IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 22 June 2023, accepted 30 July 2023, date of publication 7 August 2023, date of current version 15 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3302691

THEORY

A Privacy-Preserving Zero-Knowledge Proof for Blockchain

PO-WEN CHI, YUN-HSIU LU, AND ALBERT GUAN
Department of CSIE, National Taiwan Normal University, Taipei 11677, Taiwan
Corresponding author: Albert Guan (albert.zj.guan@gmail.com)

This work was supported in part by the National Science and Technology Council under Grant 110-2221-E-003-002-MY3.

ABSTRACT Zero-Knowledge Proof (ZKP) is a useful tools for proving that a prover possesses a secret without revealing it to the verifier. Designated Verifier Proof (DVP) is a special type of ZKP that adds the ability to restrict the identity of verifiers so that only pre-determined authorized verifiers can verify. However, DVP and other similar schemes do not work if the verifier provides some additional information to indicate the provenance of the proof. Since this information may be stored on the blockchain, the proof can be accepted by third parties even if the verifier is willing to protect the privacy of the prover. In this paper, we propose the concept of Blockchain Designated Verifier Proof (BDVP), and design a BDVP scheme suitable for blockchain applications. The key technique behind our BDVP scheme is that the verifier can forge a fake secret to simulate the proof. Therefore, a third party cannot determine whether the prover possesses the secret. This enables the verifier to protect the privacy of the prover, which is required by law or regulation. We also address the quantum attack problem and propose a post-quantum solution. We evaluate and compare the performances of the proposed protocol with other related protocols.

INDEX TERMS Zero-knowledge proof, privacy protection, chameleon hash function, non-transferable, quantum-resistance.

I. INTRODUCTION

A blockchain is a public, decentralized, distributed ledger in which records are chronologically ordered and consistent across network nodes. No one can modify the record without the consensus of the network nodes. When using blockchain, as with any other information system, laws and regulations dictate that private information must be properly protected. Zero-knowledge proofs (ZKP) can be used to protect private information in the blockchain. Through ZKP, the prover can prove the ownership of some secret without revealing the secret itself.

There are many applications based on ZKP. For example, Soewito et al. applied the ZKP in a wireless ad hoc network [1]. Alshameri et al. proposed an identification scheme based on ZKP for securing a software-defined network controller during the data and control plane communication [2]. Xi et al. built a mutual authentication system based on ZKP for the vehicle network [3].

The associate editor coordinating the review of this manuscript and approving it for publication was Barbara Masucci.

Recently, the application of ZKP on the blockchain has become popular [4]. Partala et al. survey several state-of-the-art ZKP schemes and their applications to confidential transactions and private smart contracts on blockchains [5]. Li et al. proposed a decentralized and location-aware architecture to address the data integrity along with the privacy-preserving issues in blockchain-based traffic management system integrates with ZKP protocol [6]. Yang et al. leverage the smart contracts and ZKP algorithms to improve the existing claim identity model in blockchain to realize the identity unlinkability, effectively avoiding the exposure of the ownership of attributes [7].

While ZKPs allow for secrecy, ZKPs cannot prevent a verifier from revealing to others the fact that the prover holds a secret. If the verifier reveals the proof to others, the verifier also discloses the prover's status. Taking the medical scenario as an example, through ZKP, patients can prove to the insurance company that they have a rare disease. If the insurance company passes the proof on to the company the prover is interviewing for a job, the illness may be a disadvantage to the prover. For this reason, we need a non-transferable

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

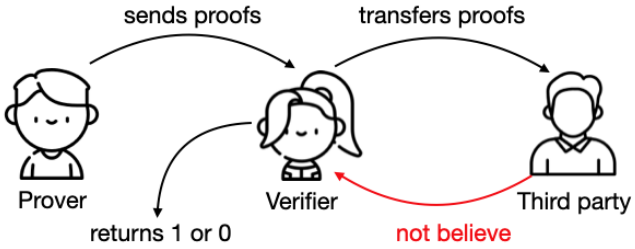

FIGURE 1. A demonstration of non-transferable ZKP. The prover on the left tries to convince the verifier in the middle. Even if the verifier is convinced that the prover possesses the secret, the verifier cannot convince a third party at the right that the prover possesses the secret.

zero-knowledge proof scheme. A simple scenario is shown in Figure 1.

There are existing schemes which have similar features to protect the prover's privacy. The first one is the designated verifier proof scheme (DVP). The DVP scheme is a ZKP that only allows the designated verifier to verify the proof. Therefore, third parties cannot be convinced by the proof. There are some research works on DVP such as [8], [9], [10], [11], and [12]. It is worth noting that Jakobsson mentioned two practical issues of this type of schemes [8]. The first one is the collusion problem where the verifier can share the same key with third party. The other problem is that if the verifier can persuade the third party to accept the source of the proof, then the verifier can transfer the proof to the third party. The other issue is that if the verifier can show the third party the transcript of the DVP, then the verifier can convince the third party that the prover has the secret, where the transcript of DVP is defined as all the data during the DVP execution. In the blockchain applications, these transcripts may be recorded in some block of the chain. This makes the current DVP not suitable for blockchain applications.

Another possible solution is the integration of ZKP and deniable authentication, which we call DAZKP. Deniable authentication allows the verifier to forge a signature with any messages. Integrated with ZKP, the verifier can forge a proof with a valid signature of the prover. Therefore, no one will accept the proof from the verifier. There are many related works about deniable authentication such as [13], [14], [15], and [16].

Back to the practical issues raised by Jakobsson et al. [8]. Undoubtedly, it is impossible to solve the collusion issue since the third party and the verifier have the same secrets and they can be treated as logically the same entity. Therefore, we assume that the verifier is willing to keep the secret of the prover which is required by law and regulations.

We first construct a ZKP scheme with the prover's signature. Empowering the verifier with the collision feature, the verifier can make a fake proof satisfying the existing rover's signature. If the verifier can convince a third party that the proof is from the prover, then the verifier can also make the third party accept the proof that the prover does not possess the secret. This idea can effectively solve the privacy-preserving issue in blockchain where DVP and DAZKP cannot solve this issue, assuming that the the transcripts of the DVP or DAZKP executions have been recorded in the blockchain.

In this paper, we refer to this privacy-preserving ZKP scheme as a blockchain designated validator proof (BDVP) scheme. The comparison between our work with other schemes are shown in Table 1.

TABLE 1. Comparison of various designated ZKPs.

	DVP [8]	DAZKP [15]	BDVP	NIQR-BDVP
Support Non-Malleable Record	X	X	O	O
Signature of the source	X	O	O	O
Non-Transferable	O	O	O	O
Hard Problem	DLP	DLP	DLP	Module-SIS

We list our contributions as follows:

- **Designation against trusted non-malleable records**
  In our scheme, a verifier can forge a valid signature even if the communication records are immutable, like in the blockchain scenario.
- **Quantum resistance**
  The proposed BDVP uses post-quantum cryptography algorithms, which means our scheme is secure in the era of quantum computing.
- **Performance evaluation**
  We analyze the computing performance of the proposed schemes. The results show that with the addition of non-negotiability, the cost is still acceptable.

A. ORGANIZATION OF THE PAPER

We first discuss related works in Section II. Next, preliminaries related to this research is given in Section III. We present a new DVP (BDVP) based on DLP in Section IV, and another DVP (NIQR-BDVP) based on Module-SIS for defending quantum attack in Section V. In Section VI, we present performance evaluations of BDVP and NIQR-BDVP, and the comparison between our proposed scheme and the related schemes. Conclusions and future work are given in Section VII.

II. RELATED WORKS

A. ZERO-KNOWLEDGE PROOF

In ZKP there are two parties, a prover and a verifier, that can communicate to each other. A valid ZKP protocol must satisfy three properties:

1) *Completeness*, which states that if the prover and the verifier follow the protocol, the verifier has a very high probability to accept the proof;
2) *Zero-Knowledge*, which prevents the verifier from learning any additional knowledge about the prover's secret from the execution of the protocol;
3) *Soundness*, which guarantees the prover can not fool the verifier into accepting the validity of a false statement.

Zero-knowledge proof (ZKP) was first proposed by Goldwasser, Micali and Rackoff [17] in 1985. Since then, many studies have been devoted to this research. In 1988, Goldreich et al. [18] showed a perfect ZKP for a decision problem. In 1989, Schnorr [19] proposed a well-known efficient interactive identification scheme and a related signature scheme that are based on discrete logarithm problems (DLP).

The classic ZKP protocol includes four algorithms:

- **Commit**() $\rightarrow r$
  The algorithm outputs a commitment r, which is used to verify the correctness of the proof about the secret s.
- **Challenge**() $\rightarrow e$
  The algorithm generates a random challenge string e, which the verifier sends to the prover.
- **Prove**$(e, \omega, k) \rightarrow s$
  The algorithm outputs a proof s computed by a given e, the witness ω, and a random string k.
- **Verify**$(r, e, s) \rightarrow \{1, 0\}$
  The verification algorithm outputs 1 if the verification result is correct; otherwise, 0.

In 1987, Santis et al. [20] proposed the first non-interactive ZKP (NIZKP). The security of their protocol relies on the fact that deciding quadratic residues in some multiplicative groups is computationally hard. In 1998, Gennaro et al. [21] proposed a more efficient non-interactive zero-knowledge proof scheme for quasi-safe prime products and other related problems. In 2006, Persiano et al. [22] presented a double-round NIZKP scheme. They showed that double-round NIZKP is more secure than one-round NIZKP. In 2008, Peikert et al. [23] proposed a NIZKP for lattice-based problems. In 2016, Martín-Fernández et al. [24] proposed an authentication scheme for IoT, which is based on NIZKP. Without interaction between devices, the performance of the authentication can be improved.

There are some research focused on quantum-resistant ZKP protocol. Xagawa and Tanaka first proposed a ZKP scheme for NTRU in 2009 [25]. In the design of a quantum resistant ZKP scheme, the quantum resistant commitment scheme must be used. Xagawa et al. used string-commitment scheme based on computational collision-resistant hash functions [26], [27], instead of using a quantum resistant commitment. In 2015, Cabarcas et al. proposed a post-quantum commitment scheme [28] based on lattice problems. They proved that their commitment scheme is statistically hiding and computationally binding in quantum computing. In 2018, Lyubashevsky et al. proposed an efficient commitment scheme based on some lattice problem which is computationally hard [29]. They also showed that their scheme can be more efficient when both hiding and binding properties are only computationally secure.

In 2022, Lyubashevsky et al. proposed a more general lattice-based ZKP scheme [30]. They presented a practical protocol based on the Module-SIS (short integer solution) and Module-LWE (learning with error) problems. This new proof system can be plugged into the construction of various lattice-based privacy protection in a black-box manner. In 2013, Xie et al. proposed a ZKP for Ring-LWE [31]. In 2019, Ma proposed a fully homomorphic commitment scheme with gadgets matrices computing and created a non-interactive ZKP scheme based on RLWE. The scheme is based on Peikert et al.'s scheme [23], who proposed a general lattice-based non-interactive ZKP scheme.

B. DESIGNATED VERIFIER PROOF

As mention above, ZKP cannot avoid revealing the fact that the prover has or owns the secret. ZKP with non-transferable property are required to protect prover's privacy. The following schemes realized the non-transferable property.

In 1996, Jakobsson et al. [8] proposed a protocol with designated verifier. They included the undeniable signature as the public information. In 2003, Steinfeld et al. proposed an application of the general idea of DVP. They called it designated verifier signature scheme [32]. In 2009, Wang et al. [9] proposed a non-interactive deniable authentication scheme based on designated verifier. They showed that their scheme is both deniable and unforgettable against a probabilistic polynomial time adversary, and the performance of this scheme is better than other related schemes. In 2018, Chaidos et al. [11] further improved the performance of the system. Their schemes allow efficiently extracting large exponents without harming the efficiency of the proof. In 2021, Campanelli et al. [12] proposed a more succinct publicly-certifiable proofs.

C. DENIABLE AUTHENTICATION

In 1998, Aumann et al. [13] proposed a deniable authentication scheme. They added a party, inquisitor *INQ*, to the communication between sender S and receiver R. *INQ* cannot prove that the message M was authored by S, and also R cannot prove the fact to a third party that M was authored by S. In 2002, Fan et al. [14] proposed a deniable authentication protocol based on Diffie-Hellman problem. In 2007, Lee et al. [15], proposed a deniable authentication protocol based on ElGamal signature scheme, and enable to forge a fake signature having the same authenticator as the original signature. In 2011, Tian et al. showed that deniable authentication protocols can also be non-interactive [33]. In 2019, Zhu et al. proposed a deniable authentication scheme in the cloud-based pay-TV system [16]. They designed a deniable authentication protocol that did not allow the pay-TV system to prove video contents that the user has watched to a third party over an unsecured network. Recently, Zeng et al. proposed a protocol, called the privacy-preserving authentication protocol, to deny the participants' involvement, in which even the sender is blind to the receiver. They also explained how to apply it to a privacy-preserving Wi-Fi system to prevent location leakage.

While these are non-transferable ZKP protocols, the public record in the blockchain allows third parties to know some information about the protocol, which makes the

non-transferable property of the ZKP no longer holds. In this paper, we propose a solution to the problem.

III. PRELIMINARY

A ZKP scheme involves an interactive communication between two parties: a prover **P** and a verifier **V**. **P** wants to show **V** that the verifier knows a secret x but without revealing it to **V**. In this section some ZKP schemes and chameleon hash functions which are used in the proposed ZKP schemes are briefly described.

A. SCHNORR'S ZKP

We first review the Schnorr's ZKP [19] for the decision version of the discrete logarithm problem, which is known to be computationally hard. Let $\mathbb{Z}_q^*$ be a finite multiplicative group, and g be a generator for $\mathbb{Z}_q^*$. **P** wants to prove that the verifier knows x such that $y = g^x$. The ZKP can be described as follows.

1) **P** makes a commitment $r \leftarrow g^k$ with a random value $k \in \mathbb{Z}_q^*$ to **V**.
2) **V** sends a challenge c to **P**, where c is randomly chosen from the challenge space $\mathbb{C}$.
3) **P** computes a proof s and sends it to **V**, where $s \leftarrow k + c \cdot x \bmod q$.
4) **V** outputs 1 if $r = g^s y^{-c} \bmod q$; otherwise, 0.

Theorem 1 (Schnorr [19]): Schnorr's ZKP scheme is complete, sound, and zero-knowledge if discrete logarithm problem is computationally hard.

B. LYUBASHEVSKY'S ZKP

The Lyubashevsky's ZKP [30] is described briefly as follows. The shortest non-zero vector problem (SVP) is:

> Given a basis **B** of a lattice **L**, find a nonzero vector **x** whose length is the shortest among all non-zero vectors in **L**.

The shortest non-zero vector problem is NP-hard. It does not have polynomial-time algorithm, unless $P = NP$. The shortest vector problem is also shown to be hard in average-case. Furthermore, no known quantum algorithms can solve this problem efficiently.

Lyubashevsky proposed a quantum-resistant ZKP scheme based on the shortest non-zero vector problem. The protocol is briefly described as follows.

1) **P** chooses random vectors $(\mathbf{y}_1, \mathbf{y}_2)$ from discrete normal distributions $\mathcal{N}_{\mathfrak{s}_1}^{k_1}$ and $\mathcal{N}_{\mathfrak{s}_2}^{k_2}$. **P** then computes a commitment $\mathbf{w} = \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2$, and sends $\mathbf{w}$ to **V**.
2) **V** sends a challenge c to **P**.
3) After receiving the challenge c from **V**, **P** computes two proofs $\mathbf{z}_1 = \mathbf{y}_1 + c\mathbf{r}$, and $\mathbf{z}_2 = \mathbf{y}_2 + x\mathbf{x}$, and sends $(\mathbf{z}_1, \mathbf{z}_2)$ to **V** for verification.
4) **V** verifies $\|\mathbf{z}_i\|_2 \leq 2\sqrt{N}$, $i = 1, 2$, $\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 - c\mathbf{t} = \mathbf{w}$, and outputs 1 if all the above equalities are true; otherwise, 0.

Theorem 2 (Lyubashevsky [29]): Lyubashevsky's QRZKP scheme is complete, sound, zero-knowledge, and quantum-resistant if the shortest vector problem is computationally hard in quantum computation model.

C. PEIKERT'S NIZKP SCHEME

Let $\Lambda = \mathcal{L}(\mathbf{B})$ be an n-dimensional lattice generated by a basis **B**. A prover **P** wants to prove that a vector **v** is sampled from $D_{\Lambda,-\mathbf{t}}$, where **t** is chosen uniformly at random from $\mathcal{P}(\mathbf{B})$. The protocol runs as follows:

- **Common Input**
  A basis **B** of an n-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{B})$.
- **Random Input**
  A vector $\mathbf{t} \in \mathbb{R}^n$ is chosen uniformly at random from $\mathcal{P}(\mathbf{B})$.
- **Prover P**
  Sample $\mathbf{v} \approx D_{\Lambda,-\mathbf{t}}$ and output $\mathbf{e} = \mathbf{t} + \mathbf{v} \in \mathbb{R}^n$ as the proof.
- **Verifier V**
  Accept if $\mathbf{e} - \mathbf{t} \in \Lambda$ and $\|e\| \leq \sqrt{n}$; otherwise, reject.

Similar to the previous results, we have to ensure the proof is sent from **P**, so we combine Peikert's NIZKP, Lyubashevsky's QRZKP [23], and a secure signature scheme. The signature protocol runs as follows:

- **Setup**() $\rightarrow (\mathbf{w}, \sigma_\mathbf{w}, \mathbf{k})$:
  P first generates a random polynomials $\mathbf{k} \leftarrow (\mathbf{y}_1, \mathbf{y}_2)$ from discrete normal distributions $\mathcal{N}_{\mathfrak{s}_1}^{k_1}$ and $\mathcal{N}_{\mathfrak{s}_2}^{k_2}$, and computes the commitment $\mathbf{w} \leftarrow \mathbf{A}_1 \cdot \mathbf{y}_1 + \mathbf{A}_2 \cdot \mathbf{y}_2$ and the corresponding signature $\sigma_\mathbf{w} \leftarrow \mathcal{S}.\mathbf{Verify}(\mathbf{w})$. **P** keeps the secret **k** and publishes $(\mathbf{w}, \sigma_\mathbf{w})$.
- **Prove**$(\mathbf{w}, \sigma_\mathbf{w}, (\mathbf{r}, \mathbf{x}), \mathbf{k}) \rightarrow (\mathbf{z}, \sigma_\mathbf{z})$:
  P computes a hash value
  $\mathbf{c} = H(\mathbf{w}||\mathbf{A}_1||\mathbf{A}_2||\mathbf{t}||\sigma_\mathbf{w})$. **P** then computes a proof $\mathbf{z} \leftarrow (\mathbf{z}_1, \mathbf{z}_2) \leftarrow (\mathbf{y}_1 + \mathbf{c} \cdot \mathbf{r}, \mathbf{y}_2 + \mathbf{c} \cdot \mathbf{x})$, and the proofs should be checked to be rejection in the sampling. Without aborting the sampling, **P** publishes $(\mathbf{z}, \sigma_\mathbf{z})$, where $\sigma_\mathbf{z}$ is the signature of **z**.
- **Verify**$(\mathbf{w}, \mathbf{c}, \mathbf{z}, \sigma_\mathbf{w}, \sigma_\mathbf{z}) \rightarrow \{1, 0\}$:
  Every verifier can run **Verify**() and check the verification. **Verify**() accepts the transcript if:
  - $\mathcal{S}.\mathbf{Verify}(\mathbf{w}, \sigma_\mathbf{w}) \stackrel{?}{=} 1$
  - $\mathcal{S}.\mathbf{Verify}(\mathbf{z}_1, \sigma_{\mathbf{z}_1}) \stackrel{?}{=} 1$
  - $\mathcal{S}.\mathbf{Verify}(\mathbf{z}_2, \sigma_{\mathbf{z}_2}) \stackrel{?}{=} 1$
  - $\|\mathbf{z}_1\| \stackrel{?}{\leq} 2\mathfrak{s}_1\sqrt{N}$
  - $\|\mathbf{z}_2\| \stackrel{?}{\leq} 2\mathfrak{s}_2\sqrt{N}$
  - $\mathbf{A}_1 \cdot \mathbf{z}_1 + \mathbf{A}_2 \cdot \mathbf{z}_2 - \mathbf{c} \cdot \mathbf{t} \stackrel{?}{=} \mathbf{w}$

According to the above protocol, all verifiers can verify and identify the proof, and can also transfer the proof to others.

D. CHAMELEON HASH FUNCTION

Chameleon hash function (CH) is a special type of hash function [34]. A hash function is usually collision-resistant, which means that it is computationally hard to find different inputs with the same hash value. In CH, collisions can be

found with trapdoor key, but without the trapdoor, the CH is still collision-resistant. A CH consists of three algorithms:

- $\mathbf{Gen}(1^\lambda) \rightarrow (pk, t, \rho)$
  On input security parameter λ, the algorithm computes the public key pk and a random string ρ for computing the hash value; the trapdoor t for computing the collision pair of the CH.
- $\mathbf{CH}(pk, m, \rho) \rightarrow h$
  The algorithm outputs a hash value h, with the input public key pk, a message m, and the string ρ.
- $\mathbf{UF}(t, m, \rho, m') \rightarrow \rho'$
  The algorithm computes a string ρ' with trapdoor t, a given random message m', the original transcript (m, ρ), such that $\mathbf{CH}(pk, m, \rho) = \mathbf{CH}(pk, m', \rho')$.

In 1998, Krawczyk and Rabin [34] introduced the CH and created CH signatures based on DLP. They mentioned that a CH has three properties:

- **Collision resistance**:
  Using only the public key pk, there is no efficient algorithm to find two pairs (m, ρ) and (m', ρ'), where $m \neq m'$ such that $\mathbf{CH}(pk, m, \rho) = \mathbf{CH}(pk, m', \rho')$, except with negligible probability.
- **Trapdoor collisions**:
  Using the secret key t, for any pair (m, ρ) and any additional message m', there is an efficient algorithm for finding ρ' such that $\mathbf{CH}(pk, m, \rho) = \mathbf{CH}(pk, m', \rho')$.
- **Uniformity**:
  All messages m induce the same probability distribution on $\mathbf{CH}(pk, m, \rho)$ for ρ chosen uniformly at random.

In 2005, Ateniese et al. [35] mentioned that the Chaum-Pedersen trapdoor commitment has key exposure problem. The signer who knows about the CH trapdoor can collaborate with others to deny any signatures that are designated to be verified by the same public key. Since then, different CH schemes have been proposed. In 2009, a handover authentication scheme using a chameleon-hashing-based credential was proposed by Choi et al. [36]. They applied the CH to achieve an efficient transfer and lower energy consumption. In 2010, Mohassel et al. [37] transformed every chameleon hash function to a strongly unforgeable one-time signature scheme. They proposed a new computationally hard problem on lattice structure. In 2013, Guo et al. [38] proposed an elliptic curve based CH in vehicular ad hoc networks. Their protocol can achieve mutual authentication with a much lower computational cost, and showed that it is suitable for a realistic vehicular environment. In 2021, Wu et al. [39] introduced a quantum-resistant key-exposure-free chameleon hash and its applications on a reducible blockchain. In their system, the structure of the blockchain is still correct, but the contents of the block are modified with the same hash value by the properties of the CH.

IV. PRIVACY-PRESERVING ZERO-KNOWLEDGE PROOF

In this section, we construct a ZKP with privacy-preserving designated verifier proof (BDVP) using DLP. The BDPV scheme can protect user's privacy when used in blockchain. The Chameleon hash function is used to obtain collisions and thus allow verifiers to forge a proof. The verifier cannot be trusted by any third party because the verifier holds the trapdoor key for the hash function. The DLP-based BDVP scheme is based on Schnorr's ZKP scheme, plus a signature scheme.

We first define the following five algorithms which will be used in our scheme. We use bold letters to denote an algorithm, such as **Publish**; uppercase bold letters represent a party, such as prover **P**; uppercase bold letters with subscripts represent matrices, such as $\mathbf{A}_1$; Bold lower case letters representing tuples or vectors, such as $\mathbf{r}$ or $\mathbf{x}$. Blackboard bold capital letters are used to denote a group, such as $\mathbb{G}$, and lower case letters to denote a value or string, such as t or ρ.

- $\mathbf{Publish}(1^\lambda) \rightarrow (\mathbf{p}, t)$:
  On input the security parameters λ, the algorithm outputs the public key $\mathbf{p}$ and the trapdoor key t.
- $\mathbf{Commit}(\mathbf{p}) \rightarrow (\mathbf{r}, \sigma_\mathbf{r}, \mathbf{k})$:
  Based on the public key $\mathbf{p}$, the algorithm generates commitment $\mathbf{r}$ and signature $\sigma_\mathbf{r}$ of $\mathbf{r}$, where the commitment is equivalent to the prover's knowledge. It also outputs a random string $\mathbf{k}$.
- $\mathbf{Challenge}() \rightarrow \mathbf{c}$
  The algorithm outputs a random challenge $\mathbf{c}$.
- $\mathbf{Prove}(\mathbf{c}, \mathbf{x}, \mathbf{k}) \rightarrow (\mathbf{s}, \sigma_\mathbf{s})$
  The algorithm outputs a proof $\mathbf{s}$ computed for the given challenge $\mathbf{c}$, knowledge $\mathbf{x}$, and a set of random string $\mathbf{k}$, together with the signature $\sigma_\mathbf{s}$ of $\mathbf{s}$.
- $\mathbf{Verify}(\mathbf{p}, \mathbf{r}, \mathbf{c}, \mathbf{s}, \sigma_\mathbf{r}, \sigma_\mathbf{s}) \rightarrow \{1, 0\}$
  The verification algorithm outputs 1 if the verification is correct; otherwise, 0.

In order to allow collisions, the prover **P** needs to use a CH in the commitment step. The verifier **V** also knows the trapdoor key of the hash function, but no third parties know the trapdoor key t.

In the construction of the protocol, we have to make sure that the proof actually comes from **P**. We modify Schnorr's ZKP scheme into a signature ZKP scheme (SZKP) as shown in Figure 2. In the figure, for example, $k \overset{\$}{\leftarrow} \mathbb{Z}_q^*$ means randomly selects a number in $\mathbb{Z}_q^*$ and assign it to k. A $\leftarrow$ or a $\rightarrow$ with variables on them means send the values of these variable to the other party.

A classical ZKP or SZKP is transferable, especially when it is used in the blockchain environment. This is because a third party **T**, can check with the proofs recorded in the blockchain. In our proposed protocol, we empower the verifier **V** to make a faked proof that the prover does not have the secret x.

Let $g \in \mathbb{G}$, and q be a large prime, $\mathbb{Z}_q^* = \mathbb{Z}/q\mathbb{Z}$, and $\mathbb{C}$ is the challenge space. **P** wants to prove the knowledge x such that $y = g^x$. Given a secure signature scheme $\mathcal{S}$, which has two algorithms **Sign**() and **Verify**(), $\mathcal{S}$.**Sign**() outputs a signature σ_m on input message m, denoted $\sigma_m \leftarrow \mathcal{S}.\mathbf{Sign}(m)$. $\mathcal{S}$.**Verify**() verifies if $\sigma_m = \mathcal{S}.\mathbf{Sign}(m)$, denotes $\{1, 0\} \leftarrow \mathcal{S}.\mathbf{Verify}(m, \sigma_m)$.

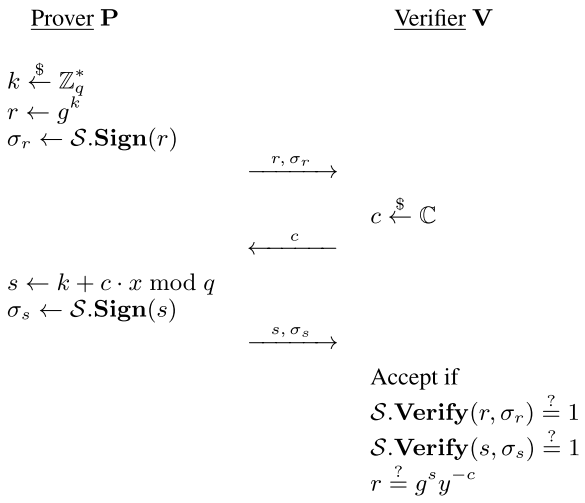

FIGURE 2. Signature ZKP protocol.

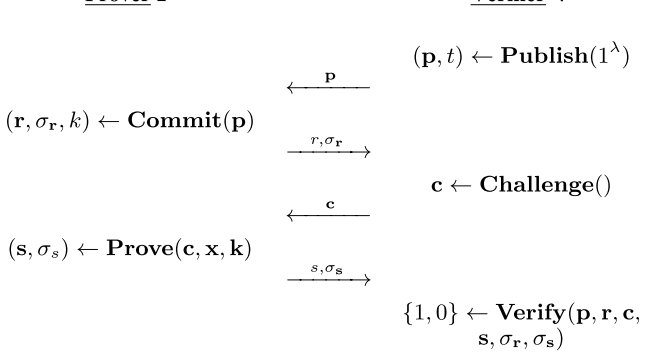

FIGURE 3. BDVP scheme.

The protocol is shown in Figure 3, and it runs as follows:

- **Publish**$(1^\lambda) \rightarrow (\mathbf{p}, t)$:
  - $(pk, t, \rho) \leftarrow \mathbf{Gen}(1^\lambda)$, where pk is public key, t is trapdoor, and ρ is a random string.
  - $\mathbf{p} \leftarrow (\mathbf{CH}, pk, \rho)$.
  - **V** sends $\mathbf{p}$ to **P** and stores t.
- **Commit**$(\mathbf{p}) \rightarrow (r, \sigma_r, k)$:
  - $r \leftarrow \mathbf{CH}(pk, g^k, \rho)$, where k is a random string in $\mathbb{Z}_q^*$.
  - $\sigma_r \leftarrow \mathcal{S}.\mathbf{Sign}(r)$.
  - **P** stores k and sends (r, σ_r) to **V**.
- **Challenge**$() \rightarrow c$:
  - **V** randomly chooses a random string $c \in \mathbb{C}$ and sends it to **P**.
- **Prove**$(c, x, k) \rightarrow (s, \sigma_s)$:
  - $s \leftarrow k + c \cdot x$.
  - $\sigma_s = \mathcal{S}.\mathbf{Sign}(s)$
  - **P** sends (s, σ_s) to **V**.
- **Verify**$(\mathbf{p}, r, c, s, \sigma_r, \sigma_s) \rightarrow \{1, 0\}$:
  V now runs **Verify**() with all information received and check if:
  1) $\mathcal{S}.\mathbf{Verify}(r, \sigma_\mathbf{r}) \stackrel{?}{=} 1$
  2) $\mathcal{S}.\mathbf{Verify}(s, \sigma_\mathbf{s}) \stackrel{?}{=} 1$
  3) $r \stackrel{?}{=} \mathbf{CH}(pk, g^s y^{-c}, \rho)$

  Verify() accepts the transcript if and only if the three conditions are all true.

In the remainder of this section, we show that our proposed scheme is complete, sound, zero-knowledge, and non-transferable.

Theorem 3: The proposed scheme is complete.

Proof: If **P** knows the knowledge x, it implies $s = k + xc \bmod q$. The protocol proceeds as follows.

1) **V** publishes $\mathbf{p} = (\mathbf{CH}, pk, \rho)$ and stores t generated by **Publish**.
2) **P** computes r and σ_r.
3) **V** sends c by **Challenge**() to **P**.
4) **P** computes s and σ_s by **Prove**() and gives them to **V**.
5) **V** runs **Verify**() and obtains 1 due to the following:

The conditions (1) and (2) are obvious true, since $\sigma_r = \mathcal{S}.\mathbf{Sign}(r)$ and $\sigma_s = \mathcal{S}.\mathbf{Sign}(s)$.

Proof of (3): $\mathbf{CH}(pk, g^s y^{-c}, \rho) = \mathbf{CH}(pk, g^k g^{x \cdot c} g^{-x \cdot c}, \rho) = \mathbf{CH}(pk, g^k, \rho) = r$. □

Theorem 4: The proposed scheme is sound if DLP is computationally hard and the hash function is collision resistance.

Proof: The proof is by contradiction. Assume that we can find two accepting transcripts (r, c, s) and (r, c', s') in two protocol runs, with the same commitment but different challenges. Then we can extract the knowledge x by $x = (s - s')/(c - c')$. □

Theorem 5: The proposed scheme is zero-knowledge if DLP is computationally hard and the signature is secure.

Proof: Assume there are several valid transcripts. In a signature scheme, **V** can query **P** for a signature corresponding to a message many times. In **V**'s view, **V** does not have x but can simulate many valid transcripts. The simulation runs as follows:

- **V** randomly chooses s' and queries **P** for σ_s'.
- $c' \stackrel{\$}{\leftarrow} \mathbb{C}$.
- $r' \leftarrow \mathbf{CH}(pk, g^{s'} y^{-c'}, \rho)$.
- **V** queries **P** for σ_r'.
- Output the transcript $(r', c', s', \sigma_r', \sigma_s')$.

$$\mathbf{P} \rightarrow \mathbf{V} : (r', \sigma_r')$$
$$\mathbf{V} \rightarrow \mathbf{P} : c'$$
$$\mathbf{P} \rightarrow \mathbf{V} : (s', \sigma_s')$$

As mentioned above, although **V** can pass the verification by simulating many valid transcripts (r_i, c_i, s_i), **V** still knows nothing about the knowledge from the transcripts. □

Theorem 6: The proposed scheme is privacy-preserving if DLP is computationally hard and the hash function is a chameleon hash function.

Proof: Assume **P** has a knowledge x', but no one can ensure that if x' is equivalent to x or not. We first run the protocol for several steps:

- **V** publishes $\mathbf{p} = (\mathbf{CH}, pk, \rho)$ and stores t generated by **Publish**().
- **P** computes r and σ_r, and gives them to **V**.

- **V** sends c to **P**.
- **P** computes s and σ_s with x', and gives them to **V**.

Due to *trapdoor collisions*, **V** can choose a string ρ':

$$\rho' = \mathbf{U}F(t, g^k, \rho, g^s y^{-c}),$$

such that $\mathbf{CH}(pk, g^k, \rho) = \mathbf{CH}(pk, g^s y^{-c}, \rho')$, so all signatures will be the same, and the verification is always passed. Due to *trapdoor collisions*, when receiving the information from **P**, **V** can always pass the verification if **P** has knowledge x with forging a string ρ'. Let $\mathbf{p}' = (\mathbf{CH}, pk, \rho')$ be the probability of non-transferable is:

$$Pr[\mathbf{Verify}(\mathbf{p}', r, c, s, \sigma_r, \sigma_s) = 1] \geq 1 - \epsilon(\lambda)$$

□

Finally, we have

Theorem 7: The BDVP protocol shown in Figure 3 is a valid privacy-preserving ZKP, assuming DLP is computationally hard.

V. LATTICE-BASED NI-BDVP

Integer factorization problem, discrete logarithm problem, and many other computationally hard problems can be solved with quantum computers by Shor's algorithm [40]. With the advent of quantum computers, post-quantum cryptography (PQC) has become increasingly important. In this section, we improve Lyubashevsky's scheme [30], which is based on Module-SIS, and show that our proposed scheme can be modified to be quantum resistant. The modification is based on lattice problems, and we also reduces the number of interactions in the ZKP protocol. We call this protocol NIQR-BDVP.

We first define the four algorithms used in the new scheme as follows.

- **KeyGen**$(1^\lambda) \rightarrow (\mathbf{p}_{V_i}, t_{V_i})$
  On input the security parameter λ, the algorithm outputs the public key $\mathbf{p}_{V_i}$ and the trapdoor key t_{V_i}.
- **Setup**$(\mathbf{p}_{V_i}) \rightarrow (\mathbf{r}, \sigma_\mathbf{r}, \mathbf{k})$
  The algorithm generates commitments $\mathbf{r}$ and a signature $\sigma_\mathbf{r}$ of $\mathbf{r}$, where the commitment is equivalent to prover's knowledge. It also generates a random string $\mathbf{k}$.
- **Prove**$(\mathbf{r}, \sigma_\mathbf{r}, \mathbf{x}, \mathbf{k}) \rightarrow (\mathbf{c}, \mathbf{s}, \sigma_\mathbf{s})$
  The algorithm outputs $\mathbf{c}$, a proof $\mathbf{s}$ computed by the given $\mathbf{c}$, knowledge $\mathbf{x}$, and a random string $\mathbf{k}$. It also outputs a signature $\sigma_\mathbf{s}$ of $\mathbf{s}$.
- **Verify**$(\mathbf{p}, \mathbf{r}, \mathbf{c}, \mathbf{s}, \sigma_\mathbf{r}, \sigma_\mathbf{s}) \rightarrow \{1, 0\}$
  The verification algorithm outputs 1 if the verification is correct; otherwise, output 0.

Every participant runs **KeyGen**() to generate his/her public key $\mathbf{p}_{V_i} = (pk_{V_i}, \mathbf{CH}_{V_i}, \rho_{V_i})$. When **P** wants to prove the knowledge to a certain verifier $\mathbf{V}_1$, **P** has to select $\mathbf{p}_{V_1}$, and gives the commitment and the proof to verifier $\mathbf{V}_1$. Every verifier can verify the proof, but they all fail except $\mathbf{V}_1$. Moreover, if $\mathbf{V}_1$ wants to protect the privacy of the prover, $\mathbf{V}_1$ can forge signatures because the verifier knows the trapdoor t_{V_1}.

A. CONSTRUCTION OF THE PROTOCOL

The NIQR-BDVP system contains a secure signature scheme $\mathcal{S}$, and every party has a quantum-resistant CH as $\mathrm{CH}_{V_i} = (\mathbf{Gen}_{V_i}, \mathbf{CH}_{V_i}, \mathbf{UF}_{V_i})$. Each party runs **KeyGen**() first to compute their trapdoor keys t_{V_i}, public keys pk_{V_i}, and a random string ρ_{V_i}. For example, a prover **P** has trapdoor key t_P, public key pk_P and a random string ρ_P. Assume the prover **P** wants to prove the knowledge $(\mathbf{r}, \mathbf{x})$ such that

$$\mathbf{t} = \mathbf{A}_1 \cdot \mathbf{r} + \mathbf{A}_2 \cdot \mathbf{x},$$

And if there is only a certain verifier $\mathbf{V}_1$ that can verify the proof, the protocol runs as follows:

- $\forall a \in \mathbb{P}$, **KeyGen**$(1^\lambda) \rightarrow (\mathbf{p}_{V_a}, t_{V_a})$:
  -- $(pk_{V_a}, \mathbf{CH}_{V_a}, \rho_{V_a}) \rightarrow \mathbf{Gen}(1^\lambda)$.
  -- $\mathbf{p}_{V_a} \leftarrow (pk_{V_a}, \mathbf{CH}_{V_a}, \rho_{V_a})$.
  -- Everyone stores t_{V_a} and publishes $\mathbf{p}_{V_a}$.
- **Setup**$(\mathbf{p}_{V_1}) \rightarrow (\mathbf{w}, \sigma_\mathbf{w}, \mathbf{k})$:
  -- $\mathbf{y}_1, \mathbf{y}_2 \leftarrow \mathcal{N}^{k_1}_{\mathfrak{s}_1}, \mathcal{N}^{k_2}_{\mathfrak{s}_2}$.
  -- $\mathbf{k} \leftarrow (\mathbf{y}_1, \mathbf{y}_2)$
  -- **P** computes a commitment $\mathbf{w} \leftarrow \mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1 \cdot \mathbf{y}_1 + \mathbf{A}_2 \cdot \mathbf{y}_2, \rho_{V_1})$.
  -- $\sigma_\mathbf{w} \leftarrow \mathcal{S}.\mathbf{Sign}(\mathbf{w})$
  -- **P** stores $\mathbf{k}$ and publishes $(\mathbf{w}, \sigma_\mathbf{w})$.
- **Prove**$(\mathbf{w}, \sigma_\mathbf{w}, (\mathbf{r}, \mathbf{x}), \mathbf{k}) \rightarrow (\mathbf{z}, \sigma_\mathbf{z})$:
  -- **P** computes a hash value $\mathbf{c} = \mathbf{CH}_P(pk_P, \mathbf{w}||\mathbf{A}_1||\mathbf{A}_2||\mathbf{t}||\sigma_\mathbf{w}, \rho_r)$ with a random string ρ_r.
  -- $\mathbf{z}_1, \mathbf{z}_2 \leftarrow \mathbf{y}_1 + \mathbf{c} \cdot \mathbf{r}, \mathbf{y}_2 + \mathbf{c} \cdot \mathbf{x}$.
  -- $\mathbf{z} \leftarrow (\mathbf{z}_1, \mathbf{z}_2)$.
  -- $\sigma_\mathbf{z} \leftarrow (\sigma_{\mathbf{z}_1}, \sigma_{\mathbf{z}_2})$.
  -- **P** publishes $((\rho_r, \mathbf{c}), \mathbf{z}, \sigma_\mathbf{z})$.
- **Verify**$(\mathbf{p}_{V_i}, \mathbf{w}, \mathbf{c}, \mathbf{z}, \sigma_\mathbf{w}, \sigma_\mathbf{z}) \rightarrow \{1, 0\}$:
  Every verifier can run **Verify**() and check the verification. **Verify**() accepts the transcript if:
  -- $\mathcal{S}.\mathbf{Verify}(\mathbf{w}, \sigma_\mathbf{w}) \stackrel{?}{=} 1$
  -- $\mathcal{S}.\mathbf{Verify}(\mathbf{z}_1, \sigma_{\mathbf{z}_1}) \stackrel{?}{=} 1$
  -- $\mathcal{S}.\mathbf{Verify}(\mathbf{z}_2, \sigma_{\mathbf{z}_2}) \stackrel{?}{=} 1$
  -- $\|\mathbf{z}_1\| \stackrel{?}{\leq} 2\mathfrak{s}_1\sqrt{N}$
  -- $\|\mathbf{z}_2\| \stackrel{?}{\leq} 2\mathfrak{s}_2\sqrt{N}$
  -- $\mathbf{CH}_{V_i}(pk_{V_i}, \mathbf{A}_1 \cdot \mathbf{z}_1 + \mathbf{A}_2 \cdot \mathbf{z}_2 - \mathbf{c} \cdot \mathbf{t}, \rho_{V_i}) \stackrel{?}{=} \mathbf{w}$

  Of course, every verifier can only use their own information $\mathbf{p}_{V_i}$ to run the verification. However, there is only one verifier $\mathbf{V}_1$ that can pass the verification.

B. PROOFS

We show that our proposed protocol is an NIQR-BDVP. That is, our scheme is complete, sound, zero-knowledge, and non-transferable.

Theorem 8: The proposed scheme is complete.

Proof: The protocol proceeds as follows.

1) Each party runs **KeyGen**() to generate $(pk_{V_i}, t_{V_i}, \rho_{V_i})$, stores the trapdoor key t_{V_i} and publishes public information $\mathbf{p}_{V_i} \leftarrow (\mathbf{CH}_{V_i}, pk_{V_i}, \rho_{V_i})$.

2) **P** computes $\mathbf{w}, \sigma_{\mathbf{w}}$, and $\mathbf{k} \leftarrow (\mathbf{y}_1, \mathbf{y}_2)$.
3) **P** computes $(\mathbf{c}, \mathbf{z}_1, \mathbf{z}_2)$ and $(\sigma_{\mathbf{z}_1}, \sigma_{\mathbf{z}_2})$ by **Prove**() and publishes them.
4) $\mathbf{V}_i$ runs **Verify**() and outputs 1 due to the following:

- The verification of the signature pairs is obviously true.
- If $(\mathbf{z}_1, \mathbf{z}_2)$ does not abort with $\mathbf{Rej}_i$, and $\|z_i\|_2 \leq 2\mathfrak{s}_i\sqrt{N}$ is always valid.
- if $i = 1$:

$$\begin{aligned}&\mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1 \cdot \mathbf{z}_1 + \mathbf{A}_2 \cdot \mathbf{z}_2 - \mathbf{c} \cdot \mathbf{t}, \rho_{V_1})\\ &\quad = \mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1 \cdot (\mathbf{y}_1 + \mathbf{c} \cdot \mathbf{r}) + \mathbf{A}_2 \cdot (\mathbf{y}_2 + \mathbf{c} \cdot \mathbf{x})\\ &\qquad - \mathbf{c} \cdot (\mathbf{A}_1 \cdot \mathbf{r} + \mathbf{A}_2 \cdot \mathbf{x}), \rho_{V_1})\\ &\quad = \mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1 \cdot \mathbf{y}_1 + \mathbf{A}_2 \cdot \mathbf{y}_2, \rho_{V_1}) = \mathbf{w}\end{aligned}$$

else: $(i \neq 1)$

$$\mathbf{CH}_{V_i}(pk_{V_i}, \mathbf{A}_1 \cdot \mathbf{z}_1 + \mathbf{A}_2 \cdot \mathbf{z}_2 - \mathbf{c} \cdot \mathbf{t}, \rho_{V_i}) \neq \mathbf{w}$$

When the NIQR-BDVP is complete, the following probability is satisfied.

$$Pr[\mathbf{Verify}(\mathbf{p}_{V_1}, \mathbf{w}, \mathbf{c}, \mathbf{z}, \sigma_{\mathbf{w}}, \sigma_{\mathbf{z}}) = 1] \geq 1 - \epsilon(\lambda)$$

□

Theorem 9: The proposed scheme is sound if Theorem 7 holds, and the shortest vector problem is computationally hard.

Proof: The proof is by contradiction. Notice that the two accepting transcripts $(\mathbf{w}, c, \mathbf{z}_1, \mathbf{z}_2)$ and $(\mathbf{w}, c', \mathbf{z}'_1, \mathbf{z}'_2)$ allow the computation of:

$$\begin{aligned}\mathbf{w} &= \mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2, \rho_{V_1})\\ &= \mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1(\mathbf{z}_1 - \mathbf{cr}) + \mathbf{A}_2(\mathbf{z}_2 - \mathbf{cx}), \rho_{V_1})\\ &= \mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1(\mathbf{z}'_1 - \mathbf{c}'\mathbf{r}) + \mathbf{A}_2(\mathbf{z}'_2 - \mathbf{c}'\mathbf{x}), \rho_{V_1}),\end{aligned}$$

E can extract $(\mathbf{r}, \mathbf{x})$ with $\left(\frac{\mathbf{z}_1 - \mathbf{z}'_1}{\mathbf{c} - \mathbf{c}'}, \frac{\mathbf{z}_2 - \mathbf{z}'_2}{\mathbf{c} - \mathbf{c}'}\right)$. **E** aborts only when it finds two soundly accepting transcripts or it exhausts the challenge space [41], [42]. □

Theorem 10: The proposed scheme is zero-knowledge if Theorem 7 and the shortest vector problem is computationally hard.

Proof: This property can be referred to as Theorem 5, **c** is computed by $\mathbf{CH}_{V_1}(pk_{V_1}, a, \rho_r)$ with a random strings a and ρ_r, the simulation runs as follows:

- $\mathbf{V}_1$ randomly chooses a and ρ_r and computes $\mathbf{c} \leftarrow \mathbf{CH}_{V_1}(pk_{V_1}, a, \rho_r)$.
- $\mathbf{V}_1$ randomly chooses $\mathbf{z}'$ and queries **P** for $\sigma'_{\mathbf{z}}$.
- $\mathbf{w}' \leftarrow \mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1 \cdot \mathbf{z}_1 + \mathbf{A}_2 \cdot \mathbf{z}_2 - \mathbf{c} \cdot \mathbf{t}, \rho_{V_1})$.
- $\mathbf{V}_1$ queries **P** for $\sigma'_{\mathbf{w}}$.
- $\mathbf{V}_1$ chooses ρ'_r such that $\mathbf{CH}_{V_1}(pk_{V_1}, a, \rho_r) = \mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1 \cdot \mathbf{z}_1 + \mathbf{A}_2 \cdot \mathbf{z}_2 - \mathbf{c} \cdot \mathbf{t}, \rho'_r)$.
- Output the transcript $(\mathbf{w}', (\rho_{r_i}, \mathbf{c}), \mathbf{z}', \sigma'_{\mathbf{w}}, \sigma'_{\mathbf{z}})$.

As mentioned above, although $\mathbf{V}_1$ can pass the verification by simulating many valid transcripts $(\mathbf{w}_i, (\rho'_r, \mathbf{c}_i), \mathbf{z}_i, \sigma_{\mathbf{w}_i}, \sigma_{\mathbf{z}_i})$, $\mathbf{V}_1$ still knows nothing about the knowledge from the transcripts. □

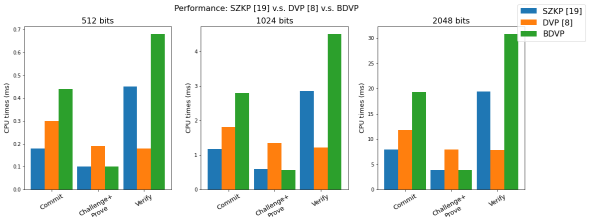

FIGURE 4. The performance comparison of SZKP [19], DVP [8] and BDVP.

Theorem 11: The proposed scheme is non-transferable if Theorem7 holds, and the shortest vector problem is computationally hard.

Proof: Due to *trapdoor collisions*, the third party **T** can believe that $\mathbf{V}_1$ is able to compute $\mathbf{s}'$ by:

$$\rho' = \mathbf{UF}_{V_1}(t_{V_1}, \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2, \rho_{V_1}, \mathbf{A}_1 \cdot \mathbf{z}_1 + \mathbf{A}_2 \cdot \mathbf{z}_2 - \mathbf{c} \cdot \mathbf{t})$$

such that $\mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2, \rho_{V_1}) = \mathbf{CH}_{V_1}(pk_{V_1}, \mathbf{A}_1 \cdot \mathbf{z}_1 + \mathbf{A}_2 \cdot \mathbf{z}_2 - \mathbf{c} \cdot \mathbf{t}, \rho')$, so the signature on commitment will be the same. Even if **P** publishes invalid proofs, $\mathbf{V}_1$ can always pass the verification. This makes the proofs non-transferable. Let $\mathbf{p}' = (\mathbf{CH}_{V_1}, pk_{V_1}, \rho')$, the probability of non-transferable is:

$$Pr[\mathbf{Verify}(\mathbf{p}', \mathbf{w}, \mathbf{c}, \mathbf{z}, \sigma_{\mathbf{w}}, \sigma_{\mathbf{z}}) = 1] \geq 1 - \epsilon(\lambda)$$

□

Based on Theorems 8 and Theorem 11, our scheme is an NIQR-BDVP protocol.

Theorem 12: Assume that everyone in NIQR-BDVP cannot transfer his/her trapdoor key t_{V_i} to others. The NIQR-BDVP protocol is completeness, soundness, zero-knowledge and privacy protection if Theorem 7 holds, and the shortest vector problem is computationally hard.

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our schemes by comparing the computational time with related scheme. Communication costs and blockchain runtime are not considered in our evaluation. The experimental environment is as follows.

1) software
   a) Sagemath 9.6
   b) Python 3.10.3
2) hardware
   a) CPU: 4-core, Intel Core i5
   b) Memory: 8GB

A. BDVP PERFORMANCE

We compared the DLP-based ZKP protocol [19], DLP-based DVP protocol [8], and DLP-based BDVP protocol. For the fairness of the comparison, we additionally implemented a simple DLP-based signature in ZKP protocol (SZKP). The chameleon hash used in our BDVP scheme is Ateniese's identity-based chameleon hash function [43]. We set the security parameters to 512, 1024, and 2048 in our experiments. We run each protocol 20 times and compute average running time. The results are shown in Figure 4.

Note that we combined key generation (**KeyGen**) and commit (**Commit**) algorithms in DVP and compared them with **Commit** defined in SZKP and BDVP. We also combined **Challenge** and **Prove** in SZKP and BDVP and compared them with prover verification (**PV**) in DVP. The reason is that DVP separate the whole process into different phases with SZKP and BDVP. The results agree with our expectations; we have a longer execution time than other two due to the chameleon hash computation. It requires a longer time to compute the commitment and the verification than others, and also needs a longer time to generate key pair. For the challenge phase and the proof generation phase, there are almost no differences. Due to the chameleon hash function, we have the **KeyGen**() algorithm only in non-transferable protocol. Considering the extra privacy features we offer, we think the cost is acceptable.

B. NIQR-BDVP PERFORMANCE

We compared the lattice-based NI-ZKP protocol and lattice-based NI-BDVP protocol (NIQR-ZKP vs NIQR-BDVP). We set the polynomial of degree $N = 1024, \mathfrak{s}_1 = \mathfrak{s}_2 = 27000, q = 2^{32}$, and set n, k_1, k_2 as $(1, 3, 3)$. We run each protocol 20 times and compute the average running times. These parameters can be checked in [29]. We use Wu's quantum-resistant chameleon hash function scheme [39], which is lattice-based. The results are shown in Figure 5.

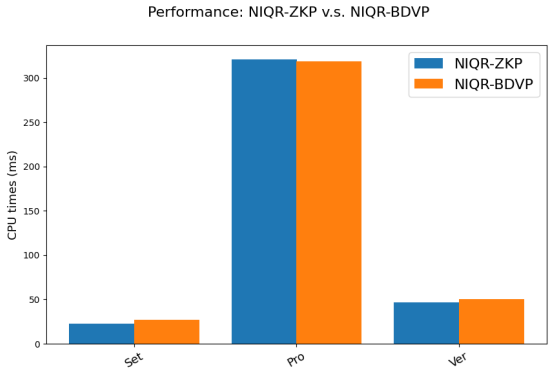

FIGURE 5. The performance comparison of NIQR-ZKP [29] and NIQR-BDVP.

The results show that we also require more time to generate the key pair in the quantum-resistant ZKP scheme. According to our results, there is a slightly longer time in **Setup**() or **Verify**() than NIQR-ZKP due to the chameleon hash computation.

VII. CONCLUSION

Designated verifier proofs are a useful tool for protecting the privacy of the prover, since only a specific verifier can be convinced that the prover possesses certain sensitive information. However, due to the immutable record service provided by the blockchain, the non-transferability of DVP cannot be guaranteed. Even if the verifier is willing to protect the privacy of the prover, a third party can learn from the blockchain record that the provider has certain secrets. In this paper, we design a new DVP scheme, the BDVP scheme, suitable for blockchains applications. The key technique behind our BDVP scheme is that the verifier can forge a fake secret to simulate the proof. Therefore, a third party cannot determine whether the prover possesses the secret. This enables the verifier to protect the privacy of the prover, which is required by law or regulations. We provides rigorous proofs and performance evaluation, and show the scheme is secure and reasonably affordable. We also address the quantum attack problem and propose a post-quantum solution. Even if the attacker has a quantum computer, the prover's privacy can be protected. Therefore, our scheme is suitable for blockchain-based applications.

REFERENCES

[1] B. Soewito, Y. Marcellinus, and M. Hapsara, "Secure wireless ad hoc networks using zero knowledge proof," *J. Comput. Sci.*, vol. 10, no. 12, pp. 2488–2493, Dec. 2014, doi: 10.3844/jcssp.2014.2488.2493.
[2] H. M. Alshameri and P. Kumar, "An efficient zero-knowledge proof based identification scheme for securing software defined network," *Scalable Comput., Pract. Exper.*, vol. 20, no. 1, pp. 181–189, Mar. 2019, doi: 10.12694/scpe.v20i1.1473.
[3] N. Xi, W. Li, L. Jing, and J. Ma, "ZAMA: A ZKP-based anonymous mutual authentication scheme for the IoV," *IEEE Internet Things J.*, vol. 9, no. 22, pp. 22903–22913, Nov. 2022.
[4] X. Sun, F. R. Yu, P. Zhang, Z. Sun, W. Xie, and X. Peng, "A survey on zero-knowledge proof in blockchain," *IEEE Netw.*, vol. 35, no. 4, pp. 198–205, Jul. 2021.
[5] J. Partala, T. H. Nguyen, and S. Pirttikangas, "Non-interactive zero-knowledge for blockchain: A survey," *IEEE Access*, vol. 8, pp. 227945–227961, 2020.
[6] W. Li, H. Guo, M. Nejad, and C.-C. Shen, "Privacy-preserving traffic management: A blockchain and zero-knowledge proof inspired approach," *IEEE Access*, vol. 8, pp. 181733–181743, 2020.
[7] X. Yang and W. Li, "A zero-knowledge-proof-based digital identity management scheme in blockchain," *Comput. Secur.*, vol. 99, Dec. 2020, Art. no. 102050. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404820303230
[8] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," in *Advances in Cryptology (EUROCRYPT)*, U. Maurer, Ed. Berlin, Germany: Springer, 1996, pp. 143–154.
[9] B. Wang, "A non-interactive deniable authentication scheme based on designated verifier proofs," IACR Cryptol. ePrint Arch., Tech. Rep. 2008/159, 2008. [Online]. Available: http://eprint.iacr.org/2008/159
[10] X. Chen, G. Chen, F. Zhang, B. Wei, and Y. Mu, "Identity-based universal designated verifier signature proof system," *Int. J. Netw. Secur.*, vol. 8, no. 1, pp. 52–58, 2009. [Online]. Available: http://ijns.jalaxy.com.tw/contents/ijns-v8-n1/ijns-2009-v8-n1-p52-58.pdf
[11] P. Chaidos and G. Couteau, "Efficient designated-verifier non-interactive zero-knowledge proofs of knowledge," in *Advances in Cryptology (EUROCRYPT)* (Lecture Notes in Computer Science), vol. 10822, J. B. Nielsen and V. Rijmen, Eds. Cham, Switzerland: Springer, 2018, pp. 193–221, doi: 10.1007/978-3-319-78372-7_7.
[12] M. Campanelli and H. Khoshakhlagh, "Succinct publicly-certifiable proofs (or: Can a blockchain verify a designated-verifier proof?)," Cryptol. ePrint Arch., Tech. Rep. 2021/1618, 2021. [Online]. Available: https://eprint.iacr.org/2021/1618
[13] Y. Aumann and M. O. Rabin, "Authentication, enhanced security and error correcting codes (extended abstract)," in *Advances in Cryptology (EUROCRYPT)* (Lecture Notes in Computer Science), H. Krawczyk, Ed., vol. 1462. Cham, Switzerland: Springer, 1998, pp. 299–303, doi: 10.1007/BFb0055736.
[14] L. Fan, C. X. Xu, and J. H. Li, "Deniable authentication protocol based on Deffie-Hellman algorithm," *Electron. Lett.*, vol. 38, no. 14, p. 705, 2002.
[15] W.-B. Lee, C.-C. Wu, and W.-J. Tsaur, "A novel deniable authentication protocol using generalized ElGamal signature scheme," *Inf. Sci.*, vol. 177, no. 6, pp. 1376–1381, Mar. 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025506002994
[16] H. Zhu, "A simplified deniable authentication scheme in cloud-based pay-TV system with privacy protection," *Int. J. Commun. Syst.*, vol. 32, no. 11, Jul. 2019, Art. no. e3967, doi: 10.1002/dac.3967.

[17] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof-systems," in *Proc. 17th Annu. ACM Symp. Theory Comput. (STOC)*. New York, NY, USA: Association for Computing Machinery, 1985, pp. 291–304, doi: 10.1145/22145.22178.
[18] O. Goldreich and E. Kushilevitz, "A perfect zero-knowledge proof for a problem equivalent to discrete logarithm," in *Advances in Cryptology (CRYPTO)* (Lecture Notes in Computer Science), S. Goldwasser, Ed., vol. 403. Cham, Switzerland: Springer, 1988, pp. 57–70, doi: 10.1007/0-387-34799-2_5.
[19] C. P. Schnorr, "Efficient identification and signatures for smart cards," in *Advances in Cryptology (CRYPTO)*, G. Brassard, Ed. New York, NY, USA: Springer, 1990, pp. 239–252.
[20] A. De Santis, S. Micali, and G. Persiano, "Non-interactive zero-knowledge proof systems," in *Advances in Cryptology (CRYPTO)*, C. Pomerance, Ed. Berlin, Germany: Springer, 1988, pp. 52–72.
[21] R. Gennaro, D. Micciancio, and T. Rabin, "An efficient non-interactive statistical zero-knowledge proof system for quasi-safe prime products," Cryptol. ePrint Arch., Tech. Rep. 1998/008, 1998. [Online]. Available: https://eprint.iacr.org/1998/008
[22] G. Persiano and I. Visconti, "On non-interactive zero-knowledge proofs of knowledge in the shared random string model," in *Mathematical Foundations of Computer Science*, R. Královič and P. Urzyczyn, Eds. Berlin, Germany: Springer, 2006, pp. 753–764.
[23] C. Peikert and V. Vaikuntanathan, "Noninteractive statistical zero-knowledge proofs for lattice problems," in *Advances in Cryptology (CRYPTO)*, D. Wagner, Ed. Berlin, Germany: Springer, 2008, pp. 536–553.
[24] F. Martín-Fernández, P. Caballero-Gil, and C. Caballero-Gil, "Authentication based on non-interactive zero-knowledge proofs for the Internet of Things," *Sensors*, vol. 16, no. 1, p. 75, Jan. 2016, doi: 10.3390/s16010075.
[25] K. Xagawa and K. Tanaka, "Zero-knowledge protocols for NTRU: Application to identification and proof of plaintext knowledge," in *Provable Security*, J. Pieprzyk and F. Zhang, Eds. Berlin, Germany: Springer, 2009, pp. 198–213.
[26] I. B. Damgard, T. P. Pedersen, and B. Pfitzmann, "Statistical secrecy and multibit commitments," *IEEE Trans. Inf. Theory*, vol. 44, no. 3, pp. 1143–1151, May 1998.
[27] S. Halevi and S. Micali, "Practical and provably-secure commitment schemes from collision-free hashing," in *Advances in Cryptology (CRYPTO)*, N. Koblitz, Ed. Berlin, Germany: Springer, 1996, pp. 201–215.
[28] D. Cabarcas, D. Demirel, F. Göpfert, J. Lancrenon, and T. Wunderer, "An unconditionally hiding and long-term binding post-quantum commitment scheme," Cryptol. ePrint Arch., Tech. Rep. 2015/628, 2015. [Online]. Available: https://eprint.iacr.org/2015/6282
[29] C. Baum, I. Damgård, V. Lyubashevsky, S. Oechsner, and C. Peikert, "More efficient commitments from structured lattice assumptions," in *Security and Cryptography for Networks* (Lecture Notes in Computer Science), vol. 11035, D. Catalano and R. D. Prisco, Eds. Amalfi, SA, Italy: Springer, Sep. 2018, pp. 368–385.
[30] V. Lyubashevsky, N. K. Nguyen, and M. Plançon, "Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general," IACR Cryptol. ePrint Arch., Tech. Rep. 284/2022, 2022. [Online]. Available: https://eprint.iacr.org/2022/284
[31] X. Xie, R. Xue, and M. Wang, "Zero knowledge proofs from ring-LWE," in *Cryptology and Network Security*, M. Abdalla, C. Nita-Rotaru, and R. Dahab, Eds. Cham, Switzerland: Springer, 2013, pp. 57–73.
[32] R. Steinfeld, H. Wang, and J. Pieprzyk, "Efficient extension of standard schnorr/rsa signatures into universal designated-verifier signatures," IACR Cryptol. ePrint Arch., Tech. Rep. 193/2003, 2003. [Online]. Available: http://eprint.iacr.org/2003/193
[33] H. Qian, Z. Cao, L. Wang, and Q. Xue, "Efficient non-interactive deniable authentication protocols," in *Proc. 5th Int. Conf. Comput. Inf. Technol. (CIT)*, 2005, pp. 142–159, doi: 10.1109/cit.2005.109.
[34] H. Krawczyk and T. Rabin, "Chameleon hashing and signatures," Cryptol. ePrint Arch., Tech. Rep. 1998/010, 1998. [Online]. Available: https://ia.cr/1998/010
[35] G. Ateniese and B. de Medeiros, "On the key exposure problem in chameleon hashes," in *Security in Communication Networks* (Lecture Notes in Computer Science), vol. 3352, C. Blundo and S. Cimato, Eds. Cham, Switzerland: Springer, 2004, pp. 165–179, doi: 10.1007/978-3-540-30598-9_12.
[36] J. Choi and S. Jung, "A handover authentication using credentials based on chameleon hashing," *IEEE Commun. Lett.*, vol. 14, no. 1, pp. 54–56, Jan. 2010.
[37] P. Mohassel, "One-time signatures and chameleon hash functions," in *Selected Areas in Cryptography*, A. Biryukov, G. Gong, and D. R. Stinson, Eds. Berlin, Germany: Springer, 2011, pp. 302–319.
[38] S. Guo, D. Zeng, and Y. Xiang, "Chameleon hashing for secure and privacy-preserving vehicular communications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 11, pp. 2794–2803, Nov. 2014.
[39] C. Wu, L. Ke, and Y. Du, "Quantum resistant key-exposure free chameleon hash and applications in redactable blockchain," *Inf. Sci.*, vol. 548, pp. 438–449, Feb. 2021, doi: 10.1016/j.ins.2020.10.008.
[40] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Rev.*, vol. 41, no. 2, pp. 303–332, Jan. 1999, doi: 10.1137/S0036144598347011.
[41] T. Attema and R. Cramer, "Compressed Σ-protocol theory and practical application to plug & play secure algorithmics," in *Advances in Cryptology (CRYPTO)* (Lecture Notes in Computer Science), vol. 12172, D. Micciancio and T. Ristenpart, Eds. Cham, Switzerland: Springer, 2020, pp. 513–543, doi: 10.1007/978-3-030-56877-1_18.
[42] T. Attema, R. Cramer, and L. Kohl, "A compressed Σ-protocol theory for lattices," in *Advances in Cryptology (CRYPTO)* (Lecture Notes in Computer Science), vol. 12826, T. Malkin and C. Peikert, Eds. Cham, Switzerland: Springer, 2021, pp. 549–579, doi: 10.1007/978-3-030-84245-1_19.
[43] G. Ateniese and B. de Medeiros, "Identity-based chameleon hash and applications," IACR Cryptol. ePrint Arch. Tech. Rep. 2003/167, 2003. [Online]. Available: http://eprint.iacr.org/2003/167

PO-WEN CHI received the B.S., M.S., and Ph.D. degrees in electrical engineering from National Taiwan University, in 2003, 2005, and 2016, respectively. From 2005 to 2016, he was an Engineer with the Institute for Information Industry, Taiwan. From 2016 to 2018, he was an Senior Engineer with Arcadyan Technology Corporation, Taiwan. He joined the Department of Computer Science and Information Engineering, National Taiwan Normal University, as an Assistant Professor, in 2018, where he is currently an Associate Professor. His research interests include network security, applied cryptography, software-defined networking, and telecommunications.

YUN-HSIU LU received the B.S. degree in physics from Nation Central University, in 2020. He studied for master of CSIE, in 2020, presented in 2022 CISC oral session, and passed his master oral defense, in 2022. His research interest includes applied cryptography.

ALBERT GUAN received the bachelor's degree in applied mathematics from National Sun Yat-sen University, in 2008, and the Ph.D. degree in computer science from National Chiao Tung University, in 2017. From 2017 to 2018, he was a Postdoctoral Research Fellow with the Department of Electrical Engineering, National Taiwan University. In 2018, he joined the Department of Applied Mathematics, National Sun Yat-Sen University, as an Assistant Professor. In 2022, he joined the Department of Computer Science and Information Engineering, National Taiwan Normal University, as an Assistant Professor. His research interests include discrete mathematics, cryptography, and its applications.