electronics

MDPI

Article

T-FIM: Transparency in Federated Identity Management for Decentralized Trust and Forensics Investigation

Bowen Xu [1,2], Zhijintong Zhang [3], Aozhuo Sun [1,2], Juanjuan Guo [4], Zihan Wang [5,*], Bingyu Li [3], Jiankuo Dong [6], Shijie Jia [1,2] and Li Song [1,2]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100045, China; xubowen@iie.ac.cn (B.X.)
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing 101408, China
[3] School of Cyber Science and Technology, Beihang University, Beijing 100191, China
[4] Cloud Computing & Big Data Research Institute, China Academy of Information and Communications Technology, Beijing 100191, China
[5] National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China
[6] School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
* Correspondence: wangzihan@cert.org.cn

Abstract: Federated Identity Management (FIM) has gained significant adoption as a means to simplify user authentication and service authorization across diverse domains. It serves as a centralized authentication and authorization method, enabling users to access various applications or resources using credentials issued by a universally trusted identity provider (IdP). However, recent security incidents indicate that the reliability of credentials issued by IdP is not absolute in practice. If the IdP fails, it can persistently access any application that trusts it as any user. This poses a significant security threat to the entire system. Furthermore, with the increasing adoption of FIM across diverse scenarios, there is a growing demand for the development of an identity management system that can effectively support digital forensics investigations into malicious user behavior. In this work, we introduce transparency to federated identity management, proposing T-FIM to supervise unconditional trust. T-FIM employs privacy-preserving logs to record all IdP-issued tokens, ensuring that only the true owner can access the exact token. We utilize identity-based encryption (IBE), but not just as a black box, encrypting tokens before they are publicly recorded. In addition, we propose a decentralized private key generator (DPKG) to provide IBE private keys for users, avoiding the introduction of a new centralized trust node. T-FIM also presents a novel approach to digital forensics that enables forensic investigators to collect evidence in a privacy-preserving manner with the cooperation of the DPKG. We conduct a comprehensive analysis of the correctness, security, and privacy aspects of T-FIM. To demonstrate the practical feasibility of T-FIM, we evaluated the additional overhead through experimental evaluations. Additionally, we compared its performance with other similar schemes to provide a comprehensive understanding of its capabilities and advantages.

Keywords: federated identity management; transparency; privacy; digital forensics

Citation: Xu, B.; Zhang, Z.; Sun, A.; Guo, J.; Wang, Z.; Li, B.; Dong, J.; Jia, S.; Song, L. T-FIM: Transparency in Federated Identity Management for Decentralized Trust and Forensics Investigation. *Electronics* **2023**, *12*, 3591. https://doi.org/10.3390/electronics12173591

Academic Editor: Dimitra I. Kaklamani

Received: 26 July 2023
Revised: 19 August 2023
Accepted: 22 August 2023
Published: 25 August 2023

1. Introduction

Federated Identity Management (FIM) [1] is considered as the most promising solution to achieve reliable and effective collaboration by providing efficient authentication mechanisms and use of identity information across multiple domains. FIM has been a de facto authentication and authorization method that is widely used in Cloud-Based Service Integration, Enterprise Authentication, Authorization, and Cross-Organizational Collaboration, etc. [2]. An FIM system involves an Identity Provider (IdP) and multiple Service Providers (SPs) that enable users to access SPs by using one set of credentials (i.e., tokens, assertions) signed by the trusted IdP. Several standardized protocols can be

used to implement an FIM system such as OAuth 2.0 [3], OIDC [4], SAML [5] and WS-Federation [6]. In addition, there are many influential Internet companies (e.g., Google [7], Facebook [8], GitHub [9], PayPal [10] and Microsoft [11]) that provide services as IdPs (e.g., Active Directory Federated Service [12], Azure Active Directory [13]).

As a method of centralized identity authentication and authorization, FIM introduces IdP as the trusted third party for user identity management. However, the IdP, serving as the trust anchor of the FIM system, is not completely reliable because it can be compromised or become malicious due to some single-point-of-failure problem [14–16]. For instance, in the significant security incident known as the "SolarWinds" attack [17], the attackers utilized the "Golden SAML" technique [16] to compromise the IdP (i.e., Active Directory Federation Services authentication). The attacker first exploits the IdP service to steal the IdP signature key. After obtaining the signature key, the attacker can impersonate the IdP to sign the SAML assertion of any user on the service. By forging SAML responses, they were able to masquerade as legitimate users with valid authentication tokens. This enabled them to access federated services and operate undetected within the compromised networks. The combination of the "SolarWinds" attack and the "Golden SAML" technique allowed the hackers to conduct a large-scale, sophisticated cyber attack, resulting in data breaches of multiple government agencies. These instances underscore the inherent vulnerabilities associated with the IdP-centric FIM approach and emphasize the imperative of implementing additional security measures and safeguards to fortify the FIM system against compromised or malicious IdP.

Researches have proved that third-party providers should not be fully trusted by default or as expected [18]. For example, in Public Key Infrastructure (PKI), another centralized identity management system, accredited Certificate Authorities (CAs) are not considered to be inherently trustworthy in practice. Many CAs have been compromised or deceived to issue fraudulent certificates [19]. In order to address the trust issues of CAs, various certificate transparency schemes [20,21] have been proposed by researchers and are now utilized in deployed PKI systems. Compared to CAs in PKI, the online signing components of IdPs have a larger attack surface, which poses security risks for FIM systems due to the presence of a single point of failure. Unfortunately, this failure becomes even more severe as it enables adversaries to gain persistent access to any SP that trusts the compromised IdP. As participants (i.e., users and SPs) in an FIM system, there is little that can be done to prevent the misbehaviour of the IdP, such as mis-issuing tokens. Therefore, it is highly advised to swiftly take necessary steps to secure the FIM infrastructure and establish effective monitoring mechanisms to detect and address such misbehaviours. Research works have delved into fortifying the FIM system against compromised or malicious IdP. For instance, TicketT scheme is proposed to provide accountable SSO services with privacy-preserving public logs against potentially fraudulent tickets issued by a compromised IdP [18]. However, TicketT faces two security concerns. Firstly, it introduces a new trust center—a private key generator (PKG). Secondly, in TicketT, the user fails to effectively oversee the tickets when maliciously encrypted with other users' keys by an adversary. Additionally, several schemes for self-sovereign identity (SSI) have been proposed [22–24] to enhance users' control over their identifiers. The SSI schemes rely on public permissionless/permissioned blockchain (distributed ledger) as the trust anchor, and utilize W3C-standardized decentralized identifiers (DIDs) to represent users' identities. In SSI system, private keys are managed by individual users. However, many security vulnerabilities have proved that personal management key obviously poses a more significant security risk.

In this paper, we introduced the concept of transparency into FIM to provide the ability to supervise IdP, known as T-FIM. The idea of transparency has been extensively applied in various domains such as PKI with Certificate Transparency (CT) [20,21], end-to-end encryption with Key Transparency [25,26], and software supply chain with Sigstore [27]. These applications of transparency aim to manage the unregulated and unconditional trust that exists in these systems. In transparency schemes, key operations of centralized

trust nodes are publicly logged for inspection by stakeholders. Although it cannot prevent misbehaviour, any violation of its promise by the trusted node will be discovered.

Unlike common transparency schemes, the credential (i.e., token, assertion) in FIM to be logged contains private information such as the service requester, service provider, the occurrence time of login/access activities, etc. Enabling users to deterministically detect IdP misbehaviour while preserving privacy is a formidable challenge. However, T-FIM provides a reliable solution for detecting IdP misbehaviour without compromising user privacy. Specifically, all IdP-issued tokens have to be recorded in privacy-preserving logs before being used, where the token can only be checked by its true owner, the user bound to the identity in the token. As monitors, users detect malicious behavior of the IdP by checking tokens issued by the IdP for themselves. We achieve this by employing identity-based encryption (IBE), which encrypts the tokens while considering potential malicious actors, rather than using IBE as a black box. Furthermore, we design a decentralized private key generator (DPKG) to generate IBE private keys for users, eliminating any unregulated single point of trust. Notably, the DPKG in T-FIM generates the master key in a distributed manner, which is distinct from the basic DPKG for identity-based cryptography, where a dealer generates and shares the key among independent authorities [28,29]. In the basic DPKG scheme, if the dealer is malicious, they have the ability to individually generate the IBE private key of any user. However, in our decentralized private key generator (DPKG) introduced in T-FIM, the IBE private key can only be leaked when the number of malicious DPKG members exceeds the threshold value.

Moreover, the logged tokens in T-FIM can also serve the purpose of digital forensics, an applied science for identifying, collecting, organizing, and presenting digital evidence. Digital forensics plays an increasingly important role in connecting individuals to criminal activities, as criminals have been utilizing web applications and emerging technologies to commit crimes in recent years [30]. The two most important security requirements of digital forensics are: (*i*) continuous integrity, which ensures that the evidence is reliable and untampered; and (*ii*) privacy preservation, which involves compliance with new privacy laws, such as the General Data Protection Regulation (GDPR) [31]. T-FIM is well-matched to the requirements of evidence collecting in digital forensics across multiple systems. Firstly, it provides a comprehensive view of the suspect's access activities for the forensic investigator. Secondly, evidence items in the log are organized as Merkle hash trees to ensure integrity. Thirdly, the design of the privacy-preserving log and DPKG prevents the intrusion of user privacy. However, digital forensics in T-FIM is facing a challenge in extracting evidence while preserving privacy. To address this, we employ a joint decryption method to precisely disclose tokens within the authorized scope (i.e., tokens utilized by a suspect during a particular timeframe) to the forensics investigator.

In summary, we design T-FIM, an FIM system with transparency that effectively eradicates security threats associated with a centralized trust model. This transparency provides accountability in two aspects: (*i*) users are able to hold the IdP accountable, which was originally trusted unconditionally, and (*ii*) the forensic investigator is able to hold users accountable, that is, it can investigate specific user login/access actions while preserving privacy. Moreover, we provide systematic proof of correctness, security, and privacy for T-FIM in a malicious model. To improve the performance of T-FIM, we employ the most efficient pairing-friendly elliptic curve that conforms to the 128-bit security level (i.e., BLS12-381) for system implementation. Furthermore, in T-FIM, the privacy-preserving log is sharded according to user identities, effectively reducing monitoring costs. Finally, we experimentally illustrate the feasibility of T-FIM for practical applications. We generate 1M random samples according to the token structure in the wild for the experiment. The evaluation results indicate that T-FIM introduces an additional 97 ms/341 ms communication overhead (when using TLS 1.3/TLS 1.2) and 17 ms computation overhead to the authentication/authorization protocol compared with traditional FIM. In addition, T-FIM allows users to efficiently monitor and check tokens at a rate of 27 K per minute, which enables quick detection of any misbehaviour by the IdP. Moreover, the cost of forensics for

investigators is t times higher than that of user monitoring, where t represents the DPKG threshold. In general, our contributions can be summarized as follows:

- ⋆ We introduce the concept of transparency in FIM and design the federated identity management with transparency—T-FIM, enabling users to detect mis-issued tokens issued by the compromised or malicious IdP while preserving privacy, thereby strengthening the trust model.
- ⋆ We propose a novel digital forensics approach for collecting login/access activities of suspects, leading to significant improvements in the streamlined forensic investigation process. This approach ensures the integrity of evidence and assists investigators in conveniently collecting evidence from multiple systems in a centralized manner.
- ⋆ We systematically prove the security of T-FIM against the malicious model and evaluate the overhead at each stage to demonstrate its feasibility.

The rest of the paper is organized as follows. We introduce preliminaries in Section 2, including IBE algorithm and transparency schemes. Next, we summarize the design goals, challenges and threat model of T-FIM in Section 3. Furthermore, the complete design of T-FIM is presented in Section 4, followed by a correctness, security, and privacy analysis in Section 5 and a performance evaluation in Section 6. Finally, we review the related work in Section 7 and conclude the paper in Section 8.

2. Preliminary

In this section, we present essential preliminary concepts and components that are relevant to our study such as IBE and Bilinear Pairing. Additionally, we discuss related transparency schemes and existing solutions that share similar design goals, such as ticket transparency.

2.1. Identity-Based Encryption

IBE is a type of public-key encryption that enables a user to generate a public key from a known unique identifier, such as an email address or an app account number. The private key generator (PKG) then calculates the corresponding private key from the public key. Therefore, IBE eliminates the need to distribute public keys in advance of exchanging encrypted data.

Next, we describe the specific algorithm of IBE from the Weil Pairing [32]. IBE consists of four key algorithms: *Setup*, *Extract*, *Encrypt*, and *Decrypt*. These algorithms are complemented by two fundamental operations: *Bilinear Pairing* and *Hash Functions*.

- Setup. Pick a master private key $sk \in \mathbb{Z}_q^*$. Set master public key $pk = [g, h = g^{sk}]$.
- Extract. For a given string $ID \in \{0,1\}^*$, PKG calculates the corresponding private key $d_{ID} = \mathcal{H}_1(ID)^{sk}$.
- Encrypt. To encrypt M under the public key ID do the following: (*i*) choose a random $r \in \mathbb{Z}_q^*$; (*ii*) set the ciphertext to be $C = [g^r, \mathcal{H}_2(\hat{e}(\mathcal{H}_1(ID), h)^r) \oplus M]$.
- Decrypt. Let $C = [u, v]$, then decrypting C using the private key d_{ID} is to calculate $M = \mathcal{H}_2(\hat{e}(d_{ID}, u)) \oplus v$.

Bilinear Pairing.Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of order q for some large prime q. A bilinear map is defined $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. The map has the following properties: (*i*) Bilinearity: $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$ where $a, b \in \mathbb{Z}_q^*$. (*ii*) Non-degeneracy: $\hat{e}$ does not send all pairs of points in $\mathbb{G}_1 \times \mathbb{G}_2$ to the identity in $\mathbb{G}_T$. If g_1/g_2 is a generator of $\mathbb{G}_1/\mathbb{G}_2$, then $\hat{e}(g_1, g_2)$ is a generator of $\mathbb{G}_T$. (*iii*) Computation: for all $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$, the map $\hat{e}(P, Q)$ is efficiently computable.

Hash Functions. $\mathcal{H}_1$: $\{0,1\}^* \to \mathbb{G}_1$ and $\mathcal{H}_2$: $\mathbb{G}_T \to \{0,1\}^n$ are used.

2.2. Transparency Schemes

Transparency schemes are intended to provide a means of policing trusted third parties that are not entirely reliable. That is, it spreads the centralized trust in a single

node to all stakeholders. Specifically, in transparency schemes, key operations (e.g., certificate issuance) of the trusted third party (e.g., CA) are publicly recorded in a *log server* for stakeholder (e.g., domain owner) inspection. The records in a log are organized as a Merkle hash tree to ensure that they are append-only. After that, the stakeholder obtains information of interest from the log through the *monitor*, and further detects misbehaviour. Additionally, redundant lightweight *auditors* ensure the correct behavior of log servers. In conclusion, transparency schemes do not rely on trust in a single node. Although they cannot prevent the emergence of misbehaviour, all misbehaviour can be discovered by stakeholders. The idea of transparency has been widely used in many fields, such as certificate transparency [20,33], key transparency [25,26], software supply chain transparency (Sigstore [34]), certificate revocation transparency (PKISN [35]), ticket transparency [18], and policy transparency (PoliCert [36], PoliCT [37]).

2.3. Ticket Transparency

Different from common transparency schemes, Ticket Transparency [18] (TicketT) places significant effort into privacy protection, given that SSO tickets contain privacy. Specifically, TicketT introduces privacy-preserving log servers to publicly record all IdP-issued tickets. Firstly, the ticket is blinded before being submitted to the log server, which signs and publicly records blindly signed tickets $\langle \mathcal{B}(ticket, s) \rangle_{S_L}$, where s denotes the blinding factor. Secondly, the secret blinding factor is stored along with the blindly signed ticket in logs, but encrypted using the user's IBE public key (e.g., $\mathcal{E}_A(s)$) so that only the user is able to un-blind their tickets. Thirdly, for more effective monitoring, ticket transparency uses bloom filters to generate pseudonyms for users, which are stored in public logs along with tickets. Eventually, the entry logged in the log is $\langle N_A, \langle \mathcal{B}(ticket, s) \rangle_{S_L}, \mathcal{E}_A(s) \rangle$, where N_A is the pseudonym of user A. The motivation for TicketT is laudable, but its solution has serious flaws as explained in Section 3.1.

3. T-FIM: Transparency in Federated Identity Management

This section first addresses the issue of centralized trust in FIM and provides a brief overview of the T-FIM rudiment. Next, we delve into the design goals and challenges of T-FIM, followed by an explanation of the system's threat model.

3.1. Motivation

As a method of centralized identity authentication and authorization, traditional federated identity systems have an inherent security threat of centralized trust, that is, the IdP has the ability to access as a victim any SP that trusts it. Additionally, the trust granted to the IdP is absolute and lacks proper regulation, which poses a significant challenge in identifying misbehaviour on their part. In real-world scenarios, attackers compromise the IdP to gain unauthorized access [14–16] and launch a large-scale cyber attack (i.e., SolarWinds attack [17]). The cyber attack, which led to data breaches across multiple government agencies, was executed with such stealth that it remained undetected for a duration of nine months.

In addition, in recent years, emerging technologies (e.g., cloud computing) and web applications have become essential components of modern life [38,39], but have also made it easier for criminals, which are harder to detect and prosecute than traditional crimes [30]. For example, criminals launch DDoS attacks from machines in the cloud [40], using cloud hosts to build a code-receiving platform and create massive accounts, or guide public opinion via social platforms. It can be seen that recording the time and location of users' access actions, along with the accessed application/resource, is crucial in investigating, tracking, and preventing such activities—similar to a bank statement. However, investigations and forensics across multiple information systems are extremely complex. Therefore, there is an urgent need for a concise investigation/forensics approach to obtain the login/access actions of the suspect .

The researchers came up with a solution called TicketT [18] aimed at empowering users to hold the IdP accountable for mis-issuances. However, TicketT does not devote much effort to digital forensics. More significantly, it suffers from two major flaws: (*i*) It introduces a new centralized trust node–PKG, which violates the decentralized trust principle of transparency schemes. (*ii*) Furthermore, serious, users cannot discover all misbehaviour of adversaries by monitoring public logs. Specifically, if an adversary encrypts the blinding factor for user A's ticket with user X's public key (i.e., $\langle N_A, \langle \mathcal{B}(ticket, s)\rangle_{\mathcal{S}_L}, \mathcal{E}_X(s)\rangle$), then user A will lose the ability to un-blind the entry. To this end, this scheme introduces the dispute resolution, i.e., if user A has doubts, the PKG will decrypt the random blinding factor s by the IBE master key and un-blind the blindly signed ticket. However, as a transparent scheme, it is irrational that additional avenues are required to generate suspicion.

3.2. Integrating Transparency into FIM

To effectively address the challenges at hand, we propose the integration of transparency into federated identity management, known as T-FIM. This approach not only enhances accountability but also promotes the decentralization of trust, resulting in a more secure and reliable system. In general, T-FIM introduces a privacy-preserving log that securely records all tokens issued by the IdP in a ciphertext state. These tokens are encrypted using the IBE algorithm to ensure that they can only be deciphered by specific users. Additionally, the IBE private key is generated by a DPKG.

Specifically, when a user accesses an SP, the SP redirects the request to the IdP, and the IdP authenticates the user. If the authentication is passed, the IdP will sign a token for the user, encrypt and submit it to the log server. Once the log server receives an encrypted token, it responds with a signed token timestamp (STT) as proof of token submission and merges it into the privacy-preserving log. Then, the IdP returns the token and STT to the SP, and the SP verifies them. If the verification is passed, the SP provides services or grants resources to the user.

By monitoring IdP-issued tokens in the privacy-preserving log with the IBE private key, the true identity owner is able to establish accountability for any instances of mis-issuance. In addition, the logged tokens not only benefit identity owners but also assist authorized forensic investigators in carrying out thorough investigations and holding cybercriminals accountable for their unlawful actions.

3.3. Thread Model

In general, there are six participants in T-FIM, namely IdP, SPs (doubling as auditors), users (doubling as monitors), the log server, the DPKG, and forensic investigators, as shown in Figure 1. Firstly, this scheme aims to protect the application and its users from fraudulent tokens, so we assume that SPs and users are honest. SPs strictly enforce token verification, while users diligently monitor any potentially fraudulent tokens. Secondly, another goal of the scheme is to assist investigators in digital forensics, so investigators are considered semi-honest, that is, they do not violate the agreement but attempt to obtain user privacy beyond legal authorization. Thirdly, the IdP and log servers are potentially malicious. Specifically, the IdP could issue fraudulent tokens, log into the application as any user, and try to avoid censorship. Additionally, the log server could sign the token without recording it in the public log, which would be detected by auditors undertaken by the SPs. Fourthly, the DPKG consists of a joint committee of n members. While individual members may have malicious intent, at least t of them work following the protocol simultaneously. Furthermore, collusion is possible among at most $t-1$ members. Finally, we assume that there are external snoopers who mine users' privacy from the public log.

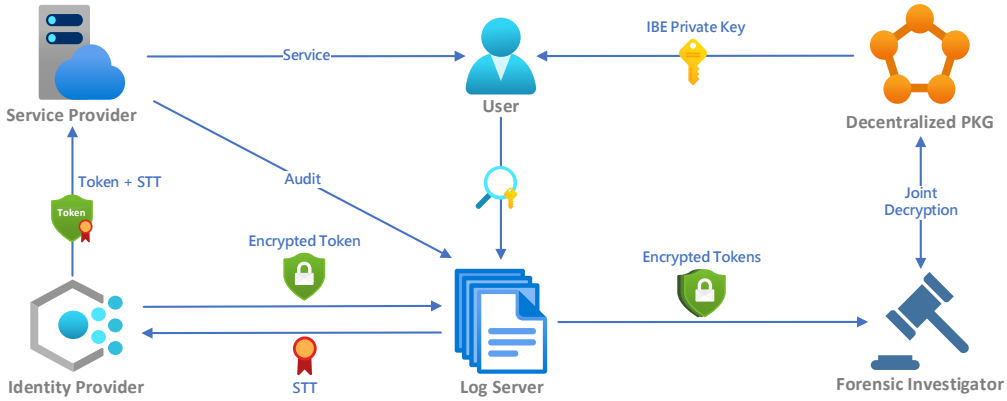

Figure 1. The framework of T-FIM.

3.4. Design Goals and Challenges

T-FIM aims to enhance the trust model and accountability of FIM by accomplishing three primary design goals. Firstly, T-FIM seeks to establish dual accountability measures: (*i*) enabling users, specifically the true identity owners, to hold the IdP accountable for any misconduct, and (*ii*) empowering authorized forensic investigators to hold malicious users (i.e., criminals and suspects) accountable by investigating the login/access activities of specific individuals. Secondly, T-FIM is able to provide strict protection of user privacy. This includes the following provisions: (*i*) personal information, application details, and private data (e.g., access time and IP address), can only be accessed by the user and authorized investigators through the privacy-preserving log; (*ii*) investigators can only access private information within the authorized scope; and (*iii*) adversaries are unable to associate multiple encrypted tokens with the same user, preserving user anonymity. Thirdly, T-FIM operates on a fully decentralized trust model. This is achieved through two approaches: (*i*) assuming roles (i.e., DPKG), that are fulfilled by distributed and redundant entities, ensuring resilience and fault tolerance, or (*ii*) establishing auditable behavior for roles (i.e., IdP and log server), allowing any stakeholder to verify that the services are being provided as specified or promised.

Achieving the three design goals (i.e., accountability, privacy, and decentralized trust) still poses certain challenges: (*i*) Compatible with transparency and privacy. In the malicious model, the IdP may violate agreements to avoid investigation by users, as mentioned in Section 3.1. Consequently, it is challenging to ensure that users can detect all misbehaviour by inspecting the privacy-preserving log. (*ii*) Decentralized master key generation. In order to uphold the fundamental principle of decentralized trust in transparency solutions, it is imperative to eradicate the centralized PKG. The commonly used solution [28,29] is to first generate the master key by a dealer, and then secret-shared it across multiple independent authorities, which jointly generate the user's IBE private key without recovering the master key. Unfortunately, with this approach, the dealer can generate any IBE private key independently. It has to be said that distributed generation of master keys is a challenging task. (*iii*) Tailor-made digital forensics. It is simple to let DPKG recover the surveyed user's IBE private key for investigators. However, this approach allows them to continue decrypting the user's encrypted tokens after investigation. Also, it is indeed tricky to revoke and renew IBE keys after investigation. It can be seen that enabling investigators to access only the tokens within the authorized scope poses a significant challenge. (*iv*) Efficient monitoring. Unlike certificates, tokens are issued much more frequently. Moreover, when faced with a mass of encrypted tokens, users have to attempt to decrypt them one by one, which is indeed inefficient. Therefore, the time cost of monitoring is worth considering.

4. Complete Design of T-FIM

This section introduces the specific workflow and algorithms of T-FIM. Table 1 provides the description of the notations.

Table 1. Summary of notations.

Symbol	Description
$\mathcal{E}_X(m)$	The result of encrypting message m with the IBE public key of X;
$\mathcal{D}_X(m)$	The result of decrypting message m with the IBE private key of X;
$\langle m \rangle_{\mathcal{S}_I}$	The message m with IdP signature
$\langle m \rangle_{\mathcal{S}_L}$	The message m with log server signature
$\mathcal{V}(\langle m \rangle_{\mathcal{S}})$	The verification algorithm whether $\mathcal{S}$ is the correct signature of message m.
P_i	The each member of DPKG distributedly
pk	The master private key
sk_i	The partial private key of each member P_i
pk_i	The partial public key of each member P_i
pk	The system master public key
$\mathcal{H}_1$	The Hash function $\mathcal{H}_1$: $\{0,1\}^* \rightarrow \mathbb{G}_1$ in IBE
$\mathcal{H}_2$	The Hash function $\mathcal{H}_2$: $\mathbb{G}_T \rightarrow \{0,1\}^n$ in IBE

4.1. Basic Concept

Decentralized private key generation. We introduce a DPKG to generate keys jointly. Each member of the DPKG generates a part of the user's IBE private key and sends it directly to the user. When the user receives more than the threshold partial IBE private key, it can assemble the complete IBE private key, see Section 4.2 for details.

Compatible with transparency and privacy. In the privacy-preserving log, the logged-encrypted tokens are transparent to their true owner, that is, only the true owner can decrypt them and check them further. To this end, we use IBE to encrypt tokens, but do not directly use IBE as a black box. In order to ensure that each token can be decrypted by its true owner, the SP additionally verifies the binding relationship between the logged-encrypted token and the identity in the token, see Section 4.3 for details.

Efficient monitoring. In T-FIM, the privacy-preserving log is sharded according to user identities. Users only need to monitor a certain log sharding, which effectively reduces the monitoring cost, see Section 4.4 for details. Futhermore, T-FIM employ the most efficient pairing-friendly elliptic curve that conforms to the 128-bit security level (i.e., *BLS12-381*) for system implementation.

Privacy-preserving digital forensics. When an investigator is legally authorized, DPKG jointly decrypts the tokens within the scope of authorization without knowing the final decryption result. Only the investigator can obtain the decrypted token, as shown in Section 4.5.

4.2. System Initialization

T-FIM introduces a committee of n members (denoted by $[n]$) to take on the role of PKG–DPKG, where any t members (denoted by $[t]$) function properly to fulfill the functions (i.e., extract IBE private keys and digital forensics). The system initialization involves the generation of the system master public key and the users' IBE private key. In addition, each participant in the system presets the master public key. We borrow from the threshold ECDSA algorithm [41] to design DPKG, and Figure 2 helps to quickly understand DPKG.

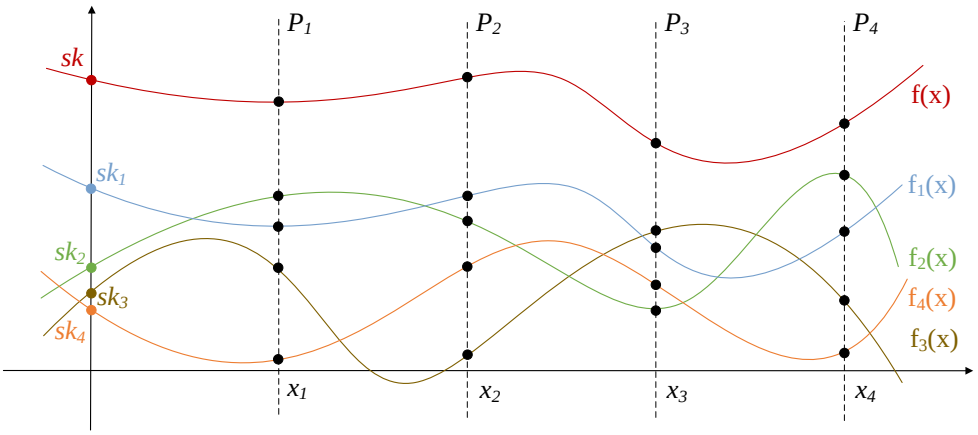

Figure 2. The schematic diagram of functions and key parameters for the initialization process of DPKG with 4 members.

4.2.1. DPKG Setup

At this stage, members of DPKG distributedly generate the system master public key (steps 1 and 2), and make preparations for IBE private key extraction (steps 3 to 5). Thus, secret sharing is performed on $\mathbb{Z}_q^*$. The process of DPKG Setup is shown in Algorithm 1.

Algorithm 1: t-of-n DPKG Setup

Data: n is the party count, t is the threshold size

1 Each member, denoted as P_i, calculate the partial private key sk_i by sampling a polynomial $f_i(x)$ of degree $t-1$ whose constant term is the partial private key.The corresponding partial public key is $pk_i = [g, h_i = g^{sk_i}]$, and the master private key is $sk = \sum_{i\in[n]} sk_i$.

2 Each member P_i obtain the partial public key of other system participants, calculate and then store the system master public key ($pk = [g, h = \prod_{i\in[n]} h_i]$).

3 Each member P_i discloses the selected abscissa x_i ($x_i \neq 0$).

4 For all pairs of parties P_i and P_j, P_i sends the partial share $f_i(x_j)$ to P_j and receives $f_j(x_i)$ in return.

5 Each member P_i computes its point $f(x_i) = \sum_{j\in[n]} f_j(x_i)$.

4.2.2. Withdrawal and Enrollment of Members

When less than $n-t$ members in DPKG actively or are forced to withdraw, the remaining members can still perform the duties of DPKG. However, the DPKG still needs new members to fill vacancies. We assume that member P_{lapsed} has lapsed from DPKG, and the specific process of new member P_{new} joining DPKG. The process is illustrated in Algorithm 2.

Algorithm 2: Withdrawal and Enrollment of Members

Data: n is the party count, t is the threshold size, P_{lapsed} is the member that has lapsed from DPKG, P_{new} is the new member joining DPKG

1 Member P_{new} selects t members to send a join request.

2 Each member P_i sends $f_{lapsed}(x_i)$ to P_{new}.

3 Member P_{new} calculates $sk_{new} = sk_{lapsed} = \sum_{i\in[t]} f_{lapsed}(x_i) \cdot \prod_{j\in[t],j\neq i} \frac{x_j}{x_j - x_i}$.

4 All n members perform DPKG setup based on existing partial private keys.

4.2.3. IBE Private Key Extraction

This subsection introduces how t members generate the IBE private key (e.g., $\mathcal{H}_1(A)^{sk}$) without recovering sk, as demonstrated in Algorithm 3.

Algorithm 3: IBE Private Key Extraction

Data: n is the party count, t is the threshold size, sk is the master private key, $pk : [g, h]$ is the master public key
Output: IBE private key

Each member P_i computes its share $s_i = f(x_i) \cdot \prod_{j \in [t], j \neq i} \frac{x_j}{x_j - x_i}$
Each member P_i computes the partial IBE private key $sk_i(A) = \mathcal{H}_1(A)^{s_i}$.
User A calculates its complete private key (i.e., $sk(A) = \prod_{i \in [t]} sk_i(A)$) after obtaining the partial private key for t members
User A verifies the correctness of the private key:
if $\hat{e}(sk(A), g) == \hat{e}(\mathcal{H}_1(A), h)$ **then**
    the private key is correct
    return IBE private key
end

4.3. Authentication and Authorization

Users in T-FIM login applications or access resources through an enhanced authentication/authorization protocol. The full protocol is shown in Figure 3, which is roughly divided into three parts as follows: token issuance, token logging and token verification. This work modifies the protocol flow of SAML 2.0, and this modification is also potentially applicable to other authentication/authorization protocols (e.g., OIDC and OAuth 2.0).

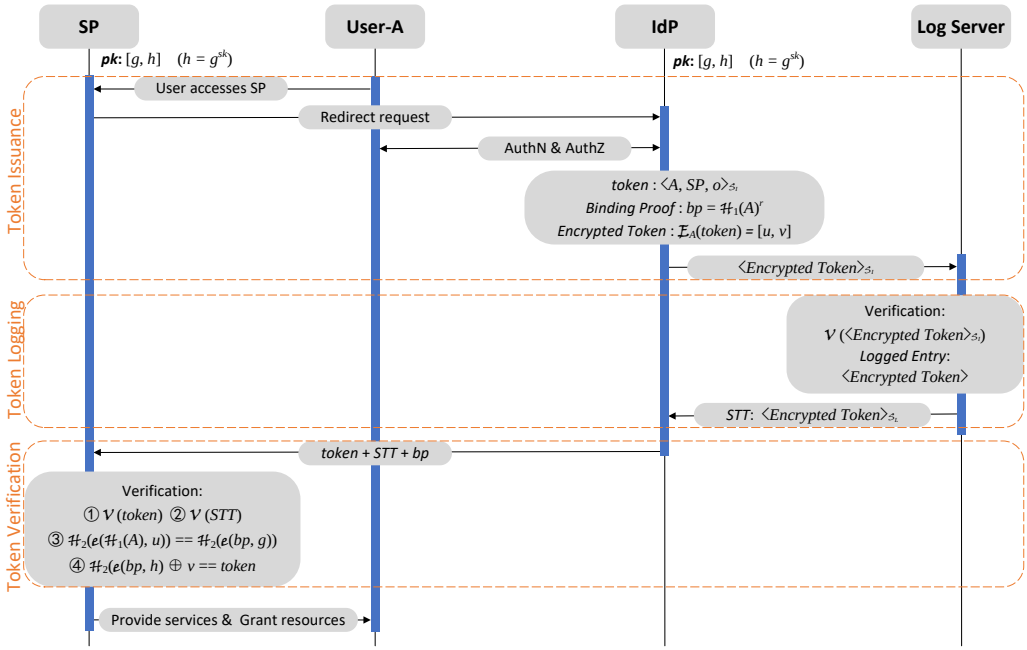

Figure 3. The full authentication/authorization protocol in T-FIM.

4.3.1. Token Issuance

Algorithm 4 illustrates the process of token issuance. Firstly, user A initiates an access request to the SP. Secondly, the SP redirects the request to the IdP. Thirdly, the IdP performs the following: (*i*) It completes the authentication for user A. (*ii*) It generates a

$token = \langle A, SP, o \rangle_{S_I}$ for user A. (*iii*) It generates a random number $r \in \mathbb{Z}_q^*$ and uses it to encrypt the token to form $\mathcal{E}_A(token) = [g^r, \mathcal{H}_2(\hat{e}(\mathcal{H}_1(A), h)^r) \oplus token]$ (also referred to as $[u, v]$). (*iv*) It maps the user ID A to the log ID (i.e., $\mathcal{M}(A) = SHA256(A) \ mod \ n$, n is the number of logs), and sends the encrypted token with signature (i.e., $\langle \mathcal{E}_A(token) \rangle_{S_I}$) to the corresponding log server.

Algorithm 4: Token Issuance

Input: A is the user A information, SP is the service provider's information, o is other authorization and/or authorization information

Output: $\langle \mathcal{E}_A(token) \rangle_{S_I}$ is the encrypted token with IdP's signature

generate $token = \langle A, SP, o \rangle_{S_I}$ for user A and IdP sign the token with a signature
generate a random number $r \in \mathbb{Z}_q^*$
encrypt the token $\mathcal{E}_A(token) = [g^r, \mathcal{H}_2(\hat{e}(\mathcal{H}_1(A), h)^r) \oplus token]$
sign the encrypted token $\langle \mathcal{E}_A(token) \rangle_{S_I}$
return $\langle \mathcal{E}_A(token) \rangle_{S_I}$

4.3.2. Token Logging

Firstly, the log server verifies the signature of the IdP and public records the encrypted token (also called logged entry). Secondly, it responds to the IdP by generating an STT (i.e., $\langle \mathcal{E}_A(token) \rangle_{S_L}$) for the encrypted token as proof that it is publicly recorded.

4.3.3. Token Verification

Algorithm 5 shows the process of token verification. Firstly, the IdP returns $\langle token, STT, bp \rangle$ to the SP, where the bp (i.e., $\mathcal{H}_1(A)^r$) is the binding proof of the logged entry and user ID. Secondly, the SP does three-part verification: (*i*) It verifies the IdP's signature on the token. (*ii*) It verifies the corresponding log server's signature on the STT. (*iii*) It verifies the binding relationship, specifically, if $\mathcal{H}_2(\hat{e}(\mathcal{H}_1(A), u)) == \mathcal{H}_2(\hat{e}(bp, g))$ and $\mathcal{H}_2(\hat{e}(bp, h)) == token \oplus v$, the verification is passed, which means that user A is certain to decrypt the token (Section 5.2 provides relevant security analysis). Finally, the SP provides corresponding services for user A.

Algorithm 5: Token Verification

Input: $token = \langle A, SP, o \rangle_{S_I}$ is the token contain authorization and/or authentication information, STT is the encrypted token with log server's signature $\langle \mathcal{E}_A(token) \rangle_{S_L}$ as proof that it is publicly recorded, bp is the binding proof of the logged entry and user ID.

Output: The verification result.

if *verify the IdP's signature of token* $= \langle A, SP, o \rangle_{S_I}$ *successfully* **then**
  if *verify log server's signature on the STT successfully* **then**
    if *verify binding relationship:*$\mathcal{H}_2(\hat{e}(\mathcal{H}_1(A), u)) == \mathcal{H}_2(\hat{e}(bp, g))$ *and* $\mathcal{H}_2(\hat{e}(bp, h)) == token \oplus v$ **then**
      return true and SP provide corresponding services for user A
    else
      return verification false
    end
  else
    return verification false
  end
else
  return verification false
end

4.4. Mis-Issuance Detection

This subsection illustrates the workflow of two crucial roles (i.e., the log server and the monitor) for detecting mis-issuance by the IdP.

4.4.1. Log Server

First, each log sharding has a unique ID and an independent public-private key pair, and only records tokens of partial users. In addition, the logs organize the submitted encrypted tokens into a Merkle hash tree to ensure immutability and continuous integrity.

4.4.2. Monitor

As a monitor, the user A only monitors the log sharding whose ID is $\mathcal{M}(A)$. Specifically, user A first downloads all logged entries (i.e., $[u, v]$) and then tries to decrypt them one by one using its IBE private key, i.e., $token = \mathcal{H}_2(\hat{e}(sk(A), u)) \oplus v$. If a token is decrypted successfully, user A further checks whether it corresponds to one of their real access behaviours. Sections 5.4 and 6.1 provide relevant evaluations of privacy and performance.

4.5. Digital Forensics

When the investigator obtains a search warrant, it can request DPKG to view tokens related to a specific user during a particular timeframe. Specifically, the investigator initiates a joint decryption request for specified encrypted tokens to t members, as follows:

(1) The investigator initiates a digital forensics request to t DPKG members with the identity of the suspect (e.g., user A).
(2) Members obtain encrypted tokens of a specified period from log servers.
(3) For each logged $entry = [u, v]$, each member P_i calculates the partial decryption value $\widetilde{\mathcal{D}}_A(entry)_i = \hat{e}(\mathcal{H}_1(A)^{s_i}, u)$.
(4) After receiving t partial decrypted values, for each encrypted token, the investigator calculates $token = \mathcal{H}_2(\prod_{i\in[t]} \widetilde{\mathcal{D}}_A(entry)_i) \oplus v$. When the identity bound in the logged entry is consistent with the identity of the suspect (i.e., $entry == \mathcal{E}_A(token)$), the token is successfully decrypted.

5. Analysis of Correctness, Security, and Privacy

In this section, we first illustrate the correctness and security of system initialization, which ensures the correct execution of subsequent processes. In addition, we systematically analyze the correctness and security of two target functions, namely, mis-issuance detection and digital forensics. Finally, we demonstrate that the use of T-FIM does not introduce any additional privacy breaches.

5.1. System Initialization

Correctness. The selected x_i of each member P_i is public, so any member can independently calculate its partial private key (i.e., $sk_i(A)$) after the exchange of $f_j(x_i)$. According to the Lagrangian interpolation formula (i.e., Formula (1)), user A can construct the IBE private key $\mathcal{H}_1(A)^{sk}$ after collecting partial private keys exceeding the threshold t, as derived from Derivation (2).

$$sk = \sum_{i\in[t]} f(x_i) \prod_{j\in[t], j\neq i} \frac{x_j}{x_j - x_i} \tag{1}$$

$$\prod_{i\in[t]} sk_i(A) = \prod_{i\in[t]} \mathcal{H}_1(A)^{s_i} = \mathcal{H}_1(A)^{\sum_{i\in[t]} s_i} = \mathcal{H}_1(A)^{\sum_{i\in[t]} f(x_i) \prod_{j\in[t], j\neq i} \frac{x_j}{x_j - x_i}} = \mathcal{H}_1(A)^{sk} \tag{2}$$

Security. Firstly, Jack et al. [41] have reduced the security of the algorithm to the difficulty of solving the Computational Diffie-Hellman problem in $\mathbb{G}_1/\mathbb{G}_2$. Secondly, in the system initialization, the DPKG determines and discloses the master public key (i.e., $pk = [g, h]$).

Therefore, $\mathcal{H}_1(A)$, g, and h in Equation (3) are all known to user A, and the $sk(A)$ generated *in violation of the protocol* will be detected by the user A.

$$\mathcal{H}_2(\hat{e}(sk(A), g)) == \mathcal{H}_2(\hat{e}(\mathcal{H}_1(A), h)) \tag{3}$$

5.2. Mis-Issuance Detection

Correctness and Security. If the IdP *follows the protocol* to encrypt and submit all issued tokens, a user holding the correct IBE private key can obviously decrypt and check the tokens related to himself correctly. This subsection provides direct proof of the security and unforgeability of T-FIM. It demonstrates that any misbehaviour by the IdP will be detected by users through their monitoring of the privacy-preserving logs.

Theorem 1. *If an SP follows the protocol correctly, encrypted tokens are searchable to their true owners, where searchable means that the owner can distinguish and decrypt the tokens associated with himself in the privacy-preserving log.*

Proof. In T-FIM, the SP performs a series of verifications before accepting a token. It first verifies the IdP's signature on the token and the log server's signature on the STT, where the log ID of the log server is $\mathcal{M}(A)$ (monitored by user A). Moreover, there are auditors to force that the tokens with an STT are merged into the Merkle tree operated by the log server. The two measures are adopted to guarantee that the tokens accepted by SP are issued by IdP and logged in the scope of related users' monitoring. Besides, for the encrypted token $\mathcal{E}_A(token) = [u, v]$, the SP verifies that the Equations (4) and (5) hold. From the equations, the knowledge of SP is g, h, $token$, and $\mathcal{H}_1(A)$.

$$\mathcal{H}_2(\hat{e}(\mathcal{H}_1(A), u)) == \mathcal{H}_2(\hat{e}(bp, g)) \tag{4}$$

$$\mathcal{H}_2(\hat{e}(bp, h)) \oplus vs. == token \tag{5}$$

A malicious IdP tries to modify the encrypted token and the bp to create a fraudulent token that can pass the SP verification and go undetected by the true owner. Suppose the constructed encrypted token $[u, v] = [g^{r_1}, \mathcal{H}_2(\hat{e}(\mathcal{H}_1(A), h)^{r_2}) \oplus token']$, the $bp = \mathcal{H}_1(A)^{r_3}$, where r_1, r_2, r_3 can be assigned as any legal values. In order to pass the verification of SP, the constructed encrypted token and the bp are subject to Equations (4) and (5). From the Derivation (6) and (7), we can conclude that if the encrypted token passes the verification of SP, then the true owner can correctly decrypt the encrypted token.

$$\begin{aligned} & \hat{e}(\mathcal{H}_1(A), u) = \hat{e}(bp, g) \\ \Rightarrow\ & \hat{e}(\mathcal{H}_1(A), g)^{r_1 - r_3} = 1_{\mathbb{G}_T} \\ \Rightarrow\ & \hat{e}(\mathcal{H}_1(A), g)^{sk \cdot r_1} = \hat{e}(\mathcal{H}_1(A), g)^{sk \cdot r_3} \end{aligned} \tag{6}$$

$$\begin{aligned} & \mathcal{H}_2(\hat{e}(bp, h)) \oplus vs. = token \\ \Rightarrow\ & \mathcal{H}_2(\hat{e}(\mathcal{H}_1(A), g)^{sk \cdot r_1}) \oplus v = token \\ \Rightarrow\ & \mathcal{H}_2(\hat{e}(sk(A), u)) \oplus v = token \end{aligned} \tag{7}$$

□

5.3. Digital Forensics

Correctness. From Derivation (8), it can be seen that the semi-honest investigator can correctly decrypt the logged entries with the assistance of t protocol-compliant DPKG members. Therefore, the investigator has a panoramic view of any activity of the suspect during a particular timeframe.

$$token = \mathcal{H}_2(\prod_{i \in [t]} \widetilde{\mathcal{D}}_A(entry)_i) \oplus vs. = \mathcal{H}_2(\prod_{i \in [t]} \hat{e}(\mathcal{H}_1(A)^{s_i}, u)) \oplus vs. = \mathcal{H}_2(\hat{e}(sk(A), u)) \oplus v \tag{8}$$

Security. Firstly, in T-FIM, users do not participate in the encryption and submission of tokens, so a suspect cannot hide a login/access activity. Secondly, malicious DPKG

members have the ability to subvert joint decryption. Although some expensive zero-knowledge techniques are potentially able to verify that the decryption result is well-formed [41], T-FIM does not detect such malicious behaviour for the sake of efficiency. Thirdly, since the user could repudiate a token used and blame the IdP for mis-issuing it, the evidence obtained by investigators from the log servers can only guarantee integrity. Nevertheless, it can still serve as one of critical evidence.

5.4. Analysis of Privacy

T-FIM can be considered as an enhancement of the FIM, compared with traditional FIM, the additional information leakage in T-FIM was the information logged in privacy-preserving logs and the binding proof for SPs.

Theorem 2. *Non-repudiation: Assuming a random oracle, if the number of curious DPGK members does not exceed the threshold, then the adversary cannot obtain any information about the encryption tokens from the ciphertext recorded in log servers.*

Assumption 1. *The adversary-i (i.e., non-true owners, snoopers, and log servers) can obtain public information (i.e., the master public key and logged entries).*

Proof. In T-FIM, the logged entries are encrypted tokens (e.g., $\langle \mathcal{E}_A(token) \rangle_{\mathcal{S}_L}$). Since the IBE has been proven to be semantically secure in the random oracle model [32], *adversary-i* cannot obtain any information about the encrypted tokens. □

Assumption 2. *The adversary-ii (i.e., SPs) can obtain public information, some tokens, and the corresponding binding proof.*

Proof. Suppose the *adversary-ii* holds a binding proof $bp = \mathcal{H}_1(A)^r$ of a token, and the challenge ciphertext is $\mathcal{E}_{A'}(token') = [u', v'] = [g^{r'}, \mathcal{H}_2(\hat{e}(\mathcal{H}_1(A'), h)^{r'}) \oplus token']$. Since $\mathcal{H}_1$ maps the user ID to a prime order group, $\mathcal{H}_1(A')^{r'}$ can be represented as $\mathcal{H}_1(A)^{r''}$, and the v' can be represented as $\mathcal{H}_2(\hat{e}(\mathcal{H}_1(A), h)^{r''}) \oplus token'$. The only secret information contained in bp is r. If r cannot provide additional information about the $\mathcal{E}_{A'}(token')$, we can conclude that the $\mathcal{H}_1(A)^r$ not affect the privacy of $token'$ and the ability of *adversary-ii* is equal to *adversary-i*. Since r is randomly chosen by IdP, r and r'' are independent of each other, which means that r can be considered random and provides no additional information about the r'' in the challenge. Furthermore, the IBE scheme is semantically secure and each encryption is independent. Therefore, we can conclude that *adversary-ii* cannot obtain more information about the unmastered token than *adversary-i*. □

Assumption 3. *The adversary-iii (i.e., curious DPKG members and investigators) can obtain public information and some (partial) decrypted tokens.*

Proof. Compared to *adversary-i*, *adversary-iii* has the ability to (partially) decrypt tokens, which must be recorded in a log server. The secret s and λ^s have been proven to be privacy against less than the threshold number of curious members, where λ is on the elliptic curve group [41,42]. Extracting the IBE private key is equivalent to recovering the secret ID^{sk} where $ID \in \mathbb{G}_1$, and joint decryption is equivalent to recovering C^{sk} where $C \in \mathbb{G}_T$. Therefore, it can be concluded that the users' IBE private keys and encrypted tokens are privacy against less than the threshold number of curious DPKG members. In general, *adversary-iii* does not have stronger capabilities than *adversary-i* for tokens that are not authorized to be viewed. □

6. Experiments and Evaluation

We performed experimental measurements of the cost of applying T-FIM. All experiments were conducted on an ASUSTeK desktop computer with an Intel(R) Core(TM)

i5-6300HQ CPU @ 2.30 GHz, 32 KB L1 cache, and 4 GB RAM. The operating system is Ubuntu 18.04 with Linux kernel version 4.15.0.140, and the network bandwidth is 5 Mbps. In addition, we realized T-FIM based on bilinear pairing constructed from *BLS12-381* implemented using the open source code gnark-crypto [43]. Furthermore, as described in the token sample in [44], a test set of 1M tokens belonging to 100K different users is randomly generated and encoded in JWT format [45] (i.e., `HEADER.PAYLOAD.SIGNATURE`).

6.1. Additional Overhead

This subsection analyzes the additional communication costs and computational costs generated by T-FIM compared with traditional FIM at different stages, as shown in Table 2. Firstly, some basic operations are labelled, M indicates the cost of running a multiplication on $\mathbb{G}_T$, P for the power operation on $\mathbb{G}_1$, and B for the bilinear mapping (the bitwise XOR is omitted). Secondly, $\mathcal{C}$ represents the cost of a round communication (The specific time cost of network communication depends on various factors, such as the network bandwidth used, geographical location, and the protocol used. Lee et al. [46] measured the average round-trip time from eight different regions to TLS 1.2/1.3 web servers. Their results showed that TLS 1.3 took 97 ms, while TLS 1.2 took 341 ms on average), and $\mathcal{T}$ represents the cost of transmitting/downloading an encrypted token. Finally, $\mathcal{E}$ means the cost of performing an IBE encryption, $\mathcal{D}$ for the IBE decryption, $\mathcal{S}$ for the digital signature, and $\mathcal{V}$ for signature verification (the hash operation $\mathcal{H}_1$ and $\mathcal{H}_2$ is omitted). Table 3 shows the time cost of basic operations and cryptographic operations.

Table 2. Additional overhead for each stage of T-FIM.

	Login/Access		**Monitoring**		**Digital forensics**	
	$\rightleftharpoons$	$\trianglerighteq$	$\rightleftharpoons$	$\trianglerighteq$	$\rightleftharpoons$	$\trianglerighteq$
IdP	$\mathcal{C}^{\gamma}$	$P+\mathcal{E}+\mathcal{S}$	-	-	-	-
SP	-	$3B+\mathcal{V}$	-	-	-	-
Log Server	-	$\mathcal{S}+\mathcal{V}$	-	-	-	-
Monitor	-	-	$m\mathcal{T}^{\gamma}$	$m\mathcal{D}$	-	-
DPKG [N]	-	-	-	-	$m\mathcal{T}^{\gamma}$	$P+m\mathcal{D}$
Investigator	-	-	-	-	$mt\mathcal{T}^{\P}$	mtM
Time cost	$\mathcal{C}^{\gamma}$	+16.99 ms	2.2 ms		$t \times$ 2.2 ms $+ t \times$ 7.7 µs	

[N]: The table shows the cost of each member of DPKG. $\rightleftharpoons$ indicates network communication overhead. $\trianglerighteq$ represents computational overhead. Network communication is two-way, but only one side of the communication is shown. $^{\gamma}$ indicates communication with log servers. ¶ indicates communication with DPKG. m indicates the number of tokens involved.

System Initialization. Since system initialization is a one-time preparatory work before protocol execution, it is not time-sensitive. Therefore, this graph presents the theoretical communication overhead of the initialization phase, while the computational overhead (a fraction to a few milliseconds) is negligible. In the DPKG setup phase, each DPKG member conducts $2(n-1)$ round communication to exchange partial shares. Whenever a new member enrollment, the new member needs to perform $t+2(n-1)$ round communication, while other members need $2(n-1)$ times. In addition, when extracting IBE private key, a user needs to communicate with each of the t members once.

Login/Access Process. Compared with the login/access process based on standard protocols, T-FIM introduces 1 additional round of communications between IdP and the log server. In addition, computing the binding proof and encrypting/signing the token takes about 7.64 ms from the IdP, and about 6.04 ms from the log server to verify the signature and issue an STT. In addition, T-FIM introduces additional signature verification and binding verification to the SP, which takes about 3.31 ms. In general, for users, the login/access process takes an extra round of communication time plus 16.99 ms.

Monitoring. We encrypt the 1M tokens in the test set, and the average size of the encrypted tokens is 1.38 KB. According to the measured data, it is calculated that on

average, the user (with a network bandwidth of 5Mbps) can download a token every 2.2 ms. Furthermore, we arbitrarily select a user's IBE private key to decrypt the 1 M tokens, which requires a total of 1598 s. That is, the average time to decrypt a single token is 1.6 ms, which means that 37.5 K tokens can be decrypted per minute. When downloading and decrypting tokens are performed in parallel, the efficiency of monitoring is limited by the speed of downloading tokens. As a result, it takes approximately 2.2 ms to check a single token, allowing for a maximum of 27 K tokens to be checked per minute.

Digital Forensics. First, each participating DPKG member downloads and partially decrypts tokens with almost the same cost as a user monitoring them. After that, the investigator spends $t \times 2.2$ ms to download partial decryption values of each encrypted token from t DPKG members, and then synthesizes the full decrypted value locally at an average computational cost of $t \times 7.7\mu$s.

Table 3. The time cost of basic operations and cryptographic operations.

	Basic Operations			**IBE Scheme**		**RSA Signature** [§]		**Blind Signature** [§]	
	$\mathcal{M}$	$\mathcal{P}$	$\mathcal{B}$	$\mathcal{E}$	$\mathcal{D}$	$\mathcal{S}$	$\mathcal{V}$	**Blind**	**Un-Blind**
Time [†]	7.7 µs	0.14 ms	0.93 ms	1.6 ms	0.94 ms	5.9 ms	0.14 ms	0.28 ms	0.011 ms

[†]: It shows the average time for 1M operations on 1M tokens in the test set. [§]: The test uses the 128-bit security level algorithm (i.e., *RSA-3072*) in RSA Blind [47].

Log Sharding vs. Monitoring Efficiency. We insert 100 K randomly generated user IDs into different numbers of log shards and count the distribution. It can be seen from Figure 4 that users are almost evenly distributed in each log sharding. Therefore, the cost of monitoring and forensics is reduced to about $1/n$ of that before sharding, where n is the number of shards.

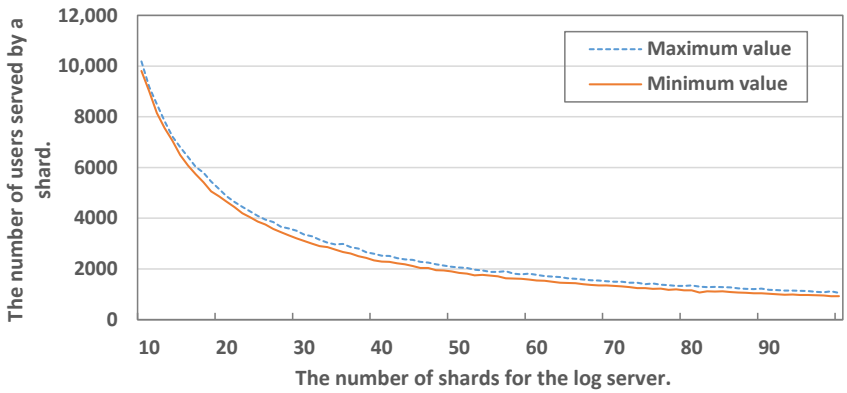

Figure 4. The trend of monitoring costs for users when log servers are sharded into different numbers.

6.2. Comparison with Ticket Transparency

This subsection provides a comparison between T-FIM and TicketT in terms of transparency, decentralized trust, and performance as shown in Table 4.

Transparency. In TicketT, the adversary can evade the user's censorship by encrypting the blinding factor with the IBE private key of an unreal user, see Section 3.1 for details. However, in T-FIM, the binding verification of SP ensures that the encrypted token is searchable to its true owner.

Decentralized Trust. TicketT introduces a trusted PKG to generate IBE private key for users, that is, PKG is omniscient about the entries in the privacy-preserving log. However, T-FIM uses a distributed IBE private key generation method, eliminating centralized trust.

Performance. Firstly, when issuing tokens, the consumption of the two schemes is almost equal. However, in terms of token verification time, TicketT outperforms T-FIM with a verification time of 0.29 ms, compared to T-FIM's 3.31 ms. As a result, TicketT demonstrates slightly faster authentication/authorization processes. Secondly, the $\langle \mathcal{E}_X(token) \rangle$ (about

1.38 KB) logged by the log server in T-FIM is smaller than the $\langle N_A, \langle \mathcal{B}(token, s)\rangle_{\mathcal{S}_L}, \mathcal{E}_X(s)\rangle$ (at least 2KB) in TicketT. Thirdly, the number of entries required to be downloaded for monitoring purposes is reduced to $1/n$ in T-FIM compared to TicketT, where n represents the number of log shards. Consequently, T-FIM offers improved efficiency in user monitoring. In general, T-FIM is more optimal in terms of user monitoring efficiency.

Table 4. Comparison of TicketT and T-FIM in terms of properties and performance.

		TicketT	T-FIM
Property	Transparency	⊘	√
	Decentralized	⊘	√
Performance	Issuance	$Blind + \mathcal{E} + 2\mathcal{S} + \mathcal{V} + \mathcal{C}$	$\mathcal{E} + 2\mathcal{S} + \mathcal{V} + \mathcal{C} + P$
	Verification	$2\mathcal{V}$ + Un-blind	$4\mathcal{H} + 3\mathcal{B} + 2\mathcal{V}$
	Log Entry	$\langle N_A, \langle \mathcal{B}(token, s)\rangle_{\mathcal{S}_L}, \mathcal{E}_X(s)\rangle$	$\langle \mathcal{E}_X(token)\rangle$
	Monitoring	$m\mathcal{T}' + k\mathcal{D}$	$k\mathcal{T} + k\mathcal{D}$

Green markers indicate the better-performing side and yellow marks indicate comparable performance on both sides. $\mathcal{T}'$ indicates the cost of downloading a logged entry in the TicketT (i.e., $\langle N_A, \langle \mathcal{B}(token, s)\rangle_{\mathcal{S}_L}, \mathcal{E}_X(s)\rangle$). m is the total number of tokens generated in the system over a period of time. k is the number of encrypted tokens that are indistinguishable without holding the IBE private key.

7. Related Work

7.1. Identity Management

Authentication and Authorization. FIM facilitates the implementation of authentication and authorization protocols for applications spanning multiple domains or organizations, by leveraging standardized protocols (e.g., OAuth 2.0, OIDC, and SAML). Many researchers have studied these authentication or authorization protocols at the protocol and implementation level. Chen et al. [48] examined the differences in the OAuth protocol for mobile applications and web applications. Fett et al. [49,50] conducted a formal analysis of OAuth 2.0 and OIDC based on a comprehensive and expressive web model, including attacks from malicious RPs and IdPs as well as corrupted browsers/users. Secondly, several studies focus on the vulnerabilities of systems implementations in the wild, including OAuth/OIDC [51–55], and SAML [56]. Additionally, several automated tools [57–59] for detecting the implementation vulnerabilities are designed and implemented. These studies primarily focus on analyzing the security implications of protocol flows and detecting vulnerabilities. They aim to identify potential threats such as attackers signing into Relying Parties (RPs) on behalf of other users or unauthorized disclosure of private information. However, these studies do not propose a viable solution to address the security challenges associated with authentication or authorization protocols in the Federated Identity Management (FIM) system. Recently some solutions for the security and privacy of authentication and authorization protocols have emerged. Chu et al. came up with TicketT aimed at enhancing the accountability of SSO systems. Singh et al. [60] proposed augmentation of architectural design by incorporating additional features which have the potential to increase the overall security effectiveness of the protocol. Sucasas et al. [61] enhance OAuth 2.0 protocol privacy by integrating a pseudonym-based signature scheme and a signature delegation scheme. However, these security enhancement schemes do not offer a secure solution to the issue of trust in the IdP.

Self-sovereign Identity. Various schemes for self-sovereign identity (SSI) have been proposed to enhance users' control over their identifiers. Examples of these schemes include uPort [22], Sovrin [23], and Identity Overlay Network (ION) [24]. These approaches enable permanent, resolvable, cryptographically verifiable, and decentralized identities. The SSI schemes rely on public permissionless/permissioned blockchain (distributed ledger) as the trust anchor, and utilize W3C-standardized decentralized identifiers (DIDs) [62] to represent users' identities. To demonstrate the sovereignty of DIDs, users must prove their ownership of the corresponding private keys. In the SSI system, the private key is everything, as anyone who holds the correct private key can rightfully claim ownership of the DID. Therefore, the private keys are managed by individual users. However, a large

number of security incidents [63] indicate that even technology companies engaged in security-related industries are still vulnerable to key leakage. The personal management key poses a more significant security risk. However, our solution does not introduce such risks, while still maintaining a high level of accountability and control.

7.2. Digital Forensics

Digital forensic investigators reveal the truth of an event by discovering and exposing the remnants (footprints or artifacts) of the event left on information systems. Several researchers reviewed the techniques and challenges of digital forensics in cloud [64,65] and IoT [66] environments. Some improvements to traceability, integrity, and/or privacy in digital forensics were proposed based on technologies such as blockchain [30,67,68] and IBE [69]. Moreover, some researchers focused on the interpretability of evidence. Shalaginov [70] presented novel improvements to the Neuro-Fuzzy architecture and corresponding results, which enhance the ability to learn an understandable and precise fuzzy rule-based model. Amato et al. [71] proposed a new methodology to support investigators during the analysis process, correlating evidence found through different forensics tools.

8. Conclusions

In this paper, we present the concept of transparency integrated into Federated Identity Management (FIM) to introduce the supervisory capability of the Identity Provider (IdP), referred to as T-FIM. T-FIM effectively eliminates security threats associated with a centralized trust model and establishes accountability by holding the user responsible for IdP actions and enabling forensic investigators to be accountable to users. We provide a systematic proof of correctness, security, and privacy for T-FIM within a malicious model. To showcase the practicality of T-FIM, we conduct experimental demonstrations to illustrate its feasibility for real-world applications. We also provide a comparison between T-FIM and TicketT, which demonstrates that T-FIM is more optimal in terms of user-monitoring efficiency. Specifically, while TicketT outperforms T-FIM with a verification time of 0.29 ms for token verification, T-FIM offers a smaller token size (about 1.38 KB) logged by the log server and reduces the number of entries required to be downloaded for monitoring purposes to 1/n when compared to TicketT.

Author Contributions: Conceptualization, B.X., A.S. and B.L.; Data curation, J.D., S.J. and L.S.; Formal analysis, B.X., Z.Z. and J.G.; Investigation, J.D.; Methodology, B.X., A.S., J.G., Z.W. and B.L.; Resources, J.D., S.J. and L.S.; Software, Z.Z.; Validation, Z.Z. and A.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key RD Plan of China grant number 2020YFB1005800, the National Natural Science Foundation of China under Grant 62002011, Youth Top Talent Support Program of Beihang University under Grant YWF-22-L-1272.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

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

Abbreviations

The following abbreviations are used in this manuscript:

FIM	Federated Identity Management
IdP	Identity Provider
SP	Service Provider
OIDC	OpenID Connect
SAML	Security Assertion Markup Language
PKI	Public Key Infrastructure
CA	Certificate Authority
CT	Certificate Transparency

IBE	Identity-based encryption
PKG	Private key generator
DPKG	Decentralized private key generator
STT	Signed token timestamp

References

1. Okta Inc. What Is Federated Identity? 2023. Available online: https://www.okta.com/identity-101/what-is-federated-identity/ (accessed on 25 July 2023).
2. Bendiab, G.; Shiaeles, S.; Boucherkha, S.; Ghita, B. FCMDT: A novel fuzzy cognitive maps dynamic trust model for cloud federated identity management. *Comput. Secur.* **2019**, *86*, 270–290.
3. Hardt, D. The OAuth 2.0 Authorization Framework. Technical Report. 2012. Available online: https://datatracker.ietf.org/doc/html/rfc6749 (accessed on 25 July 2023).
4. OpenID Foundation. OpenID Connect. 2022. Available online: https://openid.net/connect/ (accessed on 25 July 2023).
5. Hughes, J.; Maler, E. Security assertion markup language (saml) v2. 0 technical overview. *OASIS SSTC Work. Draft Sstc-Saml* **2005**, *13*, 12.
6. Goodner, M.; Hondo, M.; Nadalin, A.; McIntosh, M.; Schmidt, D. Understanding ws-Federation. 2007. Available online: http://xml.coverpages.org/UnderstandingWS-Federation20070528.pdf (accessed on 25 July 2023).
7. Google Inc. Enable Users to Sign into Apps and Authorize Apps to Use Google Services. 2023. Available online: https://developers.google.com/identity?hl=en (accessed on 25 July 2023).
8. Facebook Inc. Facebook Login Overview. 2023. Available online: https://developers.facebook.com/docs/facebook-login/overview/ (accessed on 25 July 2023).
9. GitHub Inc. Authentication with SAML Single Sign-On. 2023. Available online: https://docs.github.com/en/enterprise-cloud@latest/authentication/authenticating-with-saml-single-sign-on (accessed on 25 July 2023).
10. PayPal Inc. Integrate Log in with PayPal with Identity API. 2023. Available online: https://developer.paypal.com/docs/log-in-with-paypal/ (accessed on 25 July 2023).
11. Microsoft Inc. Add Azure Active Directory (Azure AD) as an Identity Provider for External Identities. 2023. Available online: https://learn.microsoft.com/en-us/azure/active-directory/external-identities/default-account (accessed on 25 July 2023).
12. Microsoft Inc. Active Directory Federation Services. 2023. Available online: https://learn.microsoft.com/en-us/windows-server/identity/active-directory-federation-services (accessed on 25 July 2023).
13. Microsoft Inc. Azure Active Directory (Azure AD). 2023. Available online: https://azure.microsoft.com/en-us/products/active-directory (accessed on 25 July 2023).
14. Ghasemisharif, M.; Ramesh, A.; Checkoway, S.; Kanich, C.; Polakis, J. O single {Sign-Off}, where art thou? An empirical analysis of single {Sign-On} account hijacking and session management on the web. In Proceedings of the 27th USENIX Security Symposium (USENIX Security 18), Baltimore, MD, USA, 15–17 August 2018; pp. 1475–1492.
15. Mainka, C.; Mladenov, V.; Schwenk, J. Do not trust me: Using malicious IdPs for analyzing and attacking single sign-on. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbrucken, Germany, 21–24 March 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 321–336.
16. Reiner, S. Golden SAML: Newly Discovered Attack Technique Forges Authentication to Cloud Apps. 2017. Available online: https://www.cyberark.com/resources/threat-research-blog/golden-saml-newly-discovered-attack-technique-forges-authentication-to-cloud-apps (accessed on 25 July 2023).
17. Jena, B.K. SolarWinds Attack and All The Details You Need to Know about It. 2023. Available online: https://www.simplilearn.com/tutorials/cryptography-tutorial/all-about-solarwinds-attack (accessed on 25 July 2023).
18. Chu, D.; Lin, J.; Li, F.; Zhang, X.; Wang, Q.; Liu, G. Ticket transparency: Accountable single sign-on with privacy-preserving public logs. In Proceedings of the International Conference on Security and Privacy in Communication Systems, Orlando, VA, USA, 23–25 October 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 511–531.
19. SSLMate Inc. Security Incident Report. 2018. Available online: https://sslmate.com/resources/certificate_authority_failures (accessed on 25 July 2023).
20. Laurie, B.; Langley, A.; Käsper, K. RFC6962: Certificate Transparency. 2013. Available online: https://datatracker.ietf.org/doc/html/rfc6962 (accessed on 23 August 2023).
21. Laurie, B.; Langley, A.; Messeri, E.; Stradling, R. RFC9162: Certificate Transparency Version 2.0. 2021. Available online: https://datatracker.ietf.org/doc/rfc9162/ (accessed on 23 August 2023).
22. Naik, N.; Jenkins, P. uPort open-source identity management system: An assessment of self-sovereign identity and user-centric data platform built on blockchain. In Proceedings of the 2020 IEEE International Symposium on Systems Engineering (ISSE), Vienna, Austria, 12 October–12 November 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–7.
23. Windley, P.J. Sovrin: An identity metasystem for self-sovereign identity. *Front. Blockchain* **2021**, *4*, 30. [CrossRef]
24. Tsai, H. Microsoft Identity Overlay Network. Available online: https://github.com/decentralized-identity/ion (accessed on 25 July 2023).

25. Melara, M.S.; Blankstein, A.; Bonneau, J.; Felten, E.W.; Freedman, M.J. {CONIKS}: Bringing Key Transparency to End Users. In Proceedings of the 24th {USENIX} Security Symposium ({USENIX} Security 15), Washington, DC, USA, 12–14 August 2015; pp. 383–398.
26. McMillion, B. Key Transparency 2023. Available online: https://datatracker.ietf.org/doc/draft-mcmillion-key-transparency/ (accessed on 25 July 2023).
27. Newman, Z.; Meyers, J.S.; Torres-Arias, S. Sigstore: Software signing for everybody. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, Los Angeles, CA, USA, 7–11 November 2022; pp. 2353–2367.
28. Kate, A.; Goldberg, I. Asynchronous Distributed Private-Key Generators for Identity-Based Cryptography. *Cryptology ePrint Archive* **2009**. Available online: https://eprint.iacr.org/2009/355.pdf (accessed on 25 July 2023).
29. Kate, A.; Goldberg, I. Distributed private-key generators for identity-based cryptography. In Proceedings of the Security and Cryptography for Networks: 7th International Conference, SCN 2010, Amalfi, Italy, 13–15 September 2010; Proceedings 7; Springer: Berlin/Heidelberg, Germany, 2010; pp. 436–453.
30. Li, S.; Qin, T.; Min, G. Blockchain-based digital forensics investigation framework in the internet of things and social systems. *IEEE Trans. Comput. Soc. Syst.* **2019**, *6*, 1433–1441. [CrossRef]
31. Inc, I.C. General Data Protection Regulation. 2016. Available online: https://gdpr-info.eu/ (accessed on 25 July 2023).
32. Boneh, D.; Franklin, M. Identity-based encryption from the Weil pairing. In Proceedings of the Advances in Cryptology—CRYPTO 2001: 21st Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 August 2001; Springer: Berlin/Heidelberg, Germany, 2001; pp. 213–229.
33. Transparency, C. Working Together to Detect Maliciously or Mistakenly Issued Certificates. 2021. Available online: https://certificate.transparency.dev/ (accessed on 25 July 2023).
34. Sigstore Inc. A New Standard for Signing, Verifying and Protecting Software. 2022. Available online: https://www.sigstore.dev/ (accessed on 25 July 2023).
35. Szalachowski, P.; Chuat, L.; Perrig, A. PKI safety net (PKISN): Addressing the too-big-to-be-revoked problem of the TLS ecosystem. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbrucken, Germany, 21–24 March 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 407–422.
36. Szalachowski, P.; Matsumoto, S.; Perrig, A. PoliCert: Secure and flexible TLS certificate management. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, 3–7 November 2014; pp. 406–417.
37. Sun, A.; Li, B.; Wan, H.; Wang, Q. PoliCT: Flexible Policy in Certificate Transparency Enabling Lightweight Self-monitor. In Proceedings of the International Conference on Applied Cryptography and Network Security, Kamakura, Japan, 21–24 June 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 358–377.
38. Zhang, Y.; Xu, C.; Yu, S.; Li, H.; Zhang, X. SCLPV: Secure certificateless public verification for cloud-based cyber-physical-social systems against malicious auditors. *IEEE Trans. Comput. Soc. Syst.* **2015**, *2*, 159–170. [CrossRef]
39. Li, S.; Da Xu, L.; Zhao, S. 5G Internet of Things: A survey. *J. Ind. Inf. Integr.* **2018**, *10*, 1–9.
40. Magazine, I. DDoS-ers Launch Attacks From Amazon EC2. 2014. Available online: https://www.infosecurity-magazine.com/news/ddos-ers-launch-attacks-from-amazon-ec2/ (accessed on 25 July 2023).
41. Doerner, J.; Kondi, Y.; Lee, E.; Shelat, A. Threshold ECDSA from ECDSA assumptions: The multiparty case. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), Francisco, CA, USA, 20–22 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1051–1066.
42. De Santis, A.; Desmedt, Y.; Frankel, Y.; Yung, M. How to share a function securely. In Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, Montreal, QC, Canada, 23–25 May 1994; pp. 522–533.
43. Botrel, G. Gnark-Crypto. 2023. Available online: https://github.com/ConsenSys/gnark-crypto (accessed on 25 July 2023).
44. Google Inc. Token Types. 2023. Available online: https://cloud.google.com/docs/authentication/token-types (accessed on 25 July 2023).
45. Jones, M.; Bradley, J.; Sakimura, N. Json Web Token (jwt). Technical Report. 2015. Available online: https://datatracker.ietf.org/doc/html/rfc7519 (accessed on 25 July 2023).
46. Lee, H.; Kim, D.; Kwon, Y. TLS 1.3 in practice: How TLS 1.3 contributes to the internet. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 70–79.
47. Hayes, P. RSA Blind. 2017. Available online: https://github.com/cryptoballot/rsablind (accessed on 25 July 2023).
48. Chen, E.Y.; Pei, Y.; Chen, S.; Tian, Y.; Kotcher, R.; Tague, P. Oauth demystified for mobile application developers. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, 3–7 November 2014; pp. 892–903.
49. Fett, D.; Küsters, R.; Schmitz, G. A comprehensive formal security analysis of OAuth 2.0. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 1204–1215.
50. Fett, D.; Küsters, R.; Schmitz, G. The web sso standard openid connect: In-depth formal security analysis and security guidelines. In Proceedings of the 2017 IEEE 30th Computer Security Foundations Symposium (CSF), Santa Barbara, CA, USA, 21–25 August 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 189–202.
51. Wang, R.; Chen, S.; Wang, X. Signing me onto your accounts through facebook and google: A traffic-guided security study of commercially deployed single-sign-on web services. In Proceedings of the 2012 IEEE Symposium on Security and Privacy, San Francisco, CA, USA, 20–23 May 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 365–379.

52. Sun, S.T.; Beznosov, K. The devil is in the (implementation) details: An empirical analysis of OAuth SSO systems. In Proceedings of the 2012 ACM Conference on Computer and Communications Security, Raleigh, NC, USA, 16–18 October 2012; pp. 378–390.
53. Wang, H.; Zhang, Y.; Li, J.; Liu, H.; Yang, W.; Li, B.; Gu, D. Vulnerability assessment of oauth implementations in android applications. In Proceedings of the 31st Annual Computer Security Applications Conference, Los Angeles, CA, USA, 7–11 December 2015; pp. 61–70.
54. Wang, H.; Zhang, Y.; Li, J.; Gu, D. The achilles heel of OAuth: A multi-platform study of OAuth-based authentication. In Proceedings of the 32nd Annual Conference on Computer Security Applications, Los Angeles, CA, USA, 5–8 December 2016; pp. 167–176.
55. Sharif, A.; Carbone, R.; Sciarretta, G.; Ranise, S. Best current practices for OAuth/OIDC Native Apps: A study of their adoption in popular providers and top-ranked Android clients. *J. Inf. Secur. Appl.* **2022**, *65*, 103097.
56. Somorovsky, J.; Mayer, A.; Schwenk, J.; Kampmann, M.; Jensen, M. On Breaking SAML: Be Whoever You Want to Be. In Proceedings of the USENIX Security Symposium, Bellevue, WA, USA, 8–10 August 2012; pp. 397–412.
57. Zuo, C.; Zhao, Q.; Lin, Z. Authscope: Towards automatic discovery of vulnerable authorizations in online services. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 799–813.
58. Zhou, Y.; Evans, D. SSOScan: Automated testing of web applications for Single Sign-On vulnerabilities. In Proceedings of the 23rd {USENIX} Security Symposium ({USENIX} Security 14), San Diego, CA, USA, 20–22 August 2014; pp. 495–510.
59. Al Rahat, T.; Feng, Y.; Tian, Y. Oauthlint: An empirical study on oauth bugs in android applications. In Proceedings of the 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE), San Diego, CA, USA, 10–15 November 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 293–304.
60. Singh, J.; Chaudhary, N.K. OAuth 2.0: Architectural design augmentation for mitigation of common security vulnerabilities. *J. Inf. Secur. Appl.* **2022**, *65*, 103091. [CrossRef]
61. Sucasas, V.; Mantas, G.; Althunibat, S.; Oliveira, L.; Antonopoulos, A.; Otung, I.; Rodriguez, J. A privacy-enhanced OAuth 2.0 based protocol for Smart City mobile applications. *Comput. Secur.* **2018**, *74*, 258–274. [CrossRef]
62. Sporny, M.; Longley, L.; Sabadello, M.; Reed, D.; Steele, O.; Allen, C. Decentralized Identifiers (DIDs) v1.0. 2022. Available online: https://www.w3.org/TR/did-core/ (accessed on 25 July 2023).
63. George, K. The Largest Cryptocurrency Hacks So Far. [Online]. Available online: https://www.investopedia.com/news/largest-cryptocurrency-hacks-so-far-year/ (accessed on 25 July 2023).
64. Zawoad, S.; Hasan, R. Trustworthy digital forensics in the cloud. *Computer* **2016**, *49*, 78–81. [CrossRef]
65. Vadlamudi, D.; Rao, D.T.; Vidyullatha, P.; AjasekharReddy, B. Analysis on digital forensics challenges and anti-forensics techniques in cloud computing. *Int. J. Eng. Technol* **2018**, *7*, 1072. [CrossRef]
66. Hou, J.; Li, Y.; Yu, J.; Shi, W. A survey on digital forensics in Internet of Things. *IEEE Internet Things J.* **2019**, *7*, 1–15.
67. Zhang, Y.; Wu, S.; Jin, B.; Du, J. A blockchain-based process provenance for cloud forensics. In Proceedings of the 2017 3rd IEEE International Conference on Computer and Communications (ICCC), Chengdu, China, 13–16 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 2470–2473.
68. Li, M.; Chen, Y.; Lal, C.; Conti, M.; Alazab, M.; Hu, D. Eunomia: Anonymous and secure vehicular digital forensics based on blockchain. *IEEE Trans. Dependable Secur. Comput.* **2021**, *20*, 225–241. [CrossRef]
69. Unal, D.; Al-Ali, A.; Catak, F.O.; Hammoudeh, M. A secure and efficient Internet of Things cloud encryption scheme with forensics investigation compatibility based on identity-based encryption. *Future Gener. Comput. Syst.* **2021**, *125*, 433–445. [CrossRef]
70. Shalaginov, A. Fuzzy Logic Model for Digital Forensics: A Trade-off between Accuracy, Complexity and Interpretability. In Proceedings of the IJCAI, Melbourne, Australia, 19–25 August 2017; pp. 5207–5208.
71. Amato, F.; Castiglione, A.; Cozzolino, G.; Narducci, F. A semantic-based methodology for digital forensics analysis. *J. Parallel Distrib. Comput.* **2020**, *138*, 172–177. [CrossRef]