smart cities

MDPI

Article

Trustworthy Users: Using IOTA and IPFS for Attribute Validation in CP-ABE and dCP-ABE Schemes

Aintzane Mosteiro-Sanchez [1,2,*], Marc Barcelo [1], Jasone Astorga [2] and Aitor Urbieta [1]

[1] Ikerlan Technology Research Centre, 20500 Arrasate-Mondragón, Spain
[2] Department of Communication Engineering, University of the Basque Country (UPV/EHU), 48013 Bilbao, Spain
* Correspondence: amosteiro@ikerlan.es

Abstract: Attribute spoofing is a major security threat in information exchange solutions based on Ciphertext-Policy Attribute-Based-Encryption (CP-ABE) and distributed CP-ABE (dCP-ABE), which can compromise privacy and security. This threat occurs when an attacker forces the Attribute Authorities to generate keys for attributes they do not possess. This paper analyzes the threat of attribute spoofing and identifies the primary attack vectors, including direct interference with the Attribute Authority and compromise of the shared attribute storage database. The authors propose a solution based on IOTA, a DAG-type DLT, and Interplanetary File System (IPFS) to prevent attribute spoofing. The solution requires distributed attribute storage, validation, and user authentication to counteract the two attack vectors effectively. The proposed solution mitigates the consequences of attribute spoofing, including privilege escalation and reduction, acquisition of private keys, and cutoff of data access. The authors also evaluate their proposal through a value-chain use case and conclude that it effectively mitigates the consequences of attribute spoofing.

Keywords: CP-ABE; dCP-ABE; IOTA; IPFS; FIM; value chain; Industry 4.0

Citation: Mosteiro-Sanchez, A.; Barcelo, M.; Astorga, J.; Urbieta, A. Trustworthy Users: Using IOTA and IPFS for Attribute Validation in CP-ABE and dCP-ABE Schemes. *Smart Cities* **2023**, *6*, 913–928. https://doi.org/10.3390/smartcities6020044

Academic Editorss: Miguel Pincheira and Massimo Vecchio

Received: 30 January 2023
Revised: 5 March 2023
Accepted: 6 March 2023
Published: 10 March 2023

1. Introduction

Traditional encryption schemes guarantee data confidentiality between two endpoints. However, their performance is reduced when the same information is intended for multiple users. In 2005, Sahai and Waters proposed Attribute-Based Encryption (ABE) [1] to address this limitation. ABE schemes correlate encryption and decryption with access policies and attributes. Thus, the same ciphertext can be decrypted by multiple users if the attributes and the policy match. ABE continued to be developed in the cryptographic field, giving rise to many different modes. However, it can be considered that all of them may be grouped under three different modes: Key-Policy Attribute-Based Encryption (KP-ABE) [2], Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [3] and a decentralized version called Decentralized CP-ABE (dCP-ABE) [4]. The main difference between the three modes is how they correlate encryption and decryption with the aforementioned access policies.

In KP-ABE, access policies are used to create users' private keys, while ciphertexts are generated based on attributes. Meanwhile, in CP-ABE, users have keys generated according to attributes, and the ciphertext is generated according to access policies. This distinct difference between CP-ABE and KP-ABE provides data owners using CP-ABE with more flexibility and control to define who can access their data. This flexibility makes CP-ABE the most widely used ABE mode. However, it also creates a significant point of failure in CP-ABE: the authority. The entire key generation depends on a single authority, which puts the encryption system at risk if it is compromised. Instead of relying on a single Key Generation Center (known in ABE schemes as Attribute Authorities, AAs), dCP-ABE generates each private key combining multiple Attribute Authorities. This makes it a suitable scheme for distributed environments.

Thus, it can be seen that since CP-ABE and dCP-ABE allow data owners to retain control over who can access the information, distributed environments may benefit from their use over KP-ABE. This is especially relevant in Industry 4.0, in which combining CP-ABE and dCP-ABE with symmetric encryption like AES Galois/Counter Mode (AES-GCM) can provide End-to-End (E2E) confidentiality and integrity.

Despite these schemes' advantages, in CP-ABE and dCP-ABE, to exploit their flexibility to the fullest, the secure generation of private keys according to attributes is crucial. For this purpose, the Attribute Authorities must know which attributes belong to which users. Some solutions [5] consider that the attribute authorities should handle and distribute the attributes. However, this solution implies that the generators know the attributes of each user in the system, which reduces the scalability of the solution. Furthermore, since attributes are used to define the roles or privileges of users (e.x., what company they belong to, their role, clearances, or department they belong to), it can also create trust issues between collaborating and competing companies. Overloading Attribute Authorities with managing and maintaining that much information is unfeasible. In this context, we identify a new risk: attribute spoofing. Spoofing is caused by users who take advantage of the low scalability of the system to demand keys from Attribute Authorities that reflect privileges they do not possess.

Therefore, attribute spoofing is a security risk that must be considered to design a comprehensive security system to ensure E2E data confidentiality between partners in real-world scenarios. To this end, it is necessary to have a system that allows distributed attribute management, freeing the attribute authority from storing and managing this information internally. Solutions like Distributed Ledger Technologiess (DLTs) can provide a secure attribute management system that prevents attribute impersonation. They also guarantee integrity, immutability, and auditability, which builds trust among all chain members. In addition, they relieve the Attribute Authorities from storing and managing this information internally.

PROBLEM STATEMENT. Our previous work [6] presented an attribute spoofing prevention system for dCP-ABE. However, it was limited in its definition and analysis of attribute spoofing and its attack vectors.

OUR CONTRIBUTIONS. This paper goes a step further and extends our previous work with the following contributions.

- We extend the attribute spoofing prevention system to CP-ABE and dCP-ABE.
- We define attribute spoofing attack vectors, establish the system assumptions under which they take place in value chains, and the consequences for the chain if attackers succeed.
- We present experimental results for the proposal.

The rest of the paper is organized as follows: Section 2 presents the related work and background, Section 3 defines attribute spoofing, outlines the assumptions under which it takes place, and defines the attack vectors and the potential solution requirements. Our proposal for attribute spoofing prevention is presented in Section 4, based on the requirements defined in Section 3. In Section 5, the system is evaluated through qualitative and experimental methods. Finally, the paper concludes in Section 6.

2. Related Work & Background

This section discusses previous work related to the issue of attribute spoofing in the context of CP-ABE and dCP-ABE and provides background information on key generation in these schemes.

2.1. Related Work

Cryptographic schemes that support one-to-many encryption, such as those based on ABE and dCP-ABE, are particularly useful in settings where large amounts of information need to be shared with many recipients [7]. While other solutions have been proposed that also enable one-to-many information sharing [8], they typically require knowledge

of the identity of each recipient. In contrast, ABE and dCP-ABE schemes do not have this limitation, as they use attributes to generate private keys. However, the effective use of these schemes requires a robust attribute management system, which can be a complex problem in distributed systems such as value chains.

The literature review has demonstrated that CP-ABE and dCP-ABE are viable solutions for protecting data sharing and achieving secure data distribution in value chains [9] by limiting decryption to those users fulfilling an access policy, e.g., (Engineer AND CompanyA). To address the challenge of attribute distribution, some approaches [5] propose that attributes are managed and distributed by an Attribute Authority. Such solutions also consider that the authority always generates a private key that accurately reflects the users' privileges. However, this approach can lead to performance bottlenecks [10] if the authority's computational capabilities are limited, generating trust issues between different partners in the value chain and a single point of failure by centralizing all the management on a single node (in the case of CP-ABE). Other authors retrieve attributes from LDAP services, enterprise databases, or SAML Attribute Authorities [11]. However, they do not detail how this integration occurs or whether a consensus is established about the attribute format. For example, retrieving attributes from the companies' LDAP services in international value chains can lead to the same privilege (e.g., "researcher") being defined in different languages, resulting in different attributes. If the attributes and policies do not match, decryption will not occur. In another paper [12], authors still consider that attributes can be retrieved from third parties. However, the risk of attribute spoofing still exists.

Therefore, it is common in the literature to assume that somehow Attribute Authorities already know users' attributes and that they are always correct and accurate. In fact, it is an assumption that recent works like [13] or [14] continue to hold. However, attackers requesting private keys from authorities that allow them to access the information they should not have access to is a security risk that must be considered to design a holistic security system that ensures E2E data confidentiality between partners in real-world scenarios.

Distributed Hash Table (DHT)-based infrastructures are effective for the distribution of private keys in CP-ABE and dCP-ABE systems [15]. The referenced paper combines a DHT infrastructure with a Secret Sharing Scheme (SSS) to distribute the private keys. Therefore, DHT-based distributed storage solutions have significant potential as part of a solution to prevent attribute spoofing in CP-ABE and dCP-ABE systems. One of the most prominent distributed systems based on DHTs is Interplanetary File System (IPFS). Alternatively, other data-sharing solutions use distributed ledger technologies DLTs instead of DHTs to share industrial data, and it has even been studied which DLTs are most suitable for use in industrial environments [16]. Therefore, these technologies are a potential solution to the problem of attribute management in CP-ABE and dCP-ABE systems and can help prevent attribute spoofing.

2.2. CP-ABE Setup & Key Generation

This section provides an overview of the functions used to set up and generate private keys for users in a CP-ABE scheme. Detailed explanations of the mathematical operations involved in these functions are beyond the scope of this paper, as they vary depending on the specific CP-ABE scheme being used. However, it is sufficient to understand these functions' input requirements and output to deploy an attribute spoofing prevention system.

The first function to be run when setting up a CP-ABE-based data encryption system is *Setup*. This function generates the Master Public Key (*MPK*) to be used in the system and the Master Private Key (*MSK*) required by the Attribute Authority to generate private keys.

Setup: Using a non-zero random value r, it generates the *MSK* and the *MPK*.

$$Setup(r) \rightarrow (MSK,\ MPK) \tag{1}$$

Once the *MSK* and *MPK* are created, users can request private keys to Attribute Authorities. The authorities will generate the private keys using the following function:

User Private Key Generation: It uses the users' attribute set $\mathbb{A}$, the *MSK* and the *MPK* to generate the private key SK_{CP-ABE}. The key generation is randomized, so private keys generated with similar $\mathbb{A}s$ are different and cannot be combined, preventing key collusion.

$$KeyGen(\mathbb{A},\ MPK,\ MSK) \rightarrow SK_{CP-ABE} \tag{2}$$

2.3. dCP-ABE Setup & Key Generation

This section presents the functions for setting up and generating private keys for users in a dCP-ABE scheme. These functions resemble those used in CP-ABE but include the required coordination between different authorities during key generation. As in CP-ABE, dCP-ABE must also prevent key collusion. However, in a decentralized setting, the prevention of collusion cannot be solely achieved through randomness and requires the implementation of an additional measure.

Setup: It takes the security parameter k as input and outputs the system's global parameters GP .

$$Setup(K^k) \rightarrow GP \tag{3}$$

Attribute Authorities Setup: Every Attribute Authority runs this algorithm. Each Attribute Authority manages an attribute subset $\mathbb{D}_i = \{att_1, att_2, \cdots, att_n\} \in AA$ where $1 \leq i \leq m$; such that $n \geq 1$ and m is the total number of attributes in the system. Thus, the Attribute Authorities take their identity AID and the GP as inputs. Next, they use these parameters to generate the public key PK_{AID} and the private key SK_{AID} related to $\mathbb{D}_i$. Authorities can be set at any time after GP generation.

$$AASetup(GP, AID) \rightarrow PK_{AID},\ SK_{AID} \tag{4}$$

User Private Key Generation: Users are defined according to their attribute subset $\mathbb{A}$. However, in dCP-ABE it is defined as $\mathbb{A} = \mathbb{D}'_1 \cup \mathbb{D}'_2 \cup \cdots \cup \mathbb{D}'_p$ in which $1 \leq p \leq m$. Users also provide their unique identifier UID to the Attribute Authorities that fulfill $\mathbb{D}'_p \subset \mathbb{D}_i$. In return, the Attribute Authorities return $SK_{UID,\ \mathbb{D}'_p}$. By combining those, users obtain the private key $SK_{UID,\ \mathbb{A}}$. By tying every piece of the user private key to their identifier UID, dCP-ABE prevents collusion resistance [4].

$$KeyGen(UID,\ AID,\ \mathbb{A},\ SK_{AID},\ GP) \rightarrow SK_{UID,\mathbb{A}} \tag{5}$$

3. Attribute Spoofing Definition & System Requirements

In this section, we first define attribute spoofing and identify the main attack vectors for value chains. Based on this analysis, we then outline the requirements that an attribute spoofing prevention system should fulfill to mitigate these attacks effectively in the identified use case.

Using value chains as use case allows us to identify the specific threats that must be addressed and adequately determine the requirements for an attribute spoofing prevention system.

3.1. Attribute Spoofing Definition

Information exchange solutions based on CP-ABE and dCP-ABE usually overlook how the Attribute Authorities determine the users' attributes. The main drawback of this approach is that it does not consider the risk of attribute spoofing. Attribute spoofing forces authorities to generate keys based on attributes users do not possess. This attack can be based on escalation, i.e., users request a key with higher privileges than they have, or on pure forgery: attackers from outside the system force the authority to grant them attributes they do not possess. The attribute spoofing identified in this paper is based on the following assumptions about the value chain use case.

- Assumption 1: Attribute Authorities do not know the defined attribute universe $\mathbb{U}$.
- Assumption 2: Attribute Authorities do not know users' attribute sets, $\mathbb{A}$.

- Assumption 3: Every legitimate user belongs to a participating company in a value chain.
- Assumption 4: Every participating company knows which users depend on them and what attributes they have.

The following subsections define the two attack vectors identified as capable of taking advantage of the established assumptions. The attack vectors are defined considering that the main attack points for attackers are either the Attribute Authority or the shared attribute storage database. Companies are also considered to have implemented security measures that prevents them from being attacked.

3.1.1. Attack Vector 1: Directly Interfering with the Attribute Authority

We denote this attack vector as **AV 1**. In CP-ABE, key generation is randomized to ensure that users with the same set of attributes e.g., $\mathbb{A} = (Researcher\ AND\ Company_A)$ obtain different CP-ABE private keys (SK_{ABE}). As a result, colluding keys and combining them to create a new SK_{ABE} with higher privileges is not possible. Similarly, in dCP-ABE, private keys are tied to the user's *UIDs*, so combining private key pieces from different users is also ineffective.

Instead, if a malicious user or an attacker wants to obtain a SK_{ABE} according to attributes they do not posses (e.g., claiming to have $\mathbb{A}' = (Researcher\ AND\ Company_B)$ instead of the designed $\mathbb{A}$), they can try to obtain it from the Attribute Authority. Two ways of exploiting this vector have been identified: by a malicious user and by an external attacker.

In the case of the malicious user, we assume they have the legitimate attribute set $\mathbb{A}$. However, when interacting with the Attribute Authorities, they request a key for the attribute set $\mathbb{A}'$, which contains different attributes than $\mathbb{A}$. Therefore, authorities deliver a private key that entitles the user to access data they should not be able to access. The success of this attack is based on the following:

- It is a legitimate user, so it passes the authentication process with the authorities.
- Based on Assumption 2, authorities do not know what attributes the user has and therefore do not distinguish between $\mathbb{A}$ and $\mathbb{A}'$.

The case of the external attacker is based on credential theft. If an external attacker has been able to steal a legitimate system user's credentials, they can interact with the authorities and bypass the authentication process. The attacker can then request keys for any possible attribute set $\mathbb{A}$ or users' *UID*. The success of this attack lies in the following:

- Based on Assumption 2, the authorities do not know what attributes the original user to whom the credentials belong has and, therefore, will generate any key requested by the attacker.

Finally, it is necessary to consider that the generation of keys in CP-ABE is randomized, so examining them is insufficient to detect overprivileged keys.

3.1.2. Attack Vector 2: Interfering with the Attribute Storage

We denote this attack as AV 2. As stated by Assumption 3, every user belongs to one of the companies in the value chain. Furthermore, according to Assumption 4, companies know which attributes their users hold. Therefore, a straightforward solution would be for companies to store attributes in a shared centralized database so Attribute Authorities can retrieve them. However, attackers can interfere with attribute storage by compromising the database and modifying the information related to users' attributes, as Figure 1 shows. Thus, additional security mechanisms allowing Attribute Authorities to retrieve the attributes reliably, ensuring their integrity and confidentiality, are needed.

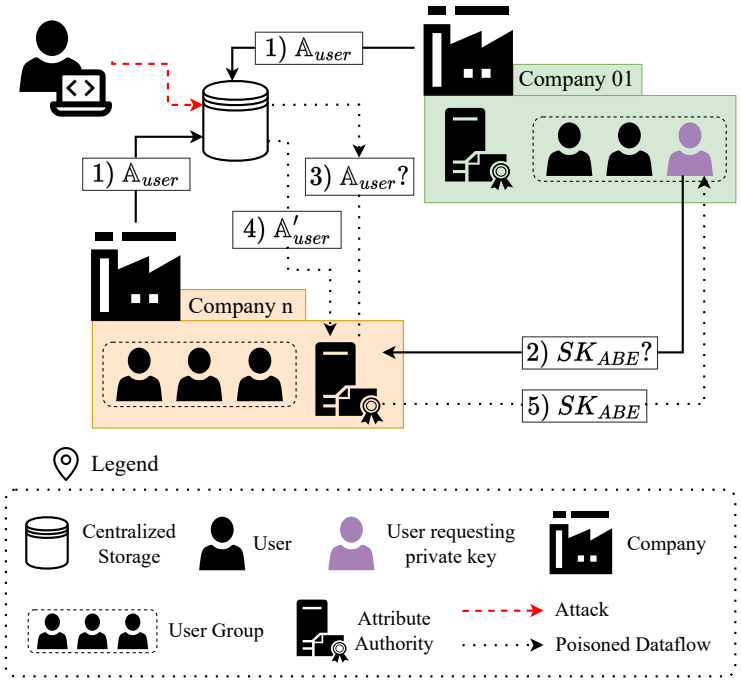

Figure 1. AV 2—Interfering with the Attribute Storage. The attacker poisons the database, resulting in the user getting SK'_{ABE} instead of SK_{ABE}.

This attack can create a significant disruption. The Attribute Authorities would not know that they are generating incorrect keys, and the users and their companies would not be aware that users' private keys do not reflect their privileges. In addition, since analyzing the private keys does not provide information on the attributes contained in them, it is not straightforward to detect which private keys have been altered and which have not. The success of this attack is based on the following:

- According to Assumption 1, Attribute Authorities do not know the attribute universe $\mathbb{U}$, so modification on existing attributes would go unnoticed.
- According to Assumption 2, authoritoes do not know users' attributes.

Therefore, a system that guarantees that Attribute Authorities generate correct keys to legitimate users and that fake users cannot obtain a SK_{ABE} is needed. Furthermore, attributes must be auditable to detect when false information has been stored and by whom.

3.2. Attribute Spoofing Solution Requirements

Once the attack has been defined, and the vectors capable of exploiting it in value chains have been identified, the requirements the solution to prevent attribute spoofing in value chains has to fulfill can be established.

- R1. The solution shall provide distributed attribute storage: In order for the Authorities to always have the attributes available, the attribute storage system must not have one-point-failures. Therefore, it will benefit from distributed storage solutions.
- R2. The solution shall validate users' attributes. Although not a native feature of CP-ABE, validating user attributes protects the system and builds trust in data exchange. For this purpose, attribute validation cannot rely solely on Attribute Authorities, as it may cause a bottleneck [10]. In addition, an efficient and scalable authentication system must ensure the Authorities receive the correct *UID* in dCP-ABE.
- R3. The solution shall provide reliable and auditable attribute management: The distributed attribute storage must be accompanied by a system that enables its auditability. This way, the integrity of the attributes is protected, and the nodes that store them are guaranteed not to make unauthorized modifications.

- R4. The solution shall be suitable for Industrial IoT (IIoT) devices: The attribute validation system must be deployable on all kinds of devices, regardless of their computational capabilities.

4. Attribute-Spoofing Prevention System Definition

This section presents the proposed solution to prevent attribute spoofing. Table 1 summarizes the assumptions on which the attack vectors identified in the previous section are based and the consequences for value chains.

Table 1. Attack vector, assumption, and solution summary.

Attack Vector	Assumption 1 2	Consequence 1 2 3 4	Requirement 1 2 3
	1. Assumption 1 2. Assumption 2	1. Privilege Scalation 2. Privilege Reduction 3. SK_{ABE} Adquisiton 4. Cut-off Data Access	1. R1 2. R2 3. R3
AV 1	□ ■	■ □ ■ □	■ ■ □
AV 2	■ ■	■ ■ ■ ■	□ ■ ■

As seen in Table 1 and as explained in the previous section, the success of AV 1 is based on Assumption 2. The mitigation of this attack requires compliance with requirements R1 and R2. R1 imposes distributed attribute storage, which allows authorities to retrieve this information from a database instead of relying on users' messages. R2 calls for attribute validation and user authentication. Attribute validation guarantees that the attributes come from a trusted source, generating confidence for the authorities. User authentication guarantees that only legitimate users connect to the system.

Regarding AV 2, it is based on Assumption 1 and Assumption 2 and it has more consequences than AV 1. As before, malicious users can escalate privileges by claiming more privileges than they have, and external attackers can force the system to grant them private keys. This time, however, privilege reductions and cutting off access to data stand out. These actions would be carried out by an attacker, who seeks to harm certain users by reducing their privileges or taking them away altogether. This attack vector can be mitigated by complying with R2 and R3. If attributes are validated, authorities can detect potential spoofings. Regarding auditability, it provides a record of attribute modification, enabling it to detect integrity violations, trace the changes, identify the attackers, and discover the affected private keys.

4.1. IPFS for Attribute Distribution

R1 requires a distributed storage of attributes. The purpose of this distribution is twofold: to relieve the attribute authority from attribute storage and management and to make the information accessible at any time without single points of failure. As Section 2.1 introduced, IPFS is a peer-to-peer protocol that offers content discovery through DHTs and can be used to establish a high-performance distributed storage model. IPFS solutions have no single point of failure, nodes do not need to trust each other, and every distributed file has a timestamp [17]. Another advantage of IPFS is that there is no central server; instead, data is distributed and stored in separate locations. These properties have made IPFS one of the most supported solutions for distributed storage of industrial information [8] since it does not cause bottlenecks [18] and can be used by IIoT devices [19]. Because of this, the distributed attributes storage for the attribute spoofing prevention solution is built with IPFS.

The proposed solution is based on Assumption 4, in which companies manage their users' attributes. To do so, the companies agree on an attribute universe called $\mathbb{U}$. Afterward, each company defines a set of attributes $\mathbb{A}$ for each of their users, such that $\mathbb{A}_{user} \subset \mathbb{U}$. With the different $\mathbb{A}_{user}$ defined, companies store them in a private IPFS, as Figure 2 shows. To generate a private IPFS network, companies have to define their bootstrap and client nodes, as well as establish a swarm key that guarantees the privacy of the network. The specifics of deploying a private IPFS network are considered out of scope in this paper. The process to store $\mathbb{A}_{users}$ in IPFS is detailed below:

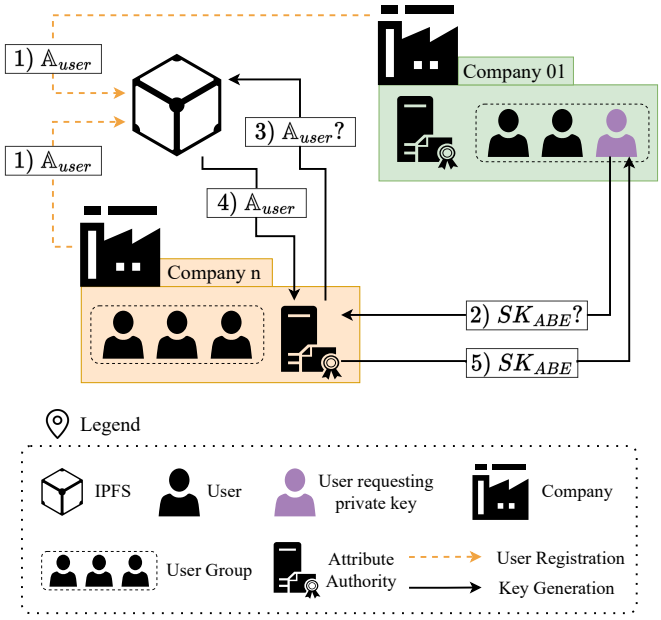

Figure 2. CP-ABE and dCP-ABE attribute storage in IPFS.

1. During a preliminary phase, companies register their users by storing their users' set $\mathbb{A}$ in IPFS.
2. When users need a private key, they request SK_{ABE} from the Attribute Authorities.
3. Attribute Authorities request users' attribute set from IPFS.
4. Attribute Authorities take $\mathbb{A}$ as input to generate users' SK_{ABE}.
5. Users get their SK_{ABE}.

Once the information has been stored in IPFS, it can be retrieved by any node that connects to the network, as long as the node knows the Content Identifier (CID) of the file it needs to retrieve. CIDs are generated from the hash of the content, which prevents data duplication and allows for data integrity verifications.

The content of the IPFS is retrieved by the Attribute Authorities that need it to generate the SK_{ABE}. The CID allows Authorities to detect if the file information in IPFS has been modified. In addition, IPFS relieves the Authorities from managing the attributes and transfers that responsibility to the companies.

4.2. IOTA for Attribute Auditability

R3 requires attribute auditability. Attribute storage in IPFS provides distributed storage, and the hash from which the CIDs are generated allows the Attribute Authorities to detect potential manipulations. However, while IPFS can detect modified content, it cannot track which user has modified it. In this regard, DLTs can provide the auditability and immutability that the system requires to prevent attribute spoofing.

DLTs are composed of nodes that contain distributed, replicated, and synchronized data. Nodes forego a central authority and instead agree on the ledger's state using a consensus protocol. Hence, DLTs are resilient and provide traceability to the attribute validation system [20]. It should be noted that, reading this description, the straightforward solution

would be to directly use a DLTs for distributed storage. However, storing credentials directly on a DLT generates several transactions [13], requiring a high-capacity network.

There are many different DLT systems, and choosing one whose performance and efficiency suit the industrial environment is crucial. The usefulness of DLTs in industry has already been proven, and its effect is considered positive in improving data confidentiality, privacy, and security in IIoT networks [21]. In this sense, the literature considers that Directed Acyclic Graph (DAG)-type DLTs are the most promising for the industry due to their scalability and transaction speed [22,23]. In fact, DAG-type DLTs are faster and more secure as transactions increase, which provides a high performance [16]. Currently, the main DAG-based DLT is IOTA (https://www.iota.org/, accessed on 1 December 2022) and it is considered one of the most-promising DAG-based DLTs [24]. IOTA is specially designed for Internet of Things (IoT) and has proven applicable in a network formed by IIoT devices [25]. IOTA's fee-less microtransactions, a throughput of 1500 tps [26], and low power consumption [27] make it the selected DLT technology to build the attribute spoofing prevention system. Readers should consider that the DLT choice works assuming that every company in the value chain has established a minimal security architecture. Instead, if IOTA nodes are at risk of being compromised, a reputation-based layer may need to be added [22].

The attribute prevention architecture based on IOTA and IPFS is presented in Figure 3. As can be seen, it is an extension of Figure 2, and it is formed by the same two phases: user registration and private key retrieval. During registration (Algorithm 1), a company node uploads the attributes to IPFS and stores the CID in the IOTA Tangle. This secure storage ensures that the CID has not been modified since IOTA provides the auditability that IPFS alone cannot guarantee.

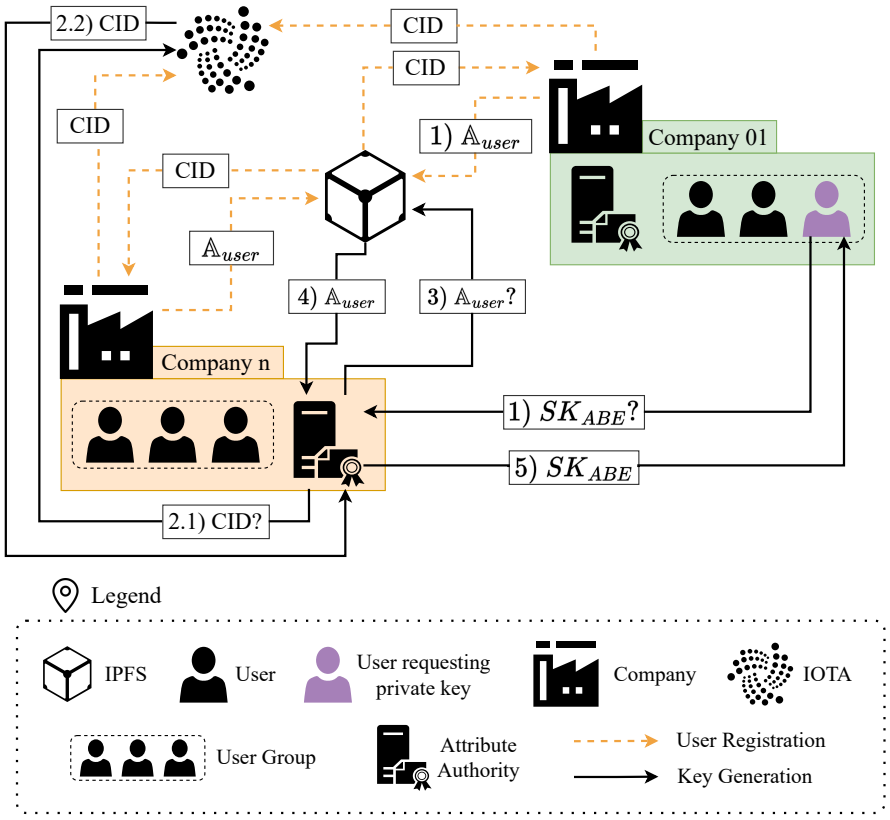

Figure 3. CP-ABE attribute validation with IPFS and IOTA.

Algorithm 1: User Registration

Input: $\mathbb{A}_{user}$, *IOTA_ClientNode*
Output: *msgID*
1 *IPFS_Node = IPFS.Create*
2 $CID = IPFS\_Node.add\ (\mathbb{A}_{user})$
3 *msgID = IOTA.send(IOTA_ClientNode, CID)*

The step by step process for Algorithm 1 is presented below:

1. Algorithm 1 takes as input users' attributes $\mathbb{A}_{user}$ and the address of the IOTA client node that will store the information, *IOTA_ClientNode*.
2. The Algorithm creates *IPFS_Node*, the IPFS node that will store the information.
3. $\mathbb{A}_{user}$ is stored in *IPFS_Node*, which generates a CID.
4. The CID is stored in IOTA, for which the CID is sent to the *IOTA_ClientNode* used as input.
5. Uploading the CID to IOTA generates the associated *msgID*, which is the output of Algorithm 1.

Once users have their $\mathbb{A}$ stored, they can require their private keys from the Attribute Authorities. The Attribute Authorities generate users' private keys following Algorithm 2. The step-by-step process is defined below:

Algorithm 2: User SK_{ABE} Generation

Input: *msgID IOTA_ClientNode*
Output: SK_{ABE}
1 $CID = IOTA.retrieve(IOTA\_ClientNode, msgID)$
2 $\mathbb{A}_{user} = IPFS.retrieve(CID)$
3 $KeyGen(\mathbb{A},\ MPK,\ MSK) \rightarrow SK_{ABE}$

1. Users requests a private key SK_{ABE} from Attribute Authorities.
2. Attribute Authorities obtain the CID from IOTA using their *IOTA_ClientNode* to search for a specific *msgID*.
3. With the CID, the Authorities retrieve the content (i.e., user's $\mathbb{A}$) from IPFS. They do not need to know which IPFS node has the content, *IPFS.retrieve* searches the IPFS network for the content matching the CID.
4. If the retrieved content passes the integrity check, the Attribute Authorities use $\mathbb{A}$ as input to generate the users' private key SK_{ABE}. The private key is generated with Equation (5).
5. Finally, users get their SK_{ABE}, according to the attribute set $\mathbb{A}$ their company conceded them.

Thus, with the combination of IPFS, IOTA, Algorithms 1 and 2, the attribute-spoofing prevention system is implemented. Attribute Authorities do not have to manage all the information; instead, attribute management is distributed by the combination of IPFS and IOTA. Thanks to IPFS, the solution obtains auditability, which allows the system to know who has stored the information, and thanks to the CID, integrity violations in IPFS files can be detected. Finally, the distributed nature of both technologies (IPFS and IOTA) protects the system against single-point failures.

4.3. Federated Identity Management for User Authentication

Finally, R3 also mandated user authentication. The scenario considered in this work requires users to authenticate with the Attribute Authorities of which SK_{ABE} they require. Similarly, Attribute Authorities must identify users coming from different environments. This identification becomes particularly critical in dCP-ABE, where the *UID* is crucial to prevent private key collision.

Identification and authentication reduce the system's scalability by adding operations prior to key generation. Therefore, it is necessary to have an efficient and scalable authentication system that follows the no-trust assumption [28]. That is, it should not be necessary for the different companies in the system to establish trust relationships between them.

The authentication system chosen for our attribute spoofing prevention system is Federated Identity Management (FIM) [29], whose behavior is shown in Figure 4. FIM allows users from different companies to use their company credentials to authenticate to different Attribute Authorities. This also implies that Attribute Authorities do not have to manage the credentials of multiple users from different companies. Instead, they verify the token issued through an Identity Provider (IdP). This IdP establishes a trust relationship with the different companies and acts as an intermediary between the Attribute Authorities and the companies. This way, Attribute Authorities only have to manage the trust relationship with the IdP. The FIM message exchange is performed through the users' browser and is described below:

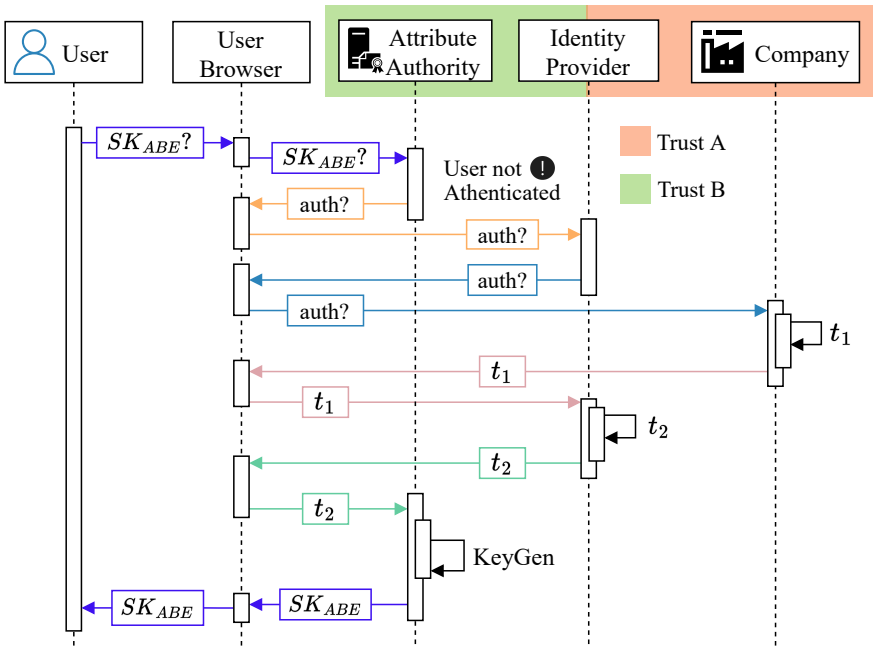

Figure 4. FIM message exchange.

1. Users request SK_{ABE} from the Attribute Authorities through their browser.
2. If the Authorities detect that the user has not been authenticated, it triggers the authentication process by sending an *auth* authentication request through the users' browsers.
3. The request is sent to the IdP, the only role with which the Attribute Authorities have a trusted connection during the authentication process.
4. The IdP has a trusted connection with the various companies in the value chain. Thus, it redirects the authentication request to the company to which the user belongs via the user's browser.
5. The company to which the user belongs sends an authentication token t_1 to the IdP that requested it.
6. The IdP uses the token t_1 to generate a second token, t_2.
7. The IdP relies on the user's browser to send token t_2 to the Attribute Authorities.
8. Once the Attribute Authorities receive t_2, they run the Algorithm 2 presented in the previous section to generate the users' SK_{ABE}.

5. Attribute-Spoofing Prevention System Evaluation

Section 3 introduced the security risks posed by attribute spoofing in CP-ABE and dCP-ABE schemes and the requirements the attribute spoofing prevention system has to

fulfill to prevent it. Table 2 summarizes how meeting the defined requirements prevents the identified attack vectors and which technology achieves that compliance.

Table 2. Requirement fulfilment. ▣ stands for not relevant.

Requirement	Attack Vector 1 2	Proposed Solution 1 2 3
	1. AV 1 2. AV 2	1. IOTA 2. IPFS 3. FIM
R1	■ □	□ ■ □
R2	■ ■	■ □ ■
R3	□ ■	■ □ □
R4	▣ ▣	▣ ▣ ▣

The fulfilment of R1, R2 and R3 is analysed in Section 5.1, while the fulfilment of R4 is experimentally verified in Section 5.2.

5.1. Qualitative Evaluation

R1 requires distributed attribute storage, and meeting this requirement by using IPFS deters AV 1. This attack depends on users transmitting their attributes to the Authorities. Therefore, although the Authorities still do not know which attributes belong to which users (Assumption 2), that information is provided by the companies storing the information in IPFS. Furthermore, users and attackers cannot claim a private key containing more attributes than they possess since they do not interact with the information stored in the IPFS. Therefore, the main consequences of AV 1 (privilege escalation and illegal acquisition of SK_{ABE}) are stopped.

R2 requires attribute validation and user authentication. The ones that store information in IPFS are the original companies, which we presume are honest at the beginning. However, if one is compromised or an external attacker gets access to the stored information, the CID stored in IOTA changes. Thus, IPFS detects the integrity violation, and the attacker or the compromised node can be traced back using IOTA. In addition, knowing the set of compromised attributes also allows partners in the value chain to know which private keys have been compromised. Regarding user authentication, it can be provided by FIM based on Assumption 3 and Assumption 4. Since the distributed storage is protected, the solution prevents AV 2. Attackers can no longer modify the information contained in the storage solution, preventing the consequence shown in Table 1. Furthermore, the attack is prevented even in the presence of Assumption 1 and Assumption 2. The Attribute Authorities do not need to know the attribute universe U or users' attribute sets $\mathbb{A}$: all that information is retrieved from IOTA and IPFS. In addition, FIM prevents a user from providing a false *UID*.

R3 is fulfilled by combining IPFS with IOTA. As explained, IOTA provides traceability and, combined with IPFS, ensures attribute integrity and auditability. Auditability protects the system against AV 2 since data modifications can be traced back to the user who made them.

Therefore, the chosen combination of technologies meets most established requirements and avoids the identified attack vectors. However, the last requirement (R4) alludes to the suitability of the solution for IIoT devices. In order to validate this last requirement, the following subsection presents the experimental evaluation.

5.2. Experimental Evaluation

The experimental evaluation assesses the fulfillment of R4. The experiment tests the time required by the Authorities to retrieve the attributes from the attribute spoofing prevention system and use them to generate the users' private keys. The experiment also considers other metrics to validate R4, like the power consumption or the Transactions per Second (tps) to download the attributes from the system. The chosen library to generate the private keys is OpenABE (https://github.com/zeutro/openabe, accessed on 1 November 2022). The experiment uses a Raspberry Pi 4 (RPI4) with a 32-bit Ubuntu Server TLS and 8GB of RAM as the Attribute Authority, while data has been uploaded to IOTA and IPFS using the WSL running Ubuntu 20.04.5 LTS. IPFS stores a local copy of the uploaded data; thus, uploading it from a different device is crucial to measure the time correctly. Users' attributes are stored in JSON files, following the structure shown in Listing 1.

Listing 1. JSON with users' attribute set $\mathbb{A}$.

```
{
    "issuer": <Company ID>
    "userID" : <User ID>,
    "attribute" : "(Attr1||Attr2||...||Attrn)",
    "timestamp": <timestamp>
}
```

We name the time elapsed between the Attribute Authorities requesting the user's attributes and obtaining them as T_{AS}. The obtained user's attribute set is $\mathbb{A} = (\text{Attr1}\|\text{Attr2}\|\text{Attr3}\|\text{Attr4}\|\text{Attr5}\|\cdots\|\text{Attrn})$. Measurements are performed from n = 1 to n = 20. Afterward, the native OpenABE benchmarking tool measures the time required to generate SK_{ABE} in W11 according to the $\mathbb{A}$ retrieved from IPFS. We run 100 iterations for each $\mathbb{A}$ and calculate the mean time. We denote this time as T_{GEN}.

To establish whether the solution is deployable on IIoT devices, we need to establish a baseline against which to compare. Regarding the power-consumption for running IOTA, in a RPI4 is around 1.18 mJ–1.21 mJ per message sent or retrieved, according to the IOTA foundation (https://wiki.iota.org/learn/about-iota/energy-efficiency/, accessed on the 18 February 2023). Meanwhile, regarding the time required for our solution, obtaining attributes and generating private keys is a process that is carried out as a step before exchanging information. In this sense, conceptually, the objective is similar to that of the TLS/DTLS handshake. Therefore, we rely on the work by [30], in which the authors measure the time required to improve the efficiency of the DTLS handshake. Their proposal achieves an average time of 250 ms, which they consider adequate for IIoT devices, which we take as the baseline for our experiment. Thus, the solution proposed in this paper is feasible if $T_{AS} + T_{GEN} < 250$ ms.

Figure 5 presents the experimental evaluation results, which clearly show that the limit of 250 ms is not exceeded in any case, which is the threshold set to consider the solution acceptable. One aspect to highlight in Figure 5 is the constant time required to obtain the data from the attribute spoofing prevention solution. It is observed that, regardless of the size of the file storing the attributes, the average time to obtain them from IPFS is 145 ms. This result is related to how IPFS stores the information in 256 kB blocks. If the stored file is smaller than 256 kB, a single block is enough to store it. If it is larger, the file is split into several 256 kB blocks, and a last block is generated whose content is used to link the previous ones. Several tests have been performed with JSONS of different sizes, and the IPFS block limit is not exceeded in any cases. As a consequence, the amount of blocks to get from IPFS is always the same, which causes similar times for their download.

On the other hand, before getting the IPFS data, the CID has to be retrieved from IOTA. However, the CID is derived from the hash of the IOTA file and thus has a fixed size of

46 bytes when converted to ASCII. This value is then converted to an array and stored in IOTA, which has a maximum transaction size of 1606 bytes. Consequently, the amount of transactions to recover from IOTA is always the same, and since the CID size « IOTA transactions size, only one transaction is required to retrieve the CID. This implies that the tps is independent of the number of attributes stored in IPFS, making an average of 1.39 tps for the Raspberry Pi4 when downloading the attributes. Additionally, this implies that T_{AS} is always the same, regardless of the number of attributes to recover. The consequence of this is that, although $T_{AS} > T_{GEN}$, the one that can cause the limit set in the baseline to be exceeded is T_{GEN}.

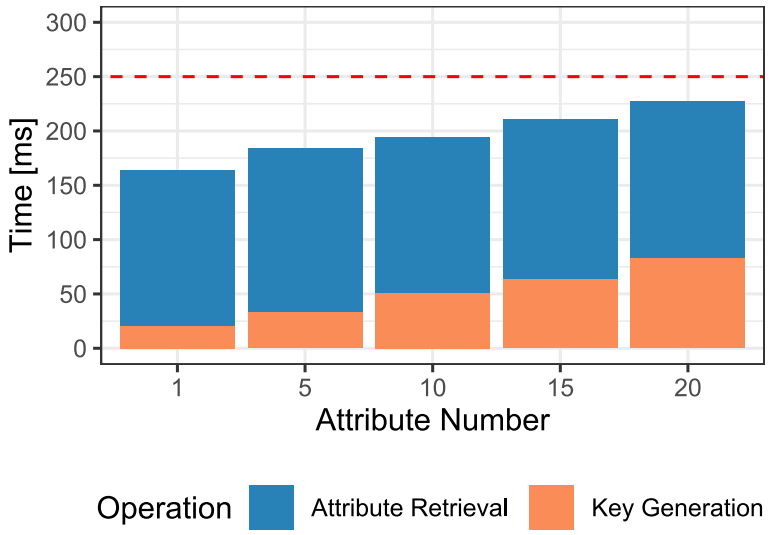

Figure 5. Time in ms to generate SK_{ABE} with attribute validation.

As Figure 5 shows, the time required for key generation increases linearly with the number of attributes contained in it. Therefore, knowing that the Authority takes an average of 145 ms to obtain the attributes and 20 ms to generate a key with 1 attribute, and 83 ms to generate a key with 20 attributes, it can be calculated that the number of attributes that will exceed the limit will be 26. Since having 26 attributes is a far-fetched assumption for a real environment, it is implausible that the 250 ms limit will be exceeded.

It can be concluded that the inclusion of the attribute spoofing prevention system satisfies the imposed constraint of $T_{AS} + T_{KG} < 250$ ms. The delay added by the solution falls within the margin considered for it to be acceptable, especially given the added security; thus, R4 is satisfied.

6. Conclusions

This paper identifies the issue of attribute spoofing in CP-ABE and dCP-ABE. It outlines the basis of the attack and proposes an attribute spoofing prevention system that relies on a DAG-type DLT, IOTA; on a distributed storage based on IPFS and on an authentication system based on FIM. This combination of solutions addresses the requirements defined after identifying the potential attack vectors for attribute spoofing.

In this regard, the combination of IOTA and IPFS ensures secure attribute storage, and using FIM for authentication prevents users from claiming pieces of private key bound to a *UID* that is not theirs. By doing so, the system distributes responsibility and trust among each system member and protects them against single-point failures, impersonation, and attribute spoofing. This, in turn, reinforces the secure E2E data exchange and ensures the integrity and confidentiality of the transmitted information.

Thus, R1, R2, and R3 defined for the system are met, and complying with them prevents the exploitation of the identified attribute spoofing attack vectors. However,

for the solution to be suitable, R4 was also established, which considers that the solution must be deployable in all types of devices, including IIoT devices. For this purpose, R4 was established, whose compliance is verified through an experimental evaluation, which demonstrates the feasibility of deploying the solution in devices with reduced capacities. Therefore, it can be concluded that the proposed solution can prevent attribute spoofing.

Author Contributions: Conceptualization, A.M.-S., M.B. and J.A.; methodology, A.M.-S.; software, A.M.-S.; validation, A.M.-S., M.B. and J.A.; formal analysis, A.M.-S., M.B. and J.A.; investigation, A.M.-S.; resources, A.U.; writing—original draft preparation, A.M.-S.; writing—review and editing, A.M.-S., M.B. and J.A.; supervision, M.B., J.A. and A.U.; project administration, A.M.-S., M.B. and J.A.; funding acquisition, A.U. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been financed by The European commission through the Horizon Europe program under the ZDZW project (grant agreement number 101057404).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sahai, A.; Waters, B. Fuzzy Identity-Based Encryption. In Proceedings of the EUROCRYPT 2005, Aarhus, Denmark, 22–26 May 2005; pp. 457–473. [CrossRef]
2. Goyal, V.; Pandey, O.; Sahai, A.; Waters, B. Attribute-based Encryption for Fine-grained Access Control of Encrypted Data. In Proceedings of the 13th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 30 October–3 November 2006; Association for Computing Machinery: New York, NY, USA, 2006; pp. 89–98. [CrossRef]
3. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-Policy Attribute-Based Encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP '07), Berkeley, CA, USA, 20–23 May 2007; IEEE—Institute of Electrical and Electronics Engineers Inc.: Berkeley, CA, USA, 2007; pp. 321–334. [CrossRef]
4. Rouselakis, Y.; Waters, B. Efficient Statically-Secure Large-Universe Multi-Authority Attribute-Based Encryption. In Proceedings of the Financial Cryptography and Data Security: 19th International Conference, FC 2015, San Juan, Puerto Rico, 26–30 January 2015; pp. 315–332. [CrossRef]
5. Pennekamp, J.; Bader, L.; Matzutt, R.; Niemietz, P.; Trauth, D.; Henze, M.; Bergs, T.; Wehrle, K. Private Multi-Hop Accountability for Supply Chains. In Proceedings of the 2020 IEEE International Conference on Communications Workshops (ICC Workshops), Virtual, 7–11 June 2020; pp. 1–7. [CrossRef]
6. Mosteiro-Sanchez, A.; Barcelo, M.; Astorga, J.; Urbieta, A. "Are you what you claim to be?" Attribute Validation with IOTA for Multi Authority CP-ABE. In Proceedings of the Blockchain and Applications, 4th International Congress, L'Aquila, Italy, 13–15 July 2022; Volume 595. [CrossRef]
7. Liu, R.; Kumar, A. Leveraging information sharing to configure supply chains. *Inf. Syst. Front.* **2011**, *13*, 139–151. [CrossRef]
8. Epiphaniou, G.; Pillai, P.; Bottarelli, M.; Al-Khateeb, H.; Hammoudesh, M.; Maple, C. Electronic Regulation of Data Sharing and Processing Using Smart Ledger Technologies for Supply-Chain Security. *IEEE Trans. Eng. Manag.* **2020**, *67*, 1059–1073. [CrossRef]
9. Qi, S.; Zheng, Y.; Li, M.; Liu, Y.; Qiu, J. Scalable Industry Data Access Control in RFID-Enabled Supply Chain. *IEEE/ACM Trans. Netw.* **2016**, *24*, 3551–3564. [CrossRef]
10. Lu, Y.; Wang, Y.; Dai, X.; Li, J.; Li, J.; Chen, M. Survey of Attribute-Based Encryption in Cloud Environment. In Proceedings of the Cognitive Cities: Second International Conference, IC3 2019, Kyoto, Japan, 3–6 September 2019; Shen, J., Chang, Y.C., Su, Y.S., Ogata, H., Eds.; Springer: Singapore, 2020; pp. 375–384.
11. Di Francesco Maesa, D.; Lunardelli, A.; Mori, P.; Ricci, L. Exploiting Blockchain Technology for Attribute Management in Access Control Systems. In Proceedings of the Economics of Grids, Clouds, Systems, and Services: 16th International Conference, GECON 2019, Leeds, UK, 17–19 September 2019; pp. 3–14. [CrossRef]
12. Di Francesco Maesa, D.; Mori, P.; Ricci, L. A blockchain based approach for the definition of auditable Access Control systems. *Comput. Secur.* **2019**, *84*, 93–119. [CrossRef]
13. Nakanishi, R.; Zhang, Y.; Sasabe, M.; Kasahara, S. Combining IOTA and Attribute-Based Encryption for Access Control in the Internet of Things. *Sensors* **2021**, *21*, 5053. [CrossRef] [PubMed]
14. Preuveneers, D.; Joosen, W.; Bernal Bernabe, J.; Skarmeta, A.F. Distributed Security Framework for Reliable Threat Intelligence Sharing. *Secur. Commun. Netw.* **2020**, *2020*, 8833765. [CrossRef]
15. Thatmann, D.; Butyrtschik, A.; Küpper, A. A Secure DHT-Based Key Distribution System for Attribute-Based Encryption and Decryption. In Proceedings of the 2015 9th International Conference on Signal Processing and Communication Systems (ICSPCS), Cairns, Australia, 14–16 December 2015; pp. 1–9. [CrossRef]
16. Cui, L.; Yang, S.; Chen, Z.; Pan, Y.; Xu, M.; Xu, K. An Efficient and Compacted DAG-Based Blockchain Protocol for Industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2020**, *16*, 4134–4145. [CrossRef]

17. Fernández-Caramés, T.M.; Blanco-Novoa, O.; Froiz-Míguez, I.; Fraga-Lamas, P. Towards an Autonomous Industry 4.0 Warehouse: A UAV and Blockchain-Based System for Inventory and Traceability Applications in Big Data-Driven Supply Chain Management. *Sensors* **2019**, *19*, 2394. [CrossRef] [PubMed]
18. Zichichi, M.; Ferretti, S.; D'Angelo, G. A Distributed Ledger Based Infrastructure for Smart Transportation System and Social Good. In Proceedings of the 2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 10–13 January 2020; pp. 1–6. [CrossRef]
19. Shahjalal, M.; Islam, M.M.; Alam, M.M.; Jang, Y.M. Implementation of a Secure LoRaWAN System for Industrial Internet of Things Integrated With IPFS and Blockchain. *IEEE Syst. J.* **2022**, *16*, 5455–5464 . [CrossRef]
20. Hu, J.; Deng, J.; Gao, N.; Qian, J. Application Architecture of Product Information Traceability Based on Blockchain Technology and a Lightweight Secure Collaborative Computing Scheme. In Proceedings of the 2020 International Conference on E-Commerce and Internet Technology (ECIT), Zhangjiajie, China, 22–24 April 2020; pp. 335–340. [CrossRef]
21. Fernández-Caramés, T.M.; Fraga-Lamas, P. A Review on the Application of Blockchain to the Next Generation of Cybersecure Industry 4.0 Smart Factories. *IEEE Access* **2019**, *7*, 45201–45218. [CrossRef]
22. Stefanescu, D.; Galán-García, P.; Montalvillo, L.; Unzilla, J.; Urbieta, A. Towards a Holistic DLT Architecture for IIoT: Improved DAG for Production Lines. In Proceedings of the Blockchain and Applications, 3th International Congress, Salamanca, Spain, 6–8 October 2021; pp. 179–188. [CrossRef]
23. Sealey, N.; Aijaz, A.; Holden, B. IOTA Tangle 2.0: Toward a Scalable, Decentralized, Smart, and Autonomous IoT Ecosystem. *arXiv* **2022**, arXiv:2209.04959. [CrossRef]
24. Stefanescu, D.; Montalvillo, L.; Galán-García, P.; Unzilla, J.; Urbieta, A. A Systematic Literature Review of Lightweight Blockchain for IoT. *IEEE Access* **2022**, *10*, 123138–123159. [CrossRef]
25. Rosenberger, J.; Rauterberg, F.; Schramm, D. Performance study on IOTA Chrysalis and Coordicide in the Industrial Internet of Things. In Proceedings of the 2021 IEEE Global Conference on Artificial Intelligence and Internet of Things (GCAIoT), Dubai, United Arab Emirates, 12–16 December 2021; pp. 88–93. [CrossRef]
26. Conti, M.; Kumar, G.; Nerurkar, P.; Saha, R.; Vigneri, L. A survey on security challenges and solutions in the IOTA. *J. Netw. Comput. Appl.* **2022**, *203*, 103383. [CrossRef]
27. Helmer, L.; Penzkofer, A. Report on the energy consumption of the IOTA 2.0 prototype network (GoShimmer 0.8.3) under different testing scenarios. *arXiv* **2022**, arXiv:2210.13996. [CrossRef]
28. Bader, L.; Pennekamp, J.; Matzutt, R.; Hedderich, D.; Kowalski, M.; Lücken, V.; Wehrle, K. Blockchain-based privacy preservation for supply chains supporting lightweight multi-hop information accountability. *Inf. Process. Manag.* **2021**, *58*, 102529. [CrossRef]
29. Hardt, D. The OAuth 2.0 Authorization Framework. Available online: https://protect-au.mimecast.com/s/zNjQCQnzV0igzL7mivocg6?domain=hjp.at (accessed on 29 January 2023).
30. Atutxa, A.; Astorga, J.; Barcelo, M.; Urbieta, A.; Jacob, E. Improving efficiency and security of IIoT communications using in-network validation of server certificate. *Comput. Ind.* **2022**, *144*, 103802. [CrossRef]