electronics

MDPI

Article

A Zero-Knowledge-Proof-Based Anonymous and Revocable Scheme for Cross-Domain Authentication

Xinjian Zhao [1], Fei Xia [1], Hanning Xia [2,*], Yunlong Mao [2] and Shi Chen [1]

[1] Information & Telecommunication Branch State Grid Jiangsu Electric Power Co., Ltd., Nanjing 210024, China; zhaoxj1@js.sgcc.com.cn (X.Z.); xiafei@js.sgcc.com.cn (F.X.); chens10@js.sgcc.com.cn (S.C.)
[2] Computer Science and Technology Department, Nanjing University, Nanjing 210023, China; maoyl@nju.edu.cn
* Correspondence: rec.rec@smail.nju.edu.cn

Abstract: Authentication is a crucial security service on the Internet. In real-world applications, multiple independent trust domains often exist, with each recognizing only certain identities within their own systems. During cross-domain access, users cannot directly use their original certificates, which presents a cross-domain authentication problem. Traditional centralized schemes typically employ a trusted third party (TTP) to facilitate the transfer of identity trust across domains. These schemes inevitably inherit the vulnerabilities associated with single points of failure. In contrast, blockchain-based decentralized schemes effectively eliminate the potential threats posed by TTPs. However, the openness and transparency of the blockchain also bring new security issues, such as privacy leakage. In this paper, we propose a zk-SNARK-based anonymous scheme on the blockchain for cross-domain authentication. Specifically, our scheme adopts an authorization-then-proof structure, which strikes a delicate balance between anonymity and revocability. We provide theoretical proofs for the security of our scheme and explain how it achieves proactive revocability. Experimental evaluation results demonstrated that our scheme is both secure and efficient, and the revocation could be accomplished by introducing only 64 bytes of on-chain storage with one hash comparison.

Keywords: cross-domain authentication; blockchain; zero-knowledge proof; zk-SNARK; security; privacy preserving

Citation: Zhao, X.; Xia, F.; Xia, H.; Mao, Y.; Chen, S. A Zero-Knowledge-Proof-Based Anonymous and Revocable Scheme for Cross-Domain Authentication. *Electronics* **2024**, *13*, 2730. https://doi.org/10.3390/electronics13142730

Academic Editor: Zbigniew Kotulski

Received: 6 June 2024
Revised: 7 July 2024
Accepted: 9 July 2024
Published: 11 July 2024

1. Introduction

Identity authentication and management are essential components of the digital world. Before providing services and resources, administrators must verify the legitimacy of users to prevent unauthorized accesses. Cryptographic primitives, such as asymmetric encryption and public key infrastructure (PKI), effectively address this problem. However, the presence of different identity management organizations leads to the formation of many independent trust domains. Users cannot directly use their original certificates when accessing resources across domains. Instead, they must complete the transfer of identity trust between the original trust domain and the target trust domain, which is known as cross-domain authentication.

Traditional cross-domain authentication schemes typically rely on a trusted third party (TTP), such as a certificate authority (CA) or a key generation center (KGC). In these schemes, the transfer of identity trust across domains is facilitated by the TTP. However, these schemes inevitably inherit the vulnerabilities of single-point failure. Additionally, the trustworthiness of TTPs is not always assured in real-world applications. It is desirable to reduce the reliance on TTPs in authentication schemes.

A blockchain is a transparent and decentralized public ledger managed by a peer-to-peer (P2P) network. Each block contains data submitted by users, which is propagated and stored across the entire network based on predefined consensus protocols [1]. A blockchain ensures reliable data transmission in untrusted networks. Moreover, user data can consist

of program codes, namely, smart contracts. A contract code is executed and verified by the entire network, further extending a blockchain's applicability [2]. A blockchain brings new possibilities for decentralized authentication schemes, but also comes with new challenges. The openness and transparency of a blockchain directly lead to privacy leakage issues, as data on the blockchain are replicated across the entire network and accessible to all users. In an authentication system, data are often closely associated with user privacy. Meanwhile, smart contracts cannot securely use private keys in the computation, considering that every node can access the value of the keys on the blockchain. Many cryptographic primitives cannot be directly migrated to blockchain-based schemes.

To overcome the above challenges, we propose a zk-SNARK-based solution for privacy-preserving cross-domain authentication. We also consider the conflict between anonymity and revocability in authentication, which lies in the challenge of maintaining privacy while gaining the capability to disclose a user's identity under specific circumstances. A zero-knowledge proof enables the verification of data compliance without revealing the contents, and zk-SNARK supports a more general-purpose proving functionality [3,4]. Based on this, we designed an authorization-then-proof structure to balance the anonymity and revocability. Compared with existing solutions, our scheme provides a more proactive capability for revocation. The main contributions of our work are as follows:

- We combined zk-SNARK with multiple cryptographic primitives to achieve privacy-preserving cross-domain authentication on a blockchain. The properties of zk-SNARK allowed us to perform effective authentication without compromising user privacy.
- We adopted an authorization-then-proof structure for anonymous and revocable authentication, which strikes a delicate balance between anonymity and revocability. Ensuring privacy preservation, our scheme provides proactive revocability with a minimal cost. Upon revoking authorization, the user not only loses the ability to access anonymously but also has their anonymous access trails revealed.
- We implemented a proof-of-concept prototype for the scheme to evaluate its performance. Compared with existing works, the experimental results show that our scheme is highly applicable.

The rest of our article is organized as follows. Section 2 presents related work on cross-domain authentication. Section 3 gives the preliminaries. Section 4 defines the model and assumptions. Section 5 provides a detailed description of our anonymous cross-domain authentication scheme. Section 6 presents the implementation and evaluation of our scheme. Finally, we conclude the article in Section 7.

2. Related Work

This section focuses on the impact of a blockchain on cross-domain authentication. Accordingly, existing works can be divided into two categories: centralized and decentralized.

2.1. Centralized Schemes

Centralized cross-domain authentication often employs a TTP, such as a CA or a KGC. In real-world applications, it is crucial to ensure the reliability of TTPs. Pole et al. proposed a bridge certification authority (BCA), which employs a third-party CA to bridge the trust chains between multiple CAs [5]. Bai et al. proposed a cross-certification model in which CAs from different trust domains establish a two-way trust relationship by issuing certificates to each other [6]. Liu and Yang designed a PKI hybrid trust model that establishes a trusted proxy for authentication [7]. Chen et al. introduced a tripartite public key infrastructure (TriPKI), which ensures reliable certificate management through multi-party checks and mutual authentication [8]. Chen et al. proposed a secure cross-domain authentication scheme based on a threshold signature [9]. In refs. [10,11], data perturbation was introduced to achieve efficient computation while ensuring privacy preservation. Ref. [12] proposed a privacy-chain-based homomorphic encryption scheme to preserve the private and sensitive information about the user. These schemes employed cutting-

edge privacy-preserving strategies. However, these strategies fail to effectively achieve both anonymity and revocability in cross-domain authentication scenarios. Zhang et al. proposed a cross-domain authentication scheme based on identity-based cryptography (IBC) [13]. IBC eliminates the dependence on certificates and features a more lightweight communication overhead, but it still requires a KGC for key generation.

2.2. Decentralized Schemes

In decentralized solutions, certificates and other data can be stored on a blockchain to eliminate the reliance on TTPs. On the other hand, it is important to consider the impact of the blockchain's transparency and openness. Some papers propose improvements based on the traditional PKI model. Wang et al. designed a blockchain-based cross-domain authentication model called BlockCAM, which stores hashed certificates on the blockchain and achieves efficient identity authentication through a hash comparison [14]. In ref. [15], a blockchain-based efficient key exchange protocol is proposed for establishing cross-domain trust. Wang et al. proposed blockchain-based certificate transparency (CT) and revocation transparency (RT) to balance the absolute authority of CAs in the PKI model [16]. Chen et al. proposed the CertChain scheme, which leverages the blockchain to address the deficiencies of the traditional PKI model [17]. For CertLedger, Kubilay et al. designed a new PKI structure based on the blockchain to achieve reliable multi-CA distributed collaboration and transparent certificate management [18].

Many studies combined a zero-knowledge proof with the blockchain to achieve generalized and privacy-preserving authentication [19–24]. In BPCDA [21], Jiang et al. utilized NIZK and Pedersen commitment for mutual and anonymous authentication. Sani et al. proposed an efficient authentication scheme, Xyreum, to address the high-complexity and latency challenges introduced by a blockchain [19]. Xyreum uses a time-based zero-knowledge proof of knowledge (T-ZKPK) for identity authentication. Yang et al. implemented a digital identity management system BZDIMS, where entities can perform anonymous authentication based on zk-SNARK [20]. With XAuth, Chen et al. achieved an efficient anonymous authentication by proving the possession of legitimate certificates [22]. Rosenberg et al. designed a flexible anonymous authentication protocol, zk-creds, which can directly convert existing identity documents into anonymous credentials [23]. However, zk-creds handles revocation by updating the membership between the domain and the user. Administrators lack a more proactive revocability and cannot distinguish between expired credentials (regular users) and revoked credentials (malicious users).

It is feasible to utilize blockchain to eliminate the reliance on TTPs, and a zero-knowledge proof can provide more generalized and privacy-preserving functionality. However, most of the above schemes fail to simultaneously consider privacy, anonymity, and revocability. Furthermore, it is essential to consider the on-chain overhead incurred by the scheme. To solve these challenges, we propose an effective scheme for cross-domain authentication, which adopts an authorization-then-proof structure to balance anonymity and revocability.

3. Preliminaries

3.1. Blockchain and Smart Contract

A blockchain is a decentralized public ledger maintained by a P2P network. Transactions containing user data are aggregated into blocks, which are subsequently linked together in a chronological chain. Smart contracts are user-written codes stored on the blockchain, which automatically execute when specific conditions are met. Each blockchain node runs the contract code locally and maintains the global state through consensus protocols. In this paper, we consider the blockchain as an ideal public platform where users can read/write data and perform specific operations. Specifically, the ideal blockchain model has the following properties:

1. Transparency: The blockchain is modeled as an ideal public ledger. User data are signed, broadcast to the entire blockchain network, and subsequently packaged into

blocks. Anyone can access the blockchain's internal state at any time. In other words, the internal state of the blockchain is open and transparent to the entire network. Accordingly, when a secret key is used on the blockchain, it is leaked to everyone.
2. Immutability: User data uploaded to the blockchain is permanently stored. Tampering with data on the blockchain is impossible unless an attacker gains control of over 51% of the network nodes.
3. Reliability: Essentially, a blockchain is a decentralized state machine driven by user behavior (including transfer transactions and smart contract execution). We assumed that blockchain network nodes are able to update the current blockchain state based on user behavior in a timely and reliable way and maintain an effective consensus in the network. We assumed that nodes can faithfully update the blockchain state, and a consensus state can be reached within a limited time.

It is worth mentioning that in practical applications, blockchains can be categorized into private, federated, and public chains. Our solution primarily emphasizes that administrators can reliably transmit certain data through the blockchain. Typically, this means that a private chain or federated chain is used, and people can choose according to the specific application scenario.

3.2. Cryptographic Primitives

3.2.1. Digital Signature

Digital signature technology, which is based on asymmetric cryptography, verifies the authenticity of digital messages. The signer generates a signature for the message using the private key, while the verifier checks the validity of the signature with the public key. Any valid signature can be effectively authenticated. Simultaneously, the unforgeability of digital signatures ensures that only with the private key can one generate a valid digital signature. A digital signature primarily comprises the following methods:

- $DS.Gen(1^{\lambda}) \rightarrow (pk, sk)$: generate a key pair, where pk is the public key and sk is the private key.
- $DS.Sign(sk, m) \rightarrow s$: sign message m with sk and generate a signature s.
- $DS.Verify(pk, m, s) \rightarrow 0/1$: verify the validity of s with pk.

3.2.2. Merkle Tree

A Merkle tree is a data structure used for verifying the integrity and consistency of datasets [25]. In this paper, we use this structure to verify membership between elements and sets, which involves the following methods:

- $MTree.Root(\vec{x}) \rightarrow rt$: Generate a Merkle tree with the sequence $\vec{x}$. The algorithm outputs the root rt of the tree.
- $MTree.Path(\vec{x}, x_0) \rightarrow \theta$: for an element x_0 in the sequence $\vec{x}$, compute a Merkle tree path θ from the leaf node to the root.
- $MTree.isValidPath(rt, \theta) \rightarrow 0/1$: Verify the Merkle tree path θ. A valid path implies an effective membership.

3.2.3. Zero-Knowledge Proof

A zero-knowledge proof is a powerful cryptographic primitive that enables one party (the prover) to prove to another party (the verifier) that the given statement is true without revealing any information other than the validity of the statement. It has the following security properties:

1. Completeness: if the statement is true, an honest verifier will be convinced.
2. Soundness: if the statement is false, no prover can cheat the verifier about the truth.
3. Zero knowledge: the verifier learns no information other than the validity of the statement.

Zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARK) further enable a more general-purpose proving functionality. The prover can generate a

succinct proof in a non-interactive way, which means more flexible and efficient proving and verifying processes. Formally, zk-SNARK proves that for a public input $\vec{x}$, there exists a private witness $\vec{w}$ such that $C(\vec{x}, \vec{w}) = 1$, where C describes a constraint system. In this paper, we use the Groth16 algorithm, which is a pairing-based zk-SNARK system that requires a trusted setup [4]. The Groth16 algorithm involves the following methods:

- $G16.Setup(circ) \rightarrow crs$: generate a common reference string crs based on the given circuit $circ$.
- $G16.Prove(crs, \vec{x}, \vec{w}) \rightarrow \pi$: Prove the existence of a private witness $\vec{w}$ such that $\vec{w}$ and the public input $\vec{x}$ satisfy the constraint system described by crs. The algorithm outputs a proof π.
- $G16.Verify(crs, \vec{x}, \pi) \rightarrow 0/1$: verify the proof π with the public input $\vec{x}$.

zk-friendly algorithms: In this study, we needed to construct proof circuits for certain cryptographic primitives, such as hash functions and digital signatures. Traditional algorithms (such as SHA256 and ECDSA) may lead to large and complex circuit structures, hence resulting in high proof costs. We introduced zk-friendly cryptographic algorithms into our scheme to optimize the circuit while gaining security. We used the Poseidon algorithm as the standard hash algorithm and the EdDSA algorithm as the digital signature algorithm.

4. Definitions

4.1. Model

In the cross-domain authentication model, there are three main parties:

- Domain administrator: Cross-domain authentication involves several different trust domains. We assumed there exists a unique administrator in each domain who is mainly responsible for the following tasks:
  - A domain administrator that manages the identity information of users in the local domain.
  - A domain administrator that interacts with the blockchain to maintain the public information required for cross-domain authentication.
  - In cross-domain authentication, a domain administrator that verifies the cross-domain visitors.
- User: User refers to an entity, such as a human or a device, that requires resources and services from different domains. We assumed that each user belongs to an initial trust domain and then establishes cross-domain access with new trust domains.
- Blockchain: We modeled the blockchain as an ideal decentralized public ledger for reliable data transmission and smart contract execution. Transactions initiated by users are guaranteed to be synchronized across the entire blockchain network within a limited time. Additionally, the storage and computation on the chain will lead to corresponding fees.

Threat model: Any cross-domain access requires effective authentication. Trust domain administrators are semi-honest: they will faithfully execute management and authentication works, while potentially being curious about information from users in other trust domains. Users are considered untrusted and may engage in various forms of malicious attacks and conspiracies. Specifically, we assumed that administrators have the ability to authenticate the identities of users within their own trust domain, including but not limited to physical authentication methods. The blockchain is trustworthy, and administrators from different trust domains are able to reliably transmit certain data through the chain. Similarly, anyone can freely access data shared on the blockchain (including historical versions).

4.2. Design Features

In order to achieve anonymous and privacy-preserving cross-domain authentication, our scheme has the following important features:

- Security: Similar to a digital signature, the security in our work primarily emphasizes unforgeability. A malicious user is unable to forge a valid certificate without domain administrators. Furthermore, users cannot arbitrarily modify the attributes promised by the certificate, ensuring the validity of the certificate.
- Unlinkability: The unlinkability of the scheme captures the concept of anonymity. After the cross-domain authentication, the user uses a pseudonym for cross-domain activities. The unlinkability implies no one can link a user's activities to their real identity. Our work emphasizes further unlinkability. Even a domain administrator cannot link any two pseudonyms of the same user.
- Revocability: On the one hand, we expect that certificates will expire periodically, and this can be achieved by binding certificates with timestamp attributes. On the other hand, we emphasize that administrators should play a more proactive role in revocability. Whenever administrators detect malicious users in the domain, they should immediately revoke the validity of user certificates (if already issued) rather than passively waiting for the certificates to expire automatically. To ensure a balance between anonymity and revocability, we employed zero-knowledge proof in our scheme. We further explore revocability in Section 5.2.
- Efficiency: In real-world applications, identity authentication is typically a high-frequency service. We needed the scheme to have considerable resource utilization efficiency for better practicality. Considering that blockchain and zero-knowledge proof are crucial parts of our work, we needed to pay close attention to on-chain/off-chain computation and storage costs, communication costs, proof circuit complexity, etc.

5. Methods

5.1. Overview

In this section, we construct an anonymous and revocable scheme for cross-domain authentication. We start by providing an overview of the proposed work.

As mentioned before, each user belongs to an initial trust domain and can be authenticated by the administrator through a trustworthy way. There are four roles in a cross-domain authentication process: user U, original domain administrator D_a, target domain administrator D_b, and the blockchain. We will continue to use these symbols to simplify the presentation. In our scheme, we adopted an authorization-then-proof structure. The user with cross-domain access requirements needs to apply for authorization from D_a to generate certificates. U initiates an access request and sends a certificate to D_b. D_b obtains basic information about the original trust domain through the blockchain and then verifies the validity of the certificate. After the authentication, U can use the pseudonym corresponding to the certificate for cross-domain access activities. The workflow of cross-domain authentication is shown in Figure 1.

To achieve anonymous and private-preserving authentication, we introduced zero-knowledge proof to our scheme. Intuitively, U needs to request certain private witnesses from D_a as authorization and then generate valid cross-domain certificates. We aimed to bind authorization with user attributes and ensured that they expire periodically. Furthermore, we expected that the administrators should be able to revoke authorizations from malicious users at any time.

5.2. Zero-Knowledge Anonymous and Revocable Authentication

We first constructed a new cryptographic primitive, balancing anonymity and revocability during authentication.

During authentication, each user can be seen as a set of attributes, and the authentication process involves demonstrating that the user's attributes conform to certain predefined constraints. Apparently, users should only provide essential information during authentication to ensure privacy preservation. For instance, when accessing restricted resources across domains, the user may need to demonstrate their privilege in the original domain,

while other information is unnecessary. We partitioned user attributes into two categories: public attributes $attr_{pub}$ and private attributes $attr_{prv}$. The privacy-preserving authentication process involves demonstrating that $attr$ satisfies a constraint set C with only $attr_{pub}$, that is, $\bigwedge_{C_i \in C} C_i(attr_{pub}, attr_{prv})$.

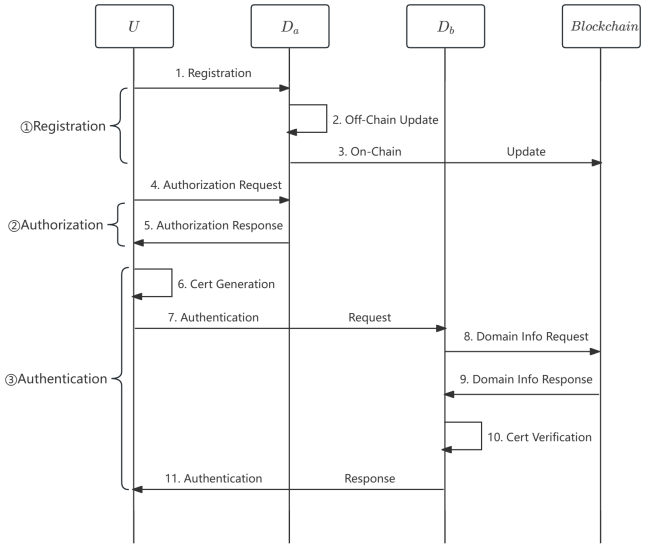


Figure 1. The workflow of cross-domain authentication.

In cross-domain authentication, membership between users and the original domain plays a pivotal role. We utilized the Merkle tree structure to verify membership. Specifically, administrators compute a Merkle tree with the user list $\vec{U}$ and publish the Merkle root rt to the blockchain. A Merkle tree path θ implies a valid membership between a user and the domain, i.e., $C_i := Mtree.isValidPath(rt, \theta)$. We note that modifications to $\vec{U}$ will result in changes to the root rt, thus rendering old paths unverifiable. Therefore, administrators will maintain a list of roots $\vec{rt}$ on the blockchain, storing roots generated over a period of time. Given that user list modifications are infrequent in practical scenarios, the size of $\vec{rt}$ is acceptable.

Our authentication scheme adopts an authorization-then-proof structure. A potential security concern pertains to the reliability of the certificates generated by users. In other words, we needed to ensure that only authorized users could produce valid certificates and that the user attributes revealed in certificates are genuine and reliable. We utilized digital signatures to bind user attributes with authorization. In the authorization process, D_a generates a signature s on user attributes $attr$ and then sends it to U; in the authentication process, U needs to demonstrate that their attributes match the signature s. This ensures the security of our scheme, as malicious users cannot generate a valid signature for the forged attributes without the private key sk. We observed that during the binding of user attributes, D_a can include extra attributes $attr_{auth}$, thereby enhancing the functionality of the authorization. An intuitive example is to bind a timestamp into the authorization. By checking the timestamp, administrators can ensure the timely expiration of authorization.

Now, we present the construction our authentication primitives. We primarily employed zk-SNARK to meet the requirements of privacy preservation. The scheme includes the following methods:

- $Setup(1^\lambda, circ)$: Initialization for authentication. The algorithm generates a $crs := G16.Setup(circ)$ for a zk-SNARK circuit and a key pair $(pk, sk) := DS.Gen(1^\lambda)$ for a digital signature, where λ is a security parameter. The algorithm outputs (pk, sk, crs).
- $Authorize(U, \vec{U}, sk, attr_{auth})$: The algorithm computes the Merkle root $rt := MTree.Root(\vec{U})$ and a Merkle path $\theta := MTree.Path(\vec{U}, U)$ for U. It then com-

putes a signature $s := DS.Sign(sk, \theta||attr||attr_{auth})$. The algorithm outputs an authorization $\sigma := (rt, \theta, attr, attr_{auth}, s)$.

- $CertGen(crs, pk, nonce, \sigma)$: The algorithm provides a certificate and a pseudonym for authentication and subsequent access. It computes the pseudonym by $PID := Hash(nonce||s)$, where *nonce* is a one-time serial number for the anonymity of the pseudonym. For zk-SNARK, we set the private witness $\vec{w} = (\theta, attr_{prv}, s)$ and the public input $\vec{x} = (pk, rt, attr_{pub}, attr_{auth}, nonce, PID)$. The algorithm generates a proof $\pi = G16.Prove(crs, \vec{x}, \vec{w})$ for the following constraints (1):

$$\begin{aligned} &ConsistencyCheck(\theta, attr) \\ &\wedge MTree.isValidPath(rt, \theta) \\ &\wedge DS.Verify(pk, \theta||attr||attr_{auth}, s) \\ &\wedge PID = Hash(nonce||s) \end{aligned} \tag{1}$$

  ConsistencyCheck verifies the consistency between *attr* and the leaf node in θ. The algorithm outputs a pseudonym PID and the certificate $cert := (nonce, rt, attr_{pub}, attr_{auth}, \pi)$.
- $CertVerify(crs, pk, cert, PID)$: the algorithm parses the public input $\vec{x} = (pk, rt, attr_{pub}, attr_{auth}, nonce, PID)$ and runs $G16.Verify(crs, \vec{x}, \pi)$ to verify the proof π.
- $RevocationCheck(nonce, s, PID)$: the algorithm compares $PID \stackrel{?}{=} Hash(nonce||s)$ to check whether PID was generated by a revoked authorization.

Revocability: We considered the revocability of pseudonyms. In our scheme, a pseudonym is generated by hashing $nonce||s$, where *nonce* is public. If s is disclosed, all pseudonyms generated by s can be revealed through a hash comparison. Therefore, domain administrators can maintain a blacklist $\vec{s}$ on the blockchain and keep all issued authorizations locally, stored in the list $\vec{\sigma}$. When the authorization σ of a malicious user needs to be revoked, the administrator can simply find the corresponding signature s of σ and add it to the blacklist $\vec{s}$. During the authentication, the administrator can verify whether an authorization has been revoked by checking whether the pseudonym was generated with an element in the blacklist. Revoked malicious users will no longer be able to generate valid certificates and pseudonyms. Moreover, all anonymous access trails will be disclosed, considering that each *nonce* is available. Our scheme not only helps administrators to mitigate threats from malicious users but also reveals the complete attack chain, thus demonstrating excellent practicality.

5.3. Cross-Domain Authentication: Protocol

Now, we introduce the detailed content of the authentication scheme. The description of our work is as follows.

5.3.1. Initialization and Registration

According to different requirements, administrators design corresponding constraint circuits *circ* and retain a zero-knowledge proof document *doc*. For initialization, administrators generate parameters for digital signature and zk-SNARK through $(pk, sk, crs) := Setup(1^{\lambda}, circ)$. Administrators from different trust domains join the blockchain network and publish (pk, crs), the zero-knowledge proof document *doc*, trust domain information, etc., on the blockchain. List $\vec{\sigma}$, $\vec{rt}$, and $\vec{s}$ are emptied during this stage.

Each user belongs to an initial trust domain. Users are able to register in the initial domain in a reliable way. After authentication, the administrator stores the registered user information U in a local user list $\vec{U}$.

Protocol 1 Initialization

Initial Setup:

1. D: generate $(pk, sk, crs) := Setup(1^{\lambda}, circ)$ and a zkp document doc.
2. D: update (pk, crs, doc) to the blockchain.
3. D: initialize user list $\vec{U}$ and authorization list $\vec{\sigma}$ off chain.
4. D: initialize Merkle root list $\vec{rt}$ and black list $\vec{s}$ on chain.

Registration:

1. U: initiate registration request to D_a.
2. D_a: verify the identity of U.
3. D_a: update $\vec{U}$ with U.

5.3.2. Authorization

In this stage, U needs to request authorization from the original domain administrator D_a for cross-domain authentication.

U downloads (pk, crs, doc) from the blockchain and initiates a request for authorization. D_a first checks whether U is compliant with the registration. After that, D_a computes a Merkle tree with the current $\vec{U}$ and updates $\vec{rt}$ on a chain if rt is not in the list. D_a specifies the content of $attr_{auth}$ according to different circumstances and generates an authorization $\sigma := Authorize(U, sk, attr_{auth})$ for U. D_a adds σ to $\vec{\sigma}$ locally and sends σ to U. At this point, U is able to generate a valid cross-domain authentication with σ.

In our scheme, administrators update the status of $\vec{rt}$ only when new authorizations are generated rather than when $\vec{U}$ is updated. Considering that only new authorizations require new Merkle tree roots, this strategy is feasible and effectively controls the on-chain overhead of the scheme.

Protocol 2 Authorization

Authorization Request:

1. U: get (pk, crs, doc) of D_a on chain.
2. U: initiate authorization request to D_a.

Authorization Response:

1. D_a: check the validity of U.
2. D_a: if not, reject U and abort.
3. D_a: generate $rt := MTree.Root(\vec{U})$.
4. D_a: if rt not in $\vec{rt}$, update $\vec{rt}$ on chain.
5. D_a: generate $attr_{auth}$ for authorization.
6. D_a: generate authorization $\sigma := Authorize(U, \vec{U}, sk, attr_{auth})$.
7. D_a: update $\vec{\sigma}$ with σ.
8. D_a: send σ to U.

5.3.3. Authentication and Revocation

In the authentication stage, U formally undergoes cross-domain authentication with D_b.

U initiates a cross-domain authentication request to the target trust domain administrator D_b. D_b generates a one-time serial number $nonce_D$ and sends it to U, while U locally generates a $nonce_U$ and obtains $nonce := nonce_D || nonce_U$. This simply ensures that the serial number $nonce$ is jointly negotiated by the user and the administrator, preventing either party from compromising anonymity. U generates $(PID, cert) := CertGen(crs, pk, nonce, \sigma)$ with the authorization σ and sends them to D_b.

For authentication, D_b needs to obtain the information $(pk, crs, doc, \vec{s})$ of the original trust domain from the blockchain. First, D_b runs $RevocationCheck(nonce, s, PID)$ for each s in $\vec{s}$ to find out whether PID was generated by a revoked authorization. After this, D_b performs a content check on $cert$, verifying whether $(attr_{pub}, attr_{auth})$ meets the conditions for cross-domain access. D_b determines the validity of rt by checking the existence of rt in

$\vec{rt}$, and then runs $CertVerify(crs, pk, cert, PID)$ to verify $cert$. If $cert$ passes all the checks above, D_b will allow U to engage in cross-domain activities with the pseudonym PID.

Protocol 3 Authentication

Authentication Request:

1. U: initiate authentication request to D_b.
2. D_b: generate serial number $nonce_D$.
3. D_b: send $nonce_D$ to U.
4. U: generate serial number $nonce_U$.
5. U: splice $nonce := nonce_U || nonce_D$.
6. U: generate $(PID, cert) := CertGen(crs, pk, nonce, \sigma)$.
7. U: send $(PID, cert)$ to D_b.

Authentication Response:

1. D_b: get $(pk, crs, doc, \vec{rt})$ of D_a on chain.
2. D_b: parse $(nonce, rt, attr_{pub}, attr_{auth}, \pi) := cert$.
3. D_b: perform pre-authentication revocation check. [1]
4. D_b: perform content check for $(attr_{pub}, attr_{auth})$.
5. D_b: perform consistency check for rt with $\vec{rt}$.
6. D_b: perform verification by $CertVerify(crs, pk, cert, PID)$.
7. D_b: if either check fails, reject $cert$ and abort.
8. D_b: authorize U for cross-domain activities with PID.

[1] More details in revocation protocol.

When malicious behavior is detected within the original trust domain, administrator D_a first locates the malicious user U. D_a queries the list $\vec{\sigma}$ for the valid authorization σ that was issued for U and, for revocation, adds the signature s in σ to the blacklist $\vec{s}$ on the blockchain. Since the *nonce* for the pseudonym PID is known, one can simply run *RevocationCheck* to determine whether PID was generated by a revoked authorization. With this strategy, D_b can detect and reject cross-domain requests with revoked authorizations in the authentication. Additionally, D_b can extract all the access activities under this authorization from access logs, thus tracing the attack chain of a malicious user.

Protocol 4 Revocation

Authorization Revocation:

1. D_a: locate malicious user U in trust domain.
2. D_a: obtain authorization σ of U from $\vec{\sigma}$.
3. D_a: parse $(rt, \theta, attr, attr_{auth}, s) := \sigma$.
4. D_a: update $\vec{s}$ with s on chain.

Pre-authentication:

1. D_b: obtain $\vec{s}$ of D_a on chain.
2. D_b: parse $(nonce, rt, attr_{pub}, attr_{auth}, \pi) := cert$.
3. D_b: perform $RevocationCheck(nonce, s, PID)$ for each s in $\vec{s}$.
4. D_b: if any s fails, reject $cert$.
5. D_b: continue to other checks.

Post-authentication:

1. D_b: locate malicious user with PID in trust domain.
2. D_b: obtain $\vec{s}$ of D_a on chain.
3. D_b: obtain $cert$ of the malicious user.
4. D_b: parse $(nonce, rt, attr_{pub}, attr_{auth}, \pi) := cert$.
5. D_b: perform $RevocationCheck(nonce, s, PID)$ for each s in $\vec{s}$.
6. D_b: obtain s of malicious authorization σ.
7. D_b: perform $RevcationCheck(nonce, s, PID)$ for each $(nonce, PID)$ in access logs.
8. D_b: reveal all malicious accesses by U.

6. Security Analysis

We formally define soundness, anonymity, and completeness properties, and prove that our cross-domain authentication scheme perfectly satisfies these properties. Then, we analyze the security of our scheme against different attacks.

6.1. Security Properties

Definition 1 (Soundness)**.** *The scheme is stable and unforgeable for all probabilistic polynomial time adversaries $\mathcal{A}$, and the following relation holds:*

$$Pr\left[\begin{array}{c}(pk, sk, crs) \leftarrow Setup(1^{\lambda}, circ)\\ \sigma' \leftarrow \mathcal{A}^{Forge}(attr_{pub}, attr_{prv}, attr_{auth})\\ (PID', cert') \leftarrow CertGen^{Forge}(crs, pk, nonce, \sigma') :\\ \Sigma_{s\in\vec{s}} RevocationCheck(nonce, s, PID') = 0\\ CertVerify(crs, pk, cert', PID') = 1\end{array}\right] \approx 0$$

where Forge generates a forged authorization or generates a valid certificate and pseudonym without authorization.

Theorem 1. *The proposed scheme is sound if the digital signature and zk-SNARK are utilized properly and the hash function is collision resistant.*

Proof. Soundness captures the unforgeability of the scheme at three levels:

1. For an adversary who does not hold authorization, they will not be able to forge a valid authorization or generate a legitimate pseudonym and certificate without authorization.
2. For an adversary who holds an authorization, they will not be able to modify the attributes bound to the authorization or forge a new valid authorization based on the old one.
3. For an adversary whose authorization has been revoked, they will not be able to use the invalid authorization to generate legitimate pseudonyms and certificates anymore.

We bind user attributes to the authorization with a digital signature. Due to the security properties of digital signatures, one cannot forge or modify the content of the authorization. It is guaranteed by zk-SNARK that a certificate *cert* can be verified if and only if a valid authorization σ exists. Therefore, our scheme establishes a binding relationship between certificates and user attributes in the case of implicitly verifying authorization. Moreover, the collision resistance of the hash function ensures the availability of authorization revocation. In summary, our scheme satisfies the theorem of soundness. □

Definition 2 (Anonymity)**.** *The scheme is a zero-knowledge one and unlinkable for all adversaries $\mathcal{A}$, and we have the following:*

$$Pr\left[\begin{array}{c}(pk, sk, crs) \leftarrow Setup(1^{\lambda}, circ)\\ \sigma' \leftarrow \mathcal{A}(attr_{pub}, attr_{prv}, attr_{auth})\\ (PID', cert') \leftarrow CertGen(crs, pk, nonce, \sigma') :\\ \Sigma_{s\in\vec{s}} RevocationCheck(nonce, s, PID') = 0\\ CertVerify(crs, pk, cert', PID') = 1\end{array}\right] \approx$$

$$Pr\left[\begin{array}{c}(pk, sk, crs, \tau) \leftarrow Setup^{Simulate}(1^{\lambda}, circ)\\ \sigma' \leftarrow \mathcal{A}(attr_{pub}, attr_{prv}, attr_{auth})\\ (PID', cert') \leftarrow CertGen^{Simulate}(crs, pk, nonce, \sigma', \tau) :\\ \Sigma_{s\in\vec{s}} RevocationCheck(nonce, s, PID') = 0\\ CertVerify(crs, pk, cert', PID') = 1\end{array}\right]$$

where Simulate generates a trapdoor τ and outputs a simulated certificate.

Theorem 2. *The authentication scheme is zero knowledgeable and unlinkable for all adversaries if the hash function and zk-SNARK are utilized properly.*

Proof. Our scheme captures anonymity as being zero knowledgeable and unlinkable. Zero knowledge means that users do not reveal any private information except for the necessary public attributes, and unlinkability indicates that no one can link pseudonyms to the user identities or link multiple anonymous accesses by the same user. Zero knowledge of the scheme is guaranteed by the security properties of zk-SNARK. Adversaries are unable to extract private witnesses through user certificates, and thus, cannot compromise user privacy. In addition, user pseudonyms are determined by both authorization and *nonce*. Since *nonce* is negotiated between the user and the administrator, multiple anonymous accesses generated by the same authorization cannot be linked without breaking the hash function. Therefore, the proposed scheme fulfills the theorem of anonymity. □

Definition 3 (Completeness)**.** *The scheme ensures users with valid authorization to be successfully cross-domain authenticated, and the following relation holds:*

$$Pr\left[\begin{array}{c}(pk, sk, crs) \leftarrow Setup(1^{\lambda}, circ)\\ \sigma \leftarrow \mathcal{A}(attr_{pub}, attr_{prv}, attr_{auth})\\ (PID, cert) \leftarrow CertGen(crs, pk, nonce, \sigma):\\ \Sigma_{s\in\vec{s}} RevocationCheck(nonce, s, PID) = 0\\ CertVerify(crs, pk, cert, PID') = 1\end{array}\right] \approx 1$$

where σ is a valid authorization exposed to $\mathcal{A}$.

Theorem 3. *The scheme is complete if the initialization, authorization, authentication, and revocation stages are executed correctly.*

Proof. Completeness is guaranteed by the correct execution of zk-SNARK. For any cross-domain authentication requirements, we can translate them into proof circuits and complete user authentications with zk-SNARK. One with valid authorization will always be successfully authenticated. Since the authorization is bound to user attributes, our scheme satisfies the completeness of cross-domain authentication. □

6.2. Security against Different Attacks

- Replay attack: An adversary $\mathcal{A}$ can eavesdrop on the certificates and pseudonyms transmitted and later attempt to replay the data fraudulently. In our scheme, the administrator binds a timestamp to the authorization. By checking the timestamp, one can determine the validity of the authorization, realizing the timed expiration function. Due to the unforgeability of digital signatures, $\mathcal{A}$ cannot tamper with the authorization timestamp, and thus, cannot implement certificate spoofing through replay. For user pseudonyms, its generation is determined by both the authorization and *nonce*, which is negotiated between the administrator and the user. This mechanism can effectively prevent the potential threat of replay attacks.
- MitM attack: An adversary $\mathcal{A}$ can insert itself between the user and the administrator from the original/target domain. In our scheme, domain information is publicly and reliably transmitted on the blockchain, making it difficult for $\mathcal{A}$ to impersonate an administrator. Additionally, the scheme utilizes zk-SNARK to implicitly verify user authorization, ensuring that no information beyond public attributes will be leaked. Consequently, $\mathcal{A}$ cannot deceive through a man-in-the-middle attack.
- Spoofing attack: An adversary $\mathcal{A}$ may attempt to spoof legitimate user identities or certificates to deceive the original/target domain administrator. Since administrators are able to verify user identities within their own domain, unauthorized users cannot deceive administrators to gain authorization. Due to the soundness of our scheme,

$\mathcal{A}$ is not able to forge valid certificates without proper authorization. Therefore, our scheme can resist spoofing attacks.

7. Implementations and Evaluations

7.1. Setup and Implementation

We implemented a prototype system for our scheme in Python (3.10). We used Ganache (7.9.2) to build a blockchain network that ran smart contracts programmed by Solidity (0.8.22), and we utilized Web3.py (6.15.1) to interact with the contracts. For zk-SNARK, we used Circom (2.1.8) and Snarkjs (0.7.3) in Node.js (18.19.0) to construct circuits and prove/verify arguments. Additionally, we used Circomlib (2.0.5) in Node.js to introduce zk-friendly cryptographic functions. Our system was deployed and evaluated on a personal computer running Windows 11 with Intel Core i7-13700F CPU@2.10 GHz and 16 GB RAM.

7.2. Experimental Results

The prototype system comprised three trust domains, with each managed by a single administrator and utilized by several users. For evaluation, we assessed the time cost, gas cost, communication cost, and proof cost, and we compared our scheme with Xyreum [19], BZDIMS [20], and BPCDA [21]. We mainly considered the impacts of the number of users and the number of attributes, and we assumed that public and private attributes increased simultaneously. We theoretically evaluated the cost of the scheme and verified it through experiments. For the same parameter setting, we used multiple experiments to reduce the experimental error.

7.2.1. Time Cost

We measured the time cost of our authentication scheme, including authorization, certificate generation, and certificate verification. We primarily considered the impact of the number of users with the results shown in Figure 2. In the authorization stage, the time cost mainly came from the Merkle tree update and digital signature, which was significantly affected by the data size. As the number of users increased, the number of leaf nodes in the Merkle tree also increased, thus leading to a larger proof circuit. Proof generation was the main cause of time cost during the certificate generation stage. Due to the succinctness of zk-SNARK, however, this only affected the certificate generation process but did not significantly affect the certificate verification process.

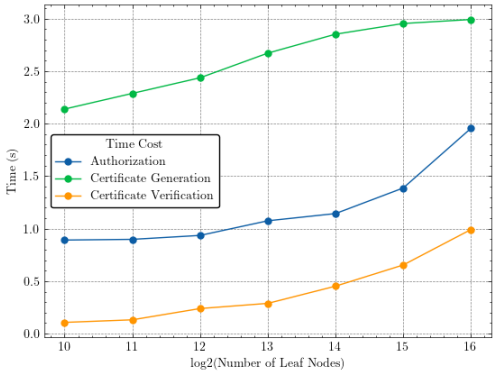


Figure 2. Time cost of the scheme.

Then, we compared our scheme with BZDIMS, which also uses zk-SNARK for privacy-preserving authentication. BZDIMS uses ZoKrates [26], which is a toolbox that consists of a domain-specific language (DSL), a compiler, and generators for witnessing and proof. ZoKrates provides a processing model for developing zero-knowledge proof projects on Ethereum. The results are shown in Table 1. We used zk-friendly algorithms, which brought about significant optimization of the time cost required for certificate generation.

In the certificate verification stage, our scheme needed to perform attribute checking and revocation checking, in addition to proof verification, thus resulting in additional time costs.

Table 1. Time cost (ms).

	Certificate Generation	Certificate Verification	Total
Our scheme	2069	53.63	2123
BZDIMS [20]	8096	11.80	8108

7.2.2. Gas Cost

In our scheme, trust domain administrators need to reliably transmit certain data on the blockchain via smart contracts, such as $(pk, crs, doc, \vec{rt}, \vec{s})$. Uploading and updating data on chain will lead to changes in the blockchain state, resulting in gas costs. We evaluated the gas cost of the scheme, as shown in Table 2. Xyreum, BPCDA, and BZDIMS bind user identity vouchers with the blockchain, while BZDIMS also computes identity verification on chain. In our scheme, the on-chain cost is mainly associated with the initialization stage, authorization stage, and revocation stage. In some cases, users/administrators only utilized on-chain data to assist with checks and computations, which did not result in a change in the state of the blockchain and, therefore, incurred no additional gas cost. Additionally, the data uploaded to the chain by the administrator primarily consisted of Merkle tree roots (hash values) and digital signatures, and our scheme ensured that on-chain states were updated only when necessary, thus maintaining a minimal on-chain cost.

Table 2. Gas cost (gas).

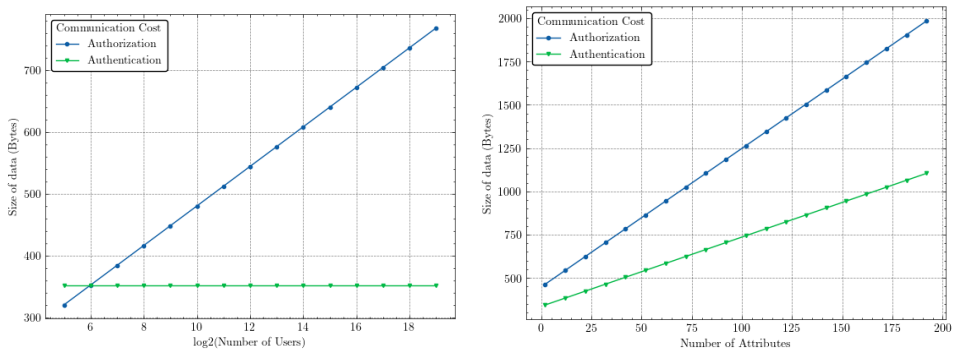
	Initialization	Authorization	Authentication	Revocation	Total
Our scheme	138,817	49,982	/	94,370	283,169
Xyreum [19]	177,183	/	177,127	25,803	380,113
BZDIMS [20]	392,037	/	375,327	32,739	800,103
BPCDA [21]	158,826	/	124,319	25,601	308,746

7.2.3. Communication Cost

We measured the communication cost of our scheme by calculating the data transmitted in the authentication process. We mainly considered the communication cost off chain, and the result is shown in Figure 3. When the number of user attributes increased, we assumed that public and private attributes increased simultaneously. In authorization, the number of users affected the size of the Merkle tree, which changed the size of the Merkle path in turn. In this stage, we had $Cost(Poseidon) \times log_2(NumOfUsers) + Cost(EdDSA) + Cost(attr_{pub} + attr_{prv} + attr_{auth})$ bytes of communication cost, where *Poseidon* incurred 32 bytes and $EdDSA$ incurred 64 bytes. In authentication, complex attribute constraints led to more auxiliary input data, increasing the communication cost. We had $Cost(Poseidon) + Cost(G16) + Cost(attr_{pub} + attr_{auth})$ bytes of communication cost. The hash algorithm and zk-SNARK effectively controlled the size of the communication data.

7.2.4. Proof Cost

Our scheme uses zk-SNARK to achieve anonymous and privacy-preserving cross-domain authentication. We evaluated the proof cost of the scheme with different data scales, as shown in Figure 4. Given the succinctness of zk-SNARK, the proof size and the verification time were typically small, and thus, we mainly focused on the proof circuit size. In our scheme, Merkle path verification is a critical component of the proof circuit, and we independently assessed its respective proof cost. By introducing zk-friendly cryptographic primitives into our scheme, we ensured a certain level of simplicity and efficiency in the zkp process.



(**a**) (**b**)

Figure 3. Communication cost of the scheme, with (**a**) different numbers of users and (**b**) different numbers of attributes.

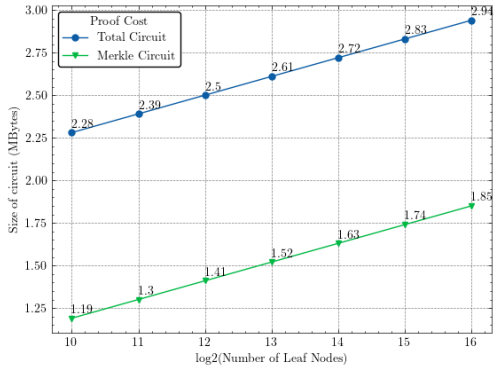


Figure 4. Proof cost of the scheme.

8. Conclusions

In this paper, we propose an anonymous and revocable scheme for cross-domain authentication. We built a decentralized scheme based on the blockchain to eliminate the potential threats of a TTP. To address the privacy and security issues inherent in the openness and transparency of the blockchain, we introduced zk-SNARK. We integrated zk-SNARK with various cryptographic primitives to meet the requirements of cross-domain authentication. Moreover, we struck a delicate balance between anonymity and revocability: our scheme adopted an authorize-then-proof structure to achieve anonymous cross-domain access while guaranteeing revocability and accountability for administrators. Ensuring privacy preservation, our scheme provides a proactive revocability with a minimal cost. When an authorization is revoked, not only is the user unable to access it anonymously again, but one's malicious access history will also be revealed, which helps to grasp the complete attack chain. Finally, we evaluated the performance of the scheme from several perspectives. Compared with existing works, it turns out that the scheme had good practicability.

Author Contributions: Conceptualization, X.Z.; methodology, X.Z. and F.X.; software, H.X.; validation, H.X. and Y.M.; formal analysis, H.X. and Y.M.; writing—original draft preparation, X.Z.; writing—review and editing, F.X.; supervision, Y.M. and S.C. All authors read and agreed to the published version of this manuscript.

Funding: This work was supported by the state Grid Jiangsu Electric Power Corporation Project (J2023178).

Data Availability Statement: The data presented in this study are available in this article.

Conflicts of Interest: Authors Xinjian Zhao, Fei Xia and Shi Chen were employed by the company Information & Telecommunication Branch State Grid Jiangsu Electric Power Co., Ltd. The remaining

authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. 2009. Available online: http://www.bitcoin.org/bitcoin.pdf (accessed on 8 July 2024).
2. Buterin, V. A Next Generation Smart Contract & Decentralized Application Platform. *White Paper* **2015**, *3*, 2-1.
3. Goldwasser, S.; Micali, S.; Rackoff, C. The knowledge complexity of interactive proof-systems. In Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing, New York, NY, USA, 5–8 May 1985; pp. 291–304. [CrossRef]
4. Groth, J. On the Size of Pairing-Based Non-interactive Arguments. In Proceedings of the Advances in Cryptology—Eurocrypt, Vienna, Austria, 8–12 May 2016; Fischlin, M., Coron, J.S., Eds.; Springer: Berlin/Heidelberg, Germany, 2016; pp. 305–326.
5. Polk, W.; Hastings, N. Bridge certification authorities: Connecting B2B public key infrastructures. In Proceedings of the 16th IST Mobile and Wireless Communications Summit, West Point, NY, USA, 18–20 June 2001.
6. Bai, Q.H.; Zheng, Y.; Zhao, L.; Chun, H.; Cheng, C. Research on Mechanism of PKI Trust Model. *Appl. Mech. Mater.* **2014**, *536–537*, 694–697. [CrossRef]
7. Liu, Y.; Yang, Z. The Research and Design of the Proxy for Certificate Validation Based on Distributed Cross-Certification. In Proceedings of the 2017 5th Intl Conf on Applied Computing and Information Technology/4th Intl Conf on Computational Science/Intelligence and Applied Informatics/2nd Intl Conf on Big Data, Cloud Computing, Data Science (ACIT-CSII-BCD), Hamamatsu, Japan, 9–13 July 2017; pp. 135–140. [CrossRef]
8. Chen, J.; Yao, S.; Yuan, Q.; Du, R.; Xue, G. Checks and balances: A tripartite public key infrastructure for secure web-based connections. In Proceedings of the IEEE INFOCOM 2017—IEEE Conference on Computer Communications, Atlanta, GA, USA, 1–4 May 2017; pp. 1–9. [CrossRef]
9. Chen, L.; Guo, C.; Gong, B.; Waqas, M.; Deng, L.; Qin, H. A secure cross-domain authentication scheme based on threshold signature for MEC. *J. Cloud Comput.* **2024**, *13*, 70. [CrossRef]
10. Sathish Kumar, G.; Premalatha, K.; Uma Maheshwari, G.; Rajesh Kanna, P.; Vijaya, G.; Nivaashini, M. Differential privacy scheme using Laplace mechanism and statistical method computation in deep neural network for privacy preservation. *Eng. Appl. Artif. Intell.* **2024**, *128*, 107399. [CrossRef]
11. Kumar, G.S.; Premalatha, K. STIF: Intuitionistic fuzzy Gaussian membership function with statistical transformation weight of evidence and information value for private information preservation. *Distrib. Parallel Databases* **2023**, *41*, 233–266. [CrossRef] [PubMed]
12. Sathish Kumar, G.; Premalatha, K.; Uma Maheshwari, G.; Rajesh Kanna, P. No more privacy Concern: A privacy-chain based homomorphic encryption scheme and statistical method for privacy preservation of user's private and sensitive data. *Expert Syst. Appl.* **2023**, *234*, 121071. [CrossRef]
13. Zhang, W.; Zhang, H.; Zhang, B.; Yang, Y. An Identity-Based Authentication Model for Multi-domain in Grid Environment. In Proceedings of the 2008 International Conference on Computer Science and Software Engineering, Wuhan, China, 12–14 December 2008; Volume 3, pp. 165–169. [CrossRef]
14. Wang, W.; Hu, N.; Liu, X. BlockCAM: A Blockchain-Based Cross-Domain Authentication Model. In Proceedings of the 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC), Guangzhou, China, 18–21 June 2018; pp. 896–901. [CrossRef]
15. Shahidinejad, A.; Abawajy, J.H. Anonymous Blockchain-Assisted Authentication Protocols for Secure Cross-Domain IoD Communications. *IEEE Trans. Netw. Sci. Eng.* **2024**, *11*, 2661–2674. [CrossRef]
16. Wang, Z.; Lin, J.; Cai, Q.; Wang, Q.; Zha, D.; Jing, J. Blockchain-Based Certificate Transparency and Revocation Transparency. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 681–697. [CrossRef]
17. Chen, J.; Yao, S.; Yuan, Q.; He, K.; Ji, S.; Du, R. CertChain: Public and Efficient Certificate Audit Based on Blockchain for TLS Connections. In Proceedings of the IEEE INFOCOM 2018—IEEE Conference on Computer Communications, Honolulu, HI, USA, 16–19 April 2018; pp. 2060–2068. [CrossRef]
18. Kubilay, M.Y.; Kiraz, M.S.; Mantar, H.A. CertLedger: A new PKI model with Certificate Transparency based on blockchain. *Comput. Secur.* **2019**, *85*, 333–352. [CrossRef]
19. Sani, A.S.; Yuan, D.; Bao, W.; Yeoh, P.L.; Dong, Z.Y.; Vucetic, B.; Bertino, E. Xyreum: A High-Performance and Scalable Blockchain for IIoT Security and Privacy. In Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Dallas, TX, USA, 7–10 July 2019; pp. 1920–1930. [CrossRef]
20. Yang, X.; Li, W. A zero-knowledge-proof-based digital identity management scheme in blockchain. *Comput. Secur.* **2020**, *99*, 102050. [CrossRef]
21. Jiang, J.; Zhang, Y.; Li, J. A Blockchain-based Privacy-Preserving Scheme for Cross-domain Authentication. In Proceedings of the 2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Wuhan, China, 9–11 December 2022; pp. 992–999. [CrossRef]
22. Chen, J.; Zhan, Z.; He, K.; Du, R.; Wang, D.; Liu, F. XAuth: Efficient Privacy-Preserving Cross-Domain Authentication. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 3301–3311. [CrossRef]

23. Rosenberg, M.; White, J.; Garman, C.; Miers, I. zk-creds: Flexible Anonymous Credentials from zkSNARKs and Existing Identity Infrastructure. In Proceedings of the 2023 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 21–25 May 2023; pp. 790–808. [CrossRef]
24. Fan, L.; Guan, J.; Liu, K.; Wang, P. An Anonymous Authentication Scheme with Low Overhead for Cross-Domain IoT. In Proceedings of the Algorithms and Architectures for Parallel Processing, Tianjin, China, 20–22 October 2023; Tari, Z., Li, K., Wu, H., Eds.; Springer: Singapore, 2024; pp. 93–113.
25. Liu, H.; Luo, X.; Liu, H.; Xia, X. Merkle Tree: A Fundamental Component of Blockchains. In Proceedings of the 2021 International Conference on Electronic Information Engineering and Computer Science (EIECS), Changchun, China, 23–26 September 2021; pp. 556–561. [CrossRef]
26. Eberhardt, J.; Tai, S. ZoKrates-Scalable Privacy-Preserving Off-Chain Computations. In Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; pp. 1084–1091. [CrossRef]