vector*

$$H(x_1, \cdots, c_n) = (g^{P(x_1,\cdots,x_n)}, g^{Q(x_1,\cdots,x_n)}) \in \mathbb{G}^s \times \mathbb{G}_T^s$$

it hard to decide whether $U = g_t^{f(x_1,\cdots,x_c)}$ or U is random if f is independent of (P,Q).

3.2 Trusted Third Party Free Threshold Secret Sharing

In a secret sharing scheme, a secret is distributed among several participants organized in an access structure listing all groups that can access the secret. The objective is to provide information specific to each participant so that only a specific group of participants can reconstruct the secret. Several practical secret sharing schemes have been proposed [3, 8, 15, 18]. In this work, we use the trusted third party free threshold secret sharing construction proposed in [15], which we briefly describe as following.

Consider a system involving a set $\mathcal{P} = \{P_1, P_2, \cdots, P_n\}$ of n participants and a threshold t ($t \leq n$). Let us suppose that to each participant $P_i \in \mathcal{P}$ is associated a unique scalar $z_i \in \mathbb{Z}$ ($\forall P_i, \forall P_j \in \mathcal{P} : P_i \neq P_j \Leftrightarrow z_i \neq z_j$) representing the public identifier of the participant in the system. First, each participant P_i selects a random scalar $s_i \in \mathbb{Z}_p$ that will represent his/her sub-secret and generates a random polynomial $f_i(x)$ of degree $t - 1$ such that $f_i(0) = s_i$. The sum of sub-secrets $S = \sum_{i=1}^{n} s_i$ will represented the master secret that will be shared by the participant. Nevertheless, S is not known to any participant. Second, each participant P_i computes the sub-shares $s_{i,j} = f_i(z_j), 1 \leq j \leq n, j \neq i$ and securely sends $s_{i,j}$ to P_j. Once a participant P_i receives sub-shares from all other $n-1$ participants, he/she/it computes $s_{i,i} = f_i(z_i)$ and computes it own master share as $S'_i = \sum_{j=1}^{n} s_{j,i}$. Once each participant P_i has computed his master share S'_i, the master secret key S can be constructed using the Lagrange interpolating formula by any t out of n participants. Let us denote by $S'_k, 1 \leq k \leq t$ the set of master shares to be used, the master secret can be constructed as following.

$$\sum_{k=1}^{t} \left(S'_k \cdot \prod_{j=1, j\neq k}^{t} \frac{z_j}{z_j - z_i} \right) = \sum_{i=1}^{n} S_i = S$$

4 System and Security Models

In this section, we introduce the system model, system definition, threats model and security requirements of our construction.

4.1 System Model

The architecture we consider in our approach involves five entities: A certificate authority, multiple attribute authorities, data providers, and data consumsers (users), and a cloud server.

- The certificate authority (CA) is a blockchain-based PKI management system e.g., [20, 25] that is charged of setting up the system parameters such as the bilinear environment to be used, the set of attributes and their respective public keys. CA is responsible of registering attribute authorities as well as data consumers. Finally its responsible of choosing the robustness level that should be satisfied, i.e., the number of attribute authorities that should collaborate to issue a decryption key. We emphasis that the CA is not involved in any decryption key issuing operation.
- Attribute authorities are mainly responsible of issuing decryption keys to data consumers. In addition, they collaborate together with the CA to set up the master public key of the system.
- Cloud storage server is an entity that provides data storage capabilities.
- The data provider is the entity aiming to share its data. It encrypts the data to be shared using a chosen access structure formulated over a set of attributes that defines who can access the shared data.
- The data consumer (data user) is the entity that will access and use the shared data. He/She/It is labeled by a set of attributes. Data consumers can download any encrypted (shared) data from the cloud service. However, only those who are labeled with proper attributes can successfully decrypt the encrypted data.

4.2 Definition of our Construction

Our construction is defined using seven algorithms that we denote *CASetup, AASetup, CAKeyGen, AAKeyGen, DecKeyGen, Encrypt, Decrypt*. The algorithms *CASetup* and *CAKeyGen* are performed by CA, *AASetup* and *AAKeyGen* are performed by the attribute authorities, *Encrypt* is performed by data providers, finally *DecKeyGen* and *Decrypt* are performed by data consumers.

- **CASetup**(λ) is a probabilistic algorithm that takes as input the security parameter λ and outputs the public parameters of the system pp.
- **AASetup**(pp) is a probabilistic algorithm that takes as input the system parameters pp and returns a secret key share sk and a master public key share pk.
- **CAKeyGen**(pp) is a probabilistic algorithm that takes as input the system parameters pp and outputs a global public master key PMK_{CA}.
- **AAKeyGen**(PMK_{CA}) is a probabilistic algorithm performed by an attribute authority A_i that takes as input the public master key PMK_{CA} and outputs a (local) public master key PK_i.
- **Encrypt**($M, \mathbb{M}, A_i$) is a probabilistic algorithm that takes a message M, and access structure $\mathbb{M}$, and the attribute authority A_i chosen (trusted) by the data provider for validating the access to the data and outputs an encrypted data item bundle χ.
- **DecKeyGen**(PMK_{CA}, $\mathbb{A}$) is a probabilistic algorithm that takes as input the global public master key PMK_{CA} and the set of registered attribute authorities $\mathbb{A}$ and output a secret decryption key $\mathcal{K}$.

- **Decrypt**$(\chi, \mathcal{K})$ is a deterministic algorithm that takes as input an encrypted data item bundle χ and a decryption key $\mathcal{K}$ and outputs the plaintext M if and only if (1) the access structure used to encrypt the the data item is satisfied by the attributes involved in $\mathcal{K}$, and (2) $\mathcal{K}$ is approved and signed by the authority attribute trusted by the data item provider.

4.3 Threat model

In our construction, the CA is assumed to be a trusted entity, that is, he is supposed to issue correct certificates to the different entities of the system. As the CA capabilities are supposed to be provided by a blockchain-based PKI management system e.g., [20, 25], then we fairly assume that the CA is a single point of failure-free entity. The attribute authorities are honest but curious entities. That is, they are supposed to honestly perform the different operations the proposed construction, however some of them may be corrupted by an adversary that aims to learn as much information as possible about the shared data. Moreover, we assume that the cloud server is also honest but curious as it will correctly follow the proposed protocol, yet may collude with malicious data consumers or compromised attribute authorities to get unauthorized access privileges. Finally we assume the data consumers to be malicious entities that can collude with each other, with the cloud server, and/or with compromised attribute authorities.

4.4 Security Requirements

Multiple malicious users may collude to access a data item that none of them can decrypt alone. We require our construction to be secure against such collusion attack. This requirement can be formalized as following.

Definition 3 (Collusion Resistance). *Let λ be the security parameter, $\mathcal{A}$ be the adversary, and $\mathcal{C}$ be the challenger. We consider the following game that we denote $Exp_{\mathcal{A}}^{Col}$.*

1. *Setup: $\mathcal{C}$ executes CASetup, AASetup, CAKeyGen, and AAKeyGen algorithms. It then transmits the public parameters pp, the global public master key PMK_{CA}, and the public master keys PK_{A_i} $(i \in [1, n])$ to $\mathcal{A}$.*
2. *Query – Phase 1: $\mathcal{A}$ can make a set of n adaptive secret decryption key queries. For each query Q_i, it submits the set of attributes Δ_i that should be involved in the decryption key and the attribute authority A_i that validate and sign the secret decryption key. For each query Q_i, $\mathcal{C}$ executes the algorithm DecKeyGen to generate a valid secret decryption key $\mathcal{K}_{A_i}$.*
3. *Challenge: $\mathcal{A}$ chooses two equal-length messages M_0, M_1, the public master key $PK_{A_{i*}}$ of $A_{i*} \in \mathbb{A}$ to be use for message encryption, and a challenge access structure $\mathbb{M}^*$ such that $\forall i \in [1, n], \Delta_i$ does not satisfy $\mathbb{M}^*$. Then it sends them to $\mathcal{C}$. The latter chooses a random $\beta \in \{0, 1\}$, encrypts M_β under $\mathbb{M}^*$ to get the challenge ciphertext $\mathcal{C}^*$, and sends $\mathcal{C}^*$ to $\mathcal{A}$.*

4. *Query – Phase 2: $\mathcal{A}$ can make adaptive secret key queries as in phase 1. The only restriction here is that the set of attributes Δ_i involved in each query does not satisfies $\mathbb{M}^*$, otherwise, $\mathcal{A}$ will trivially win the game by running the Decrypt algorithm.*
5. *Guess: $\mathcal{A}$ outputs its guess β' of β.*

We define $\mathcal{A}$'s advantage by $Adv^{Exp_{\mathbb{A}}^{Col}}(\lambda) = |Pr[\beta = \beta'] - 1/2|$. Our construction is said to be collusion resistant if $Adv^{Exp_{\mathbb{A}}^{Col}}(\lambda)$ is negligible.

In addition, we require our construction to be robust. That is, any data item encrypted by a data consumer u remains fully protected as far as no more than $t-1$ attribute authorities including the one trusted by the data provider are compromised. This requirement is formalized using the following definition.

Definition 4 ((t,n)-Robustness). *Let λ be the security parameter, $\mathcal{A}$ be the adversary, and $\mathcal{C}$ be the challenger. We consider the following game that we denote $Exp_{\mathcal{A}}^{Rob}$. We omit the first four steps of the game since they are the same as defined in $Exp_{\mathbb{A}}^{Col}$.*

5. *Compromise: In this step, $\mathcal{A}$ can perform only one of the following actions:*
   (a) Adaptively chooses $t-1 < n$ attribute authorities including A_{i*} and compromises them to get their master secret key shares $sk_i, i \in [1, t-1]$.
   (b) $\mathcal{A}$ compromise all attribute authorities except A_{i*}.
6. *Guess: $\mathcal{A}$ outputs its guess β' of β.*

We define $\mathcal{A}$'s advantage by $Adv^{Exp_{\mathbb{A}}^{Rob}}(\lambda) = |Pr[\beta = \beta'] - 1/2|$. Our construction is said to be (t,n)-Robust if $Adv^{Exp_{\mathbb{A}}^{Rob}}(\lambda)$ is negligible.

5 Our Proposed Scheme

In this section we give a detailed description of our construction. It is composed of the following four phases: System initialization, user registration and key generation, data encryption, and data decryption.

System Initialization In this phase, the system parameters are set up using the following steps.

1. **CA setup:** This sub-process is performed by CA. The CA first chooses a bilinear environment $(\mathbb{G}, \mathbb{G}_T, p, e(\cdot,\cdot))$ and choose g as a generator of $\mathbb{G}$. Then, CA defines a cryptographic hash functions $H : \mathbb{G}_T \rightarrow \{0,1\}^m$ for some m. Finally, the CA chooses an unforgeable under adaptive chosen message attacks signature system Ξ and generates a signature key *CMK* and a verification key *VMK*. This process outputs the set parameters $pp = (\mathbb{G}, \mathbb{G}_T, p, e(\cdot,\cdot), H, \Xi, VMK)$.
2. **AA registration:** Each attribute authority sends a registration request to CA. If it is a legal authority, the CA generate a random unique identifier $id_{A_i} \in \mathbb{Z}_p$ and generates a signed certificate Υ_{A_i}.

3. **Robustness level selection:** According the number n of the registered authorities, CA chooses the robustness level t ($t < n$) that should be satisfied and publishes its public master key $PMK_{CA} = (pp, n, t)$.
4. **AA setup:** Let us denote $\mathbb{A} = \{A_1, A_2, \cdots, A_n\}$ the set of registered attribute authorities. In this step, the n attribute authorities will collaborate to build a shared secret using the trusted third party free threshold secret sharing described in Section 3.2. Each $A_i \in \mathbb{A}$ selects a secret random $\alpha_i \in \mathbb{Z}_p$ and generates a random $t-1$ degree polynomial $f_i(x)$ such that $f_i(0) = \alpha_i$. Then it calculates the sub-shares $s_{i,j} = f_i(id_{A_j}), \forall i \in [1, n]$ and securely sends the sub-share $s_{i,j}$ to the entity A_j. Once, receiving $n-1$ sub-shares form all other attribute authorities, A_i computes its master secret key share $sk_i = \sum_{i=1}^{n} s_{j,i}$ and its master public key share $pk_i = e(g, g)^{sk_i}$. We emphasis here that the master shared key $\alpha = \sum_{i=1}^{n} \alpha_i$ is decided in the system, but it should not be known to any entity in the system.
5. **Global public key computation:** this step performed by the CA which randomly selects t out of the n master public key shares. Let us denote by $\mathcal{I}$ the set of indices of the t chosen master public key shares. The global public key of the system is then computed as follows:

$$\begin{aligned}\prod_{i \in \mathcal{I}} pk_i^{\prod_{j \in \mathcal{I}, j \neq i} \frac{id_{A_j}}{id_{A_j} - id_{A_i}}} &= \prod_{i \in \mathcal{I}} e(g, g)^{sk_i \cdot \prod_{j \in \mathcal{I}, j \neq i} \frac{id_{A_j}}{id_{A_j} - id_{A_i}}} \\ &= e(g, g)^{\alpha}\end{aligned}$$

   Then, the CA chooses a random master key $a \in \mathbb{Z}_p$ and computes g^a. Moreover, it chooses, for each attribute δ in the universe of attributes to be used Δ, a random public key $o_\delta \in \mathbb{Z}_p$ and computes $\Theta_\delta = g^{o_\delta}$. Then, the CA updates its public master key $PMK_{CA} = (pp, n, t, g^a, e(g, g)^\alpha, \{\Theta_\delta\}_{\delta \in \Delta})$.
6. **AA public key computation**: each $A_i \in \mathbb{A}$ chooses a random $\beta_i \in \mathbb{Z}_p$ and computes its master public key $PK_i = e(g, g)^{\alpha \cdot \beta_i}$. The master secret key of A_i is $SK_i = \{sk_i, \beta_i\}$.

We note here the global master secret $MSK = (a, \alpha)$ does not need to be obtained by any entity of the system. In addition, a does not need to be preserved by CA. It's also worth mentioning that all the information transferred between the different entities are encrypted and signed by their sender.

Data Encryption The data encryption operation Encrypt is performed by the data provider independently. Similarly to most recent ABE schemes, the data to be shared M will be firstly encrypted using a secure symmetric key algorithm such as AES. Then, the chosen symmetric key will be encrypted we described in the following steps.

1. The data provider starts by choosing the attribute authority $A_i \in \mathbb{A}$ he/she trusts for validating the access to the data to be encrypted and shared. Afterwards, he/she defines the access policy that should be enforced as a

monotone boolean formula. Then he/she executes the Encrypt algorithm who picks a random $s \in \mathbb{Z}_p$ and uses the master public key PK_i of the chosen attribute authority A_i to generate the symmetric key as:

$$\kappa = H(PK_i^s) = H\left(e(g,g)^{\alpha\cdot\beta_i\cdot s}\right)$$

Then the Encrypt algorithm encrypts M using κ to get $E_\kappa(M)$.

2. In the second step, the Encrypt algorithm uses the method presented in [13] to transforms the access policy into an LSSS access structure $(\mathbb{M},\rho)$. $\mathbb{M}$ is an $l \times k$ LSSS matrix and $\rho(x)$ maps each row of $\mathbb{M}$ to an attribute $\rho(x)$. Then, to hide the random element s used to generate the symmetric key, the Encrypt algorithm chooses a random vector $\vec{v} = \{s, v_2, \cdots, v_k\} \in \mathbb{Z}_p^k$. For each row vector $\mathbb{M}_i$ of $\mathbb{M}$, $\lambda_i = \mathbb{M}_i \cdot \vec{v}^\top$ is calculated and a random scalar $r_i \in \mathbb{Z}_p$ is chosen. The Encrypt algorithm computes the ciphertext $\mathcal{C}$ as follows:

$$\mathcal{C} = \left(C = g^s, \forall i \in [1,l] : C_i = (g^a)^{\lambda_i} \cdot \Theta_{\rho(i)}^{-r_i}, D_i = g^{r_i}\right)$$

The Encrypt algorithm output the encrypted data $E_\kappa(M)$ and the ciphertext $\mathcal{C}$. Finally the data owner sends the encrypted data item bundle $\chi = (A_i, E_\kappa(M), \mathcal{C})$ to the cloud server for storage.

User Registration and Key Generation When a user u_i joins the system, he/she sends a registration query to the CA to get a unique id_{u_i} and a signed certificate Υ_{u_i}. Then, to get a decryption key, the user has to perform the following two steps.

1. The user u chooses t out of the n attribute authorities according to his/her own preferences and individually queries each of the chosen attribute authorities a decryption key share. The user can generate his secret decryption key if and only if he/she gets t decryption key share from t different attribute authorities. To get a decryption key share from the attribute authority A_i, the user sends a signed query containing its identity id_u and its certificate Υ_u. A_i verifies the signature of the CA on Υ_u then authenticates the request by verifying the signature of the *user* on the request. If the user is legitimate, A_i assigns a set of attributes $\Delta_u^{(i)}$ to the user according to the access that A_i wants to grant to the u. Then A_i chooses a random $z_i \in \mathbb{Z}_p$ and generate a decryption key share as following:

$$\mathcal{K}_i = \{K_i = g^{sk_i} \cdot g^{a\cdot z_i}, \quad L_i = g^{b_i}, \quad \forall\delta \in \Delta_u^{(i)} : K_\delta = \Theta_\delta^{z_i}\}$$

We note here that each attribute authority may assign different set of attributes to the user. In this case, the user will be able only to compute a decryption key that involves the set of attributes $\Delta_u = \cap_{i=1}^{t}\Delta_u^{(i)}$ that has been assigned by all t attribute authorities.

Once u gets t decryption key shares from t different attribute authorities, he/she computes its decryption key as following.

$$\begin{aligned}
K &= \prod_{i=1}^{t} K_i^{\prod_{j=1, i\neq j}^{t} \frac{id_{A_j}}{id_{A_j}-id_{A_i}}} \\
&= \prod_{i=1}^{t} \left(g^{sk_i \cdot \prod_{j=1, i\neq j}^{t} \frac{id_{A_j}}{id_{A_j}-id_{A_i}}} \cdot g^{a \cdot z_i \cdot \prod_{j=1, i\neq j}^{t} \frac{id_{A_j}}{id_{A_j}-id_{A_i}}} \right) \\
&= g^{\alpha} \cdot g^{a \cdot \sum_{i=1}^{t} \left(z_i \cdot \prod_{j=1, i\neq j}^{t} \frac{id_{A_j}}{id_{A_j}-id_{A_i}} \right)} \\
L &= g^{\sum_{i=1}^{t} \left(z_i \cdot \prod_{j=1, i\neq j}^{t} \frac{id_{A_j}}{id_{A_j}-id_{A_i}} \right)} \\
\forall \delta \in \Delta_u : \quad K_{\sigma} &= \Theta_{\delta}^{\sum_{i=1}^{t} \left(z_i \cdot \prod_{j=1, i\neq j}^{t} \frac{id_{A_j}}{id_{A_j}-id_{A_i}} \right)}
\end{aligned}$$

Now, by using $d = \sum_{i=1}^{t} \left(z_i \cdot \prod_{j=1, i\neq j}^{t} \frac{id_{A_j}}{id_{A_j}-id_{A_i}} \right)$, we can simplify the different elements of the user decryption key as follows.

$$\mathcal{K} = \{K = g^{\alpha} \cdot g^{a \cdot d}, L = g^{d}, \forall \delta \in \Delta_u : K_{\delta} = \Theta_{\delta}^{d}\}$$

2. As it is, the decryption key obtained in the previous step does not allow the user to decrypt any encrypted data. To be able to decrypt data items that has been encrypted using A_i public key, the user decryption key has to be approved and signed by A_i. For this, the user chooses a random scalar $q \in \mathbb{Z}_p$ and randomizes its decryption key as follows:

$$\overline{\mathcal{K}} = \{\overline{K} = K^{q}, \overline{L} = L^{q}, \forall \delta \in \Delta_u : \overline{K}_{\delta} = K_{\delta}^{q}\}$$

Then the user sends a signed query containing its randomized decryption key $\overline{\mathcal{K}}$ and its certificate Υ_u to A_i. Once A_i receives the query, it authenticates the request using the user certificate, then, based on the attributes that are included on the key Δ_u, A_i decides whether or not the received key should be validated. If not, the user query is aborted. Otherwise, A_i uses its master secret key SK_i to compute the validated decryption key $\overline{\mathcal{K}}_{A_i}$. Then, it sends the latter to the user.

$$\overline{\mathcal{K}}_{A_i} = \{\overline{K'} = \overline{K}^{\beta_i}, \overline{L'} = \overline{L}^{\beta_i}, \forall \delta \in \Delta_u : \overline{K'}_{\delta} = \overline{K}_{\delta}^{\beta_i}\}$$

Once u receives the signed decryption key $\overline{\mathcal{K}}_{A_i}$, it removes the randomization to compute the final signed (by A_i) decryption key we denote $\mathcal{K}_{A_i}$.

$$\begin{aligned}
\mathcal{K}_{A_i} &= \{K' = \overline{K'}^{q^{-1}}, L' = \overline{L'}^{q^{-1}}, \forall \delta \in \Delta_u : K'_{\delta} = \overline{K'}_{\delta}^{q^{-1}}\} \\
&= \{K' = (g^{\alpha} \cdot g^{a \cdot d})^{\beta_i}, L' = g^{d \cdot \beta_i}, \forall \delta \in \Delta_u : K'_{\delta} = \Theta_{\delta}^{d \cdot \beta_i}\}
\end{aligned}$$

We emphasis here that in the key generation process, no interaction is required between the involved attribute authorities.

Decryption The decryption process is performed by a data consumer (user) u who runs the Decrypt algorithm. The user starts by downloading from the cloud server the data item $(A_i, E_\kappa(M), \mathcal{C})$ that is supposed to be decrypted. To be able to decrypt the data item, the decryption key issued to the data consumer needs to fulfill two requirements: (1) It needs to be approved and signed by A_i, and (2) the attribute sets Δ_u involved in the decryption key satisfies the access structure $(\mathbb{M}, \rho)$ used to encrypt the data item. Let $\mathbb{M}_u$ be a sub-matrix of $\mathbb{M}$, where each row of $\mathbb{M}_u$ corresponds to an attribute in Δ_u, and $\mathbb{I} = \{i : \rho(i) \in \mathbb{A}\}$ be a subset of $\{1, 2, \cdots, l\}$. Let us denote by $\mathbb{M}_i$ the ith row of the matrix $\mathbb{M}$. The Decrypt algorithm computes a set of constants $\{w_i\}_{i\in\mathbb{I}}$ such that $\sum_{i\in\mathbb{I}} w_i \cdot \mathbb{M}_i = (1, 0, \cdots, 0)$. Then, it uses the $\{w_i\}_{i\in\mathbb{I}}$ to decrypt the data item as following.

$$\begin{aligned}
\overline{\mathcal{C}} &= \frac{e(C, K')}{\prod_{i\in\mathbb{I}} \Big(e(C_i, L') \cdot e(D_i, K'_{\rho(i)})\Big)^{w_i}} \\
&= \frac{e(g,g)^{\alpha\cdot\beta_i\cdot s} \cdot e(g,g)^{a\cdot d\cdot\beta_i\cdot s}}{\prod_{i\in\mathbb{I}} \left(e(g,g)^{a\cdot d\cdot\lambda_i\cdot\beta_i} \cdot e(\Theta_{\rho(i)}, g)^{-r_i\cdot d\cdot\beta_i} \cdot e(g, \Theta_{\rho(i)})^{r_i\cdot d\cdot\beta_i}\right)^{w_i}} \\
&= \frac{e(g,g)^{\alpha\cdot\beta_i\cdot s} \cdot e(g,g)^{a\cdot d\cdot\beta_i\cdot s}}{\prod_{i\in\mathbb{I}} \left(e(g,g)^{a\cdot d\cdot\lambda_i\cdot\beta_i}\right)^{w_i}}
\end{aligned}$$

By considering that

$$\begin{aligned}
\sum_{i\in\mathbb{I}} w_i \cdot \lambda_i &= (w_1, w_2, \cdots, w_{|\mathbb{I}|}) \cdot \vec{\lambda}_{\mathbb{I}}^{\top} \\
&= (w_1, w_2, \cdots, w_{|\mathbb{I}|}) \cdot \mathbb{M}_u \cdot \vec{v}^{\top} \\
&= (1, 0, \cdots, 0) \cdot (s, v_2, \cdots, v_n) = s
\end{aligned}$$

we get $\overline{\mathcal{C}} = e(g,g)^{\alpha\cdot\beta_i\cdot s}$. Then, the Decrypt algorithm computes the symmetric key $\kappa = H(\overline{\mathcal{C}})$. Using κ, it decrypts the encrypted data $E_\kappa(M)$ to get M.

6 Security analysis

We now demonstrate the security and robustness of the proposed construction by proving that it fulfills the security requirements defined in Section 4.4.

Theorem 1. *Our construction is collusion resistant under the GDHE assumption.*

Theorem 2. *Our construction is (t,n)-robust under the GDHE assumption.*

7 Conclusion

In this paper, we propose a formally proved secure and robust Cyber Security information sharing solution relying on a novel attribute-based encryption scheme that combines both the advantages of centralized and decentralized ABE while overcoming their weaknesses. In contrast to centralized ABE schemes, our construction is a single point of failure-free on security since it requires the collaboration of several entities for decryption key issuing. In addition, in contrast to existing decentralized ABE schemes, our construction does not require the data providers to fully trust all attributes authorities, only a single authority should be trusted.

Acknowledgments

This research is funded by the European Union's Horizon 2020 research and innovation programme under the Secure Collaborative Intelligent Industrial Automation (SeCoIIA) project, grant agreement No 871967. Additionally, part of this work was done as part of IRT SystemX project PFS (Security of Smart Ports).

References

1. Incentives and Barriers to Information Sharing — ENISA, `https://www.enisa.europa.eu/publications/incentives-and-barriers-to-information-sharing`
2. The French CIIP Framework — Agence nationale de la sécurité des systèmes d'information, `https://www.ssi.gouv.fr/en/cybersecurity-in-france/ciip-in-france/`
3. Bertilsson, M., Ingemarsson, I.: A construction of practical secret sharing schemes using linear block codes. In: International Workshop on the Theory and Application of Cryptographic Techniques. pp. 67–79. Springer (1992)
4. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE symposium on security and privacy (SP'07). pp. 321–334. IEEE (2007)
5. Boneh, D., Boyen, X., Goh, E.J.: Hierarchical identity based encryption with constant size ciphertext. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 440–456. Springer (2005)
6. Chase, M.: Multi-authority attribute based encryption. In: Theory of cryptography conference. pp. 515–534. Springer (2007)
7. ENISA: ENISA NCSS Good Practice Guide. No. November (2016), `www.enisa.europa.eu.`
8. Ito, M., Saito, A., Nishizeki, T.: Secret sharing scheme realizing general access structure. Electronics and Communications in Japan (Part III: Fundamental Electronic Science) **72**(9), 56–64 (1989)
9. Kampanakis, P.: Security automation and threat information-sharing options. IEEE Security Privacy **12**(5), 42–51 (2014). https://doi.org/10.1109/MSP.2014.99
10. Khouzani, M., Pham, V., Cid, C.: Strategic discovery and sharing of vulnerabilities in competitive environments. In: International Conference on Decision and Game Theory for Security. pp. 59–78. Springer (2014)
11. Kumar, P., Alphonse, P., et al.: Attribute based encryption in cloud computing: A survey, gap analysis, and future directions. Journal of Network and Computer Applications **108**, 37–52 (2018)

12. Li, W., Xue, K., Xue, Y., Hong, J.: Tmacs: A robust and verifiable threshold multi-authority access control system in public cloud storage. IEEE Transactions on parallel and distributed systems **27**(5), 1484–1496 (2015)
13. Liu, Z., Cao, Z.: On efficiently transferring the linear secret-sharing scheme matrix in ciphertext-policy attribute-based encryption. IACR Cryptol. ePrint Arch. **2010**, 374 (2010)
14. Nweke, L.O., Wolthusen, S.: Legal Issues Related to Cyber Threat Information Sharing among Private Entities for Critical Infrastructure Protection. International Conference on Cyber Conflict, CYCON **2020-May**, 63–78 (2020). https://doi.org/10.23919/CyCon49761.2020.9131721
15. Pedersen, T.P.: A threshold cryptosystem without a trusted party. In: Workshop on the Theory and Application of of Cryptographic Techniques. pp. 522–526. Springer (1991)
16. Preuveneers, D., Joosen, W., Bernal Bernabe, J., Skarmeta, A.: Distributed security framework for reliable threat intelligence sharing. Security and Communication Networks **2020**, 1–15 (08 2020). https://doi.org/10.1155/2020/8833765
17. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 457–473. Springer (2005)
18. Shamir, A.: How to share a secret. Communications of the ACM **22**(11), 612–613 (1979)
19. Steinberger, J., Sperotto, A., Golling, M., Baier, H.: How to exchange security events? overview and evaluation of formats and protocols. In: 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM). pp. 261–269. IEEE (2015)
20. Talamo, M., Arcieri, F., Dimitri, A., Schunck, C.H.: A blockchain based pki validation system based on rare events management. Future Internet **12**(2), 40 (2020)
21. UNION, T.C.O.T.E.: Strategy for a Secure Information Society in Europe (2007/C 68/01), `https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32007G0324(01)`
22. Vakilinia, I., Tosh, D.K., Sengupta, S.: 3-way game model for privacy-preserving cybersecurity information exchange framework. In: MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM). pp. 829–834 (2017). https://doi.org/10.1109/MILCOM.2017.8170842
23. Vakilinia, I., Tosh, D.K., Sengupta, S.: Attribute based sharing in cybersecurity information exchange framework. In: 2017 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS). pp. 1–6 (2017). https://doi.org/10.23919/SPECTS.2017.8046770
24. Wagner, T.D., Mahbub, K., Palomar, E., Abdallah, A.E.: Cyber threat intelligence sharing: Survey and research directions. Computers & Security **87**, 101589 (2019). https://doi.org/https://doi.org/10.1016/j.cose.2019.101589, `https://www.sciencedirect.com/science/article/pii/S016740481830467X`
25. Yakubov, A., Shbair, W., Wallbom, A., Sanda, D., et al.: A blockchain-based pki management framework. In: The First IEEE/IFIP International Workshop on Managing and Managed by Blockchain (Man2Block) colocated with IEEE/IFIP NOMS 2018, Tapei, Tawain 23-27 April 2018 (2018)
26. Zibak, A., Simpson, A.: Cyber threat information sharing: Perceived benefits and barriers. In: Proceedings of the 14th international conference on availability, reliability and security. pp. 1–9 (2019)