Journal of Parallel and Distributed Computing 180 (2023) 104721

Contents lists available at ScienceDirect

Journal of Parallel and Distributed Computing

journal homepage: www.elsevier.com/locate/jpdc

PRI: PCH-based privacy-preserving with reusability and interoperability for enhancing blockchain scalability

Yuxian Li [a], Jian Weng [a,*], Wei Wu [a], Ming Li [a], Yingjiu Li [b], Haoxin Tu [c], Yongdong Wu [a], Robert H. Deng [c]

[a] College of Cyber Security, Jinan University, Guangzhou 510632, PR China
[b] School of Computer and Information Science Department, University of Oregon, United States of America
[c] School of Computing and Information Systems, Singapore Management University, Singapore

ARTICLE INFO

Article history:
Received 21 November 2022
Received in revised form 16 February 2023
Accepted 26 May 2023
Available online 1 June 2023

Keywords:
Blockchain
Reusability
Scalability
Interoperability

ABSTRACT

Blockchain systems, one of the most popular distributed systems, are well-applied in various scenarios, e.g., logistics and finance. However, traditional blockchain systems suffer from scalability issues. To tackle this issue, Payment Channel Hubs (PCHs) are proposed. Recent efforts, such as A2L (SP'21) and Teechain (SOSP'19), enhance the privacy, reusability, and interoperability properties of PCHs. Nevertheless, these solutions have intrinsic limitations: they rely on trusted hardware or suffer from the deposit lock-in problem. Furthermore, the functionalities of some of these solutions are restricted to fixed-amount payments and do not support multi-party participation. These aforementioned problems limit their capabilities to alleviate blockchain scalability issues. In this paper, we propose PRI, a novel PCH solution that simultaneously guarantees transaction **P**rivacy (i.e., relationship unlinkability and value confidentiality), deposit **R**eusability, and blockchain **I**nteroperability, which can mitigate the aforementioned problems. PRI is constructed by several new building blocks, including (1) an atomic deposit protocol that enforces user and hub to deposit equivalent assets in a shared address for building a fair payment channel; (2) a privacy-preserving deposit certification scheme that leverages the Pointcheval and Sanders signature and non-interactive zero-knowledge proof to resolve the deposit lock-in issue in maintaining payment channels; (3) a range proof which ensures the legality and confidentiality of transaction values. We conduct extensive experimental evaluations of PRI, demonstrating that it improves the state-of-the-art approaches in terms of performance.

1. Introduction

Cryptocurrencies, such as Bitcoin [1], Ethereum [2], and ZeroCash [3], have been widely adopted in Decentralized Finance (DeFi). Unfortunately, most cryptocurrencies suffer from the scalability issue causing a performance bottleneck [4,5]. Bitcoin throughput, for example, is limited to about ten transactions per second [6]. In contrast, centralized payment systems such as Visa can handle thousands of transactions per second.

An effective way of addressing the scalability issue is to process massive transactions *off-chain* instead of directly submitting them to immutable blockchains [4,7]. A leading off-chain payment solution is to process transactions off-chain via *Payment Channels* (PCs) [8,9]. Concretely, users first deposit certain assets (known as *channel capacity*) in a shared account to create a channel. Second, the users can negotiate the distribution of the deposits off-chain to pay for each other. Finally, to close the channel, one of the users in the channel makes a transaction based on the latest distribution. A primary advantage of PCs is that they avoid modifying underlying blockchains' structures and consensus mechanisms while reducing on-chain transactions significantly.

However, PCs still have some practical limitations, including the issue *deposit lock-in*, meaning that only a tiny fraction of deposits on payment channels can be spent before they are unlocked. This limitation is caused by the fact that: 1) users should lock sufficient assets while establishing payment channels and 2) their deposits cannot be divided and reused into different channels before channel closure [10]. Such limitations will be exacerbated especially when users establish multiple channels since each channel requires users to deposit certain assets.

* Corresponding author.
E-mail addresses: yuxianlijnu@gmail.com (Y. Li), cryptjweng@gmail.com (J. Weng), wuwei2019jnu@gmail.com (W. Wu), limjnu@gmail.com (M. Li), yingjiul@uoregon.edu (Y. Li), haoxintu.2020@phdcs.smu.edu.sg (H. Tu), wuyd007@qq.com (Y. Wu), robertdeng@smu.edu.sg (R.H. Deng).

https://doi.org/10.1016/j.jpdc.2023.104721
0743-7315/

To resolve this practical limitation caused by the deposit lock-in issue, some works propose a Payment Channel Network (PCN) [11,12,10,13] and Payment Channel Hub (PCH) [14–18] to reduce the number of channels. Compared with PCN, PCH allows users to perform transactions off-chain through a hub, thereby resisting the wormhole attack [5] and avoiding payment routing problem [19,20] in PCN.

Since the participation of the hub, PCH raises a significant privacy concern. That is, when hubs participate in the off-chain payments, honest but curious hubs or any attackers who compromise the hubs may collect private information, including the relationship between senders and receivers as well as the payment amount. To resolve the leakage of the payment relationship, a state-of-the-art PCH scheme, named A2L [14], achieves relationship unlinkability with the restriction of requiring all payments made at a fixed amount. Further, to protect the payment amount and achieve value confidentiality, some PCH schemes [18,15] take advantage of the virtual channel concept. However, since these schemes [18,15] require users to register their identifications to blockchains at the beginning, the relationships between senders and receivers are leaked. Additionally, some schemes address the deposit reuse issue via trusted hardware [21] but these schemes require each user equipped with trusted hardware. In short, the existing solutions suffer from some limitations such as requiring a fixed amount or trusted hardware so they can not satisfy some flexible payment situations. *Thus, it remains challenging to achieve relationship unlinkability, value confidentiality, and deposit reusability in PCH schemes simultaneously, which calls for a new solution to alleviate it.*

Solution. To address the aforementioned challenge, we propose PRI, a novel PCH-based solution to achieve relationship unlinkability, value confidentiality, and deposit reusability. Also, PRI supports interoperability to allow users who process wallets in different blockchains to establish payments and balance security to prevent honest parties from economic losses.

First, to realize confidentiality and balance security, PRI utilizes the Non-Interactive Zero-Knowledge proof (NIZK) scheme [3] and the Pedersen commitment scheme [22] to realize offline payments. In particular, the NIZK scheme and commitment scheme can hide the payment amount while ensuring the correctness of each payment. Further, an atomic deposit protocol is proposed to ensure the users and hub lock equivalent assets while building channels, which avoids the honest parties (hub or user) locking assets permanently. Second, to support reusability and interoperability, PRI unifies the payment channel in different blockchains via a privacy-preserving credential mechanism. The mechanism is constructed over the Pointcheval and Sanders signature scheme [23], the NIZK scheme [3], as well as the Pedersen commitment scheme [22]. Under the unification of the payment channel, users can divide their deposits into several parts. Further, hubs can regard the offline payments of the channels in different blockchains as the offline payment in the same blockchains. Finally, due to the privacy-preserving property of the credential mechanism and the unification of the channels, the hub can not associate the senders and receivers of the same offline payments. Further, PRI can guarantee relationship unlinkability.

1.1. Contributions

In this paper, we make the following contributions:

- *Atomic deposit protocol.* We propose an *atomic deposit* protocol with dual scripts to lock equivalent assets in an account controlled by users and hubs without an expiration time. Thus, users and hubs can be effectively prevented from losing their deposits even though the other involved parties behave dishonestly.
- *Flexible and privacy-preserving PCH construction.* We introduce a flexible and privacy-preserving PCH-based solution, namely PRI, based on the *atomic deposit* protocol and other cryptographic primitives [23,22]. First, to unify payment channels, PRI utilizes the randomizable signature scheme and Pedersen commitment scheme to construct a privacy-preserving credential mechanism for payment hubs. Second, PRI introduces the NIZK scheme for stating the correctness of offline payments in virtual channels without revealing any payment details. The unification of payment channels and zero-knowledge proofs makes it possible for users to divide their deposits and reuse them in creating virtual channels. Additionally, PRI supports multi-party payments and allows users to post offline payments in the case that payment hubs are temporarily unavailable.
- *Feasibility and performance evaluation.* The evaluation measures the computation cost, communication cost, and latency of PRI in two-party and multi-party payments. Further, compared with BOLT [24] and A2L [14], PRI outperforms them in terms of performance.

1.2. Organization

The remainder of the paper is organized as follows. Section 2 summarizes the related work. Section 3 introduces cryptographic primitives. Sections 4 and 5 present the PRI model and protocol, respectively. Section 6 analyzes the security of PRI. Section 7 details our implementation of PRI and provides experimental results. Finally, a conclusion is drawn in Section 8.

2. Related work

This section discusses layer-two solutions as a means of addressing the blockchain scalability issue. These solutions are designed to improve privacy, support diverse functionalities, and facilitate flexible applications. Specifically, we provide a summary of state-of-the-art layer-two schemes (cf. Table 1) and highlight that existing solutions can not support privacy, functionality, and flexible application simultaneously.

Privacy. To improve privacy, most existing works [15,18,16,14] pay attention to privacy concerns so that they can support relationship *unlinkability* and value *confidentiality*. S. Dziembowski, et al. [15,18] and L. Aumayr, et al. [16,26] introduce virtual channels over payment channels against the protection of value confidentiality. However, since the relationships between the transacting parties are disclosed to the intermediary, unlinkability is not supported in these protocols [15,18,16,26]. Several alternative protocols [17,14,10,25,27] incorporate adaptor signatures and zero-knowledge proofs [28] or employ Trusted Execution Environment (TEE) in order to facilitate the mixing of offline payments and enable the realization of privacy-preserving layer-two solutions. In particular, Teechain [10] achieves both properties and proposes a PCN solution that preserves privacy using trusted hardware (e.g., Intel SGX) [29,21,30]. This solution requires each user equipped with trusted hardware, and thus the user without trusted hardware can not adopt this solution directly.

In contrast to these previous works, PRI inherits the benefits of virtual channels to achieve value confidentiality. Further, PRI employs efficient cryptographic primitives, instead of relying on trusted hardware, to conceal the relationship of transacting parties.

Functionality. To achieve functionality, some schemes [24,25, 10,14,27] support 1) blockchain *interoperability*, which enables

Table 1
The comparison of the state-of-the-art layer-two schemes.

Scheme		Functionality	Privacy	multi-party Payment	Flexible Payment
PCN	BOLT [24]	◐	●	○	●
	AMCU [25]	●	◐	○	●
	BCVC [26]	○	◐	○	●
	Blitz [16]	○	◐	○	●
	Teechain [10]	◐	◐	●	●
	Thora [27]	◐	◐	●	●
	Perun [15]	○	◐	○	●
PCH	MVSC [18]	○	◐	●	●
	Tumblebit [17]	○	●	○	○
	A2L [14]	◐	●	○	○
	PRI	●	●	●	●

○ refers to the scheme that does not realize or consider the property.
◐ refers to the scheme that needs to be improved to realize the property.
● refers to the scheme that realizes the property.

users to make payments across different blockchains and 2) deposit *reusability*, which allows users to reuse one channel deposit to establish multiple virtual channels simultaneously. BOLT [24] is proposed against specific blockchains, it cannot be directly applied to other blockchains and further cannot support interoperability. Teechain [10] and AMCU [25] make unification of payment channels in different blockchains so that users can use a unified interface to exchange coins in different blockchains and further reuse their deposits. Nevertheless, Teechain [10] requires users equipped with trusted hardware, which will bring extra limitations to the users. Further, A2L [14] utilizes an adaptor signature scheme to unify payment channels but the deposits in each channel can not be used to establish multiple virtual channels.

Compared to other approaches, PRI employs a signature scheme and commitment scheme to establish a credential mechanism for unifying channels across different blockchains, thereby achieving the property.

Application. To satisfy different scenes, the existing protocols are required to support multi-party payment, which allows multiple users to pay for each other by a single payment, and flexible payment, which enables users to pay any amount rather than at a fixed amount. Flexible payment is achieved in most of the schemes [24,11], but multi-party payment is achieved in a few schemes [10,18]. Furthermore, since these schemes [10,18] require a committee for each user or rely on the smart contract mechanism, they are limited to specific scenes, e.g., blockchains supporting smart contracts.

PRI proposes a layer-two solution that differs from these solutions by leveraging virtual channels and zero-knowledge proofs, enabling multiple parties to make flexible payments. Additionally, it is compatible with most scriptless blockchains, such as Bitcoin.

3. Preliminaries

In this section, we review the used cryptographic primitives: the Pointcheval and Sanders signature scheme, commitment scheme, and non-interactive zero-knowledge proof. We define 1^λ as the security parameter, where $\lambda \in \mathbb{N}$. Further, we use $x \to A(y)$ to represent a probabilistic polynomial time (PPT) algorithm A that inputs y and outputs x.

Pointcheval and Sanders (PS) signature scheme. PRI requires a signature scheme Π_{Sig} that supports users to sign on a message included in a commitment. Thus, we introduce the PS signature scheme [23] with three algorithms (SigKeyGen, Sign, Verify). Specifically, $(sk, pk) \leftarrow \text{SigKeyGen}(1^\lambda)$ is the key generation algorithm to generate a key pair (sk, pk). $\sigma \leftarrow \text{Sign}(sk, com)$ is the signature generation algorithm to sign on a commitment com of a message m. $\{0,1\} \leftarrow \text{Verify}(\sigma, com, pk)$ is the verification algorithm, where 1 indicates the success of the verification and 0 indicates the failure of the verification. The PS signature scheme includes the following features: (1) preserving the signed message(s) from the signer and (2) satisfying the security property against *Existential Unforgeability under Chosen Message Attack* (EUF-CMA) [31]. The scheme enables users in PRI to generate a certification for unifying the legality of payment channels while protecting the private information of the channels.

Commitment scheme. Our construction requires an efficient commitment scheme Π_{cm} that enables users to commit a value in a commitment and open the commitment later. A commitment scheme consists of two algorithms (Com, Decom). In detail, Com is the commitment algorithm to generate the commitment com of a message m, such that $com \leftarrow \text{Com}(m;r)$. Concretely, com is computed by the following formula: $g^r h^m$, where $r \in \mathbb{Z}_q$ is a random number and g, h are elements of the group $\mathbb{G}$. Decom is the decommitment algorithm for verifying whether the commitment com commits the message m, such that $\{0,1\} \leftarrow \text{Decom}(com, r, m)$ where outputs 1 for representing the succeed of the verification. In our case, PRI initializes the Pedersen commitment scheme [22] because this commitment scheme satisfies the information-theoretically hiding property and computationally binding property. With these properties, the private information of payment channels can be hidden and committed in commitments which can be signed by the PS signature scheme.

Non-interactive zero-knowledge proof (NIZK). Our solution utilizes a NIZK scheme Π_{NIZK} which allows provers to prove a statement about the witness for verifiers without disclosing any information about the witness. For example, the NIZK scheme is able to state that a committed value (x) lies in a certain range ($0,\ n$), such as NIZK=$\{(x, r)$: $com=g^x h^r \ \wedge\ 0 < x < n\}$. Specifically, NIZK is composed of two algorithms (Prove, Ver). Prove is the proof generation algorithm executed by the prover. The algorithm can be denoted as $\pi \leftarrow \text{Prove}(x, stmt)$, where the witness x satisfies the statement $stmt$. Ver is the verification algorithm executed by the verifier such that $\{0,1\} \leftarrow \text{Ver}(\pi, stmt)$. In PRI, the NIZK scheme can be initialized by various proof systems [32,33,3] that satisfy zero-knowledge, completeness, and soundness properties. Under these properties, PRI can prove the correctness of payment detail without revealing them to the hub, thus achieving unlinkability and confidentiality.

4. PRI models

In this section, we first present the system model of PRI. Then, we give a formal definition of the Privacy-preserving Payment Channel Hub (PPCH). Finally, we define the threat model and de-

Table 2
The notations.

Notation	Description
$\mathcal{P}_i$	The participant i.
$H(\cdot)$	The hash function.
λ	The security parameter.
acc_i	The account is jointly managed by a hub and the user i.
$type_1$	The type of assets is Bitcoin.
$type_2$	The type of assets is ETH.
δ_i	The partial multi-signature generated by the user i.
max	The maximum amount allowed in underlying blockchains.
(pk, sk)	The key pair of the hub.
ρ	The random number used in the Com algorithm.
T_i	The time lock in a transaction, where i denotes the index.
v_i^j	The amount of assets owned by the user j, where i denotes the index.
wpk_i^j	The wallet identification is owned by the user j, where i is the index.
wsk_i^j	The secret key is corresponding to the identification wpk_i^j.
$\tilde{\sigma}_i^j$	The signature produced by the user j by wsk_i^j, where i denotes the index.
σ_i^j	The certification of a wallet owned by the user j, where i denotes the index of the wallet.

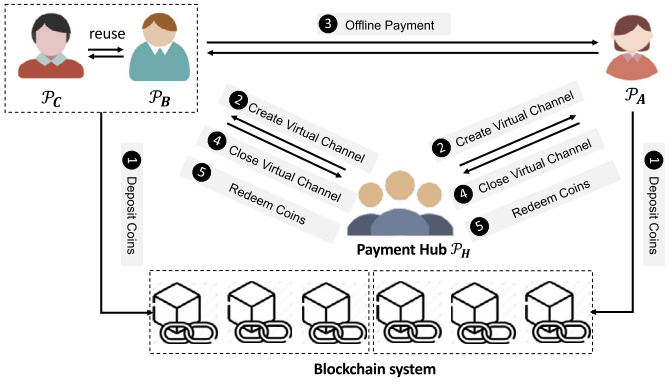


Fig. 1. The Overview of PRI.

sign goals of PRI. Also, to clarify our description, Table 2 shows the notations used in this paper.

4.1. System model

Fig. 1 shows the PRI system model. First, we list the roles that can be taken into account in our system.

- **Users** create channels with a hub and use the channels to transfer assets to other users. Fig. 1 shows an example that a user $\mathcal{P}_A$ intends to exchange assets with the user $\mathcal{P}_B$ multiple times.
- **Hubs** play a role as intermediaries to help users make offline payments. For example, Fig. 1 shows that a hub $\mathcal{P}_H$ takes responsibility for assisting users to pay for each other off-chain.

Second, we follow the existing design [15] and divide PRI into five phases: deposit coins, create virtual channels, offline payment, close virtual channels, and redeem coins. PRI starts with that $\mathcal{P}_A$ (resp. $\mathcal{P}_B$) establishes a payment channel with $\mathcal{P}_H$ via locking assets (e.g., Bitcoin, ETH) to an account controlled by them. $\mathcal{P}_H$ responds with a signature to authenticate the success of the deposit (Step ❶). $\mathcal{P}_A$ and $\mathcal{P}_B$ can utilize their partial deposits rather than their whole deposits to create a new virtual channel (Step ❷). Note that the remnant deposits can be reused to establish other virtual channels simultaneously. Then, $\mathcal{P}_A$ and $\mathcal{P}_B$ can execute offline payments via adjusting the individual balances in the virtual channel without $\mathcal{P}_H$ (Step ❸). If $\mathcal{P}_A$ and $\mathcal{P}_B$ do not pay for each other anymore, they can collaboratively upload the final balance to $\mathcal{P}_H$ to close the virtual channel (Step ❹). After that, each user (e.g., $\mathcal{P}_B$) can still reuse his remnant assets to create a new virtual channel with others (e.g., $\mathcal{P}_C$). At last, the users can withdraw their assets from $\mathcal{P}_H$ and other users (Step ❺).

4.2. Privacy-preserving Payment Channel Hub (PPCH)

Following the definition in [14], we formalize a PPCH between users and a hub. To indicate a protocol F executed between A and B, we write $F(A(a), B(b)) \rightarrow (c, d)$, where a and c are the input and output of A, as well as b and d are the input and output of B. Furthermore, a wallet in the payment channel hub scheme is defined as an attribute tuple $w = (v, pk, sk, \sigma)$, where the wallet identification and the corresponding secret key are pk and sk, as well as the attributes v and σ are the wallet balance and certification. Next, we give a formal definition of a PPCH as follows.

Definition 4.1. (PPCH scheme) A Privacy-preserving Payment Channel Hub (PPCH) scheme is equipped with six interaction protocols (Deposit, DivideDeposit, Establish, Payment, CloseChannel, Redeem) described below:

- Deposit$(\mathcal{P}_A(v_1^A, pp, acc), \mathcal{P}_{\mathcal{H}}(v_1^H, sk_H, acc))$. On inputting the initial balances and a shared account acc, if the user generates a key pair (wpk_1^A, wsk_1^A) as a wallet identification and corresponding secret key, the user initializes a wallet $w = (v_1^A, wpk_1^A, wsk_1^A, \sigma_1^A)$. At the same time, the hub generates the deposit certification σ_1^A via his secret key sk_H to authenticate the deposit.
- DivideDeposit$(\mathcal{P}_A(pp, w, v_2, v_3), \mathcal{P}_H(sk_H, pp, \mathbb{W}))$. Upon inputting the public parameter pp, the old wallet w, and the divided values (v_2, v_3), the user generates two wallets (w_2, w_3) with balances (v_2, v_3) and produces two key pairs to identify the new wallets (w_2, w_3). Further, the user produces a proof stating the validity of the old wallet and the new wallet. Next, the user submits the old wallet identification and the proof to the hub. The hub generates the new deposit certifications for the new wallets and stores the wallet identification of the old wallet w in the array $\mathbb{W}$. If the interaction succeeds, the user will receive two wallets (w_2, w_3).
- Establish$(\mathcal{P}_A(w_2^A, pp), \mathcal{P}_B(w_2^B, pp), \mathcal{P}_H(\mathbb{W}))$. For activating a new virtual channel, $\mathcal{P}_A$ and $\mathcal{P}_B$ respectively submit their unspent wallet identifications and corresponding proofs to the hub $\mathcal{P}_H$. Specifically, the proofs state the validity of their wallet identifications. If the interaction succeeds, the hub will store the wallet identifications in $\mathbb{W}$.
- Payment$(\mathcal{P}_A(w_2{}^A, pp, bal_A, bal_B), \mathcal{P}_B(w_2^B, pp, bal_A, bal_B))$. With the public parameters, wallets, and new balances, each

user creates two wallets under the new balances and generates signatures $(\widetilde{\sigma}^A, \widetilde{\sigma}^B)$ to show their agreement on this payment.
- CloseChannel$(\mathcal{P}_A(\widetilde{\sigma}^A, w_4^A, w_5^A, pp), \mathcal{P}_B(\widetilde{\sigma}^B, w_4^B, w_5^B, pp), \mathcal{P}_H(sk_H, pp))$. When inputting the latest wallets, signatures, and signing keys, the users receive the corresponding new certifications from the hub.
- Redeem$(\mathcal{P}_A(w, addr, sk_A), \mathcal{P}_H(sk_H, acc), \{\mathcal{P}_i(sk_i, acc_i)\})$. Inputting the redeeming wallet and the secret keys, the user's address *addr* receives tokens from the shared accounts. If the remaining assets in the shared account can not afford the redemption, other users can participate in the redemption via their shared accounts *acc* and $\{acc_i\}$. Note that the participants will be paid some fee as the reward in reality while the hub redeems tokens locked in their shared accounts.

4.3. Threat model

We consider the hub and users to be rational and selfish so that they behave following their best financial interests. Thus, we define the potential dishonest behaviors of dishonest users and a hub as follows:

- **Dishonest user(s).** A dishonest user may deceive that he/she has enough money to build a channel with other users.
- **Dishonest hub.** A dishonest hub may refuse to send assets back to a user. Additionally, the hub may be curious about payment details, such as identification, amount, and final balances of a virtual channel.

Remark. A dishonest hub may corrupt dishonest users to steal assets from deposits so that the dishonest hub does not have enough assets to settle the requests from honest users. In light of the problems above, several solvency protocols, such as Provisions [34] and Solidus [35], have been proposed to ensure the hub controls sufficient assets. We did not consider designing solvency protocols here as this line of work is out of the scope of this paper.

4.4. Design goals

Next, we define the following design goals including *unlinkability*, *confidentiality*, *balance security*, and *functionality*.

Unlinkability. It indicates that the relationship between payer-payee pairs in virtual channels is not revealed to the hub $\mathcal{P}_H$. We inherit the definition of unlinkability in A2L [14]. That is, the hub interacts with (1) several oracles that implement real-world protocols for users $\mathcal{P}_A$ and $\mathcal{P}_B$ or (2) a simulator $\mathcal{S}$ that performs the functionality of users. The simulator is required to simulate users without the users' wallets and the payment relationship between users. Additionally, we assume that the simulator is accessible to side information, e.g., control of random oracles or simulation trapdoors. If no adversary can distinguish whether he/she is in the world (1) or (2), the definition of unlinkability will hold. This definition implies unlinkability since the simulator has no information about the wallet simulated by the simulator, resulting in hiding the payment relationship between any pairs of users in a virtual channel.

Confidentiality. It denotes that $\mathcal{P}_H$ cannot obtain the private information of individual payments and the initial and final balances of users. The hub communicates with (1) a series of oracles representing users $\mathcal{P}_A$ and $\mathcal{P}_B$ in the real world or (2) a simulator $\mathcal{S}$ acting as users with access to random oracles and simulation trapdoors. The simulator in the world (2) simulates users without knowing the payment detail in a channel or the users' wallets. Therefore, confidentiality will hold if two worlds (1) and (2) are identical.

Balance security. This property guarantees that no adversarial counterpart can redeem more coins than they owned. Following the definition in BOLT [24], balance security consists of two cases: (1) $\mathcal{P}_H$ does not lose its assets even though all users in a virtual channel are corrupted and (2) a user does not lose his/her assets in a virtual channel even though the other user in the virtual channel and the hub behave dishonestly. Therefore, balance security is related to two games: one for the user and another for the hub. In both cases, we assume the execution of the Deposit and Redeem protocols is honest. Furthermore, the adversarial user (resp. hub) can call oracles that take on a role as the user (resp. hub). The parties involved can create several virtual channels. The adversarial party can initiate the CloseChannel protocol with the counterparty to obtain channel closure messages. The adversary wins these games, while the coins redeemed in the Redeem protocol are inconsistent with the total of coins it owns. If the adversary cannot win these games, the balance security will hold.

Functionality. We consider *interoperability* and *reusability* as necessary functionality goals for PRI. Specifically, interoperability ensures that a pair of users can proceed with offline payments even though they possess assets in different blockchains. Reusability supports that a user can divide his/her deposit locked in the shared account controlled by himself/herself and the hub into several parts for establishing multiple virtual channels.

5. The PRI protocol

In this section, we first give a brief overview of PRI. Next, we introduce a proposed protocol named "atomic deposit" to enforce two participants to deposit coins. Finally, based on the proposed protocol, we give a formal protocol specification to detail PRI.

5.1. Solution overview

PRI achieves a PPCH scheme that overcomes the drawbacks of the current PCH schemes [15,14,10] and yet inherits the privacy and interoperability.

The core of PRI is as follows. PRI enables a hub and users to lock assets via a deposit transaction so that these assets can be utilized to build a capital pool. To guarantee the user and hub lock equivalent assets simultaneously, PRI designs an *atomic deposit* protocol. After making a deposit, users can divide their locked assets into several parts to establish multiple virtual channels to make offline payments. When users do not make offline payments anymore, they can withdraw their assets from the capital pool. Since all assets are controlled by the hub and users, they need to cooperate to respond to the redemption request. To encourage them to actively respond to the service, a certain fee can be paid to the users involved.

To perform the aforementioned operations, we begin with an *atomic deposit* protocol to force a user and hub to deposit equivalent assets into a jointly managed account. Then, via the cryptographic primitives mentioned in Section 3, PRI unifies the channels in different underlying blockchains through the deposit certification mechanism. Through the unification of payment channels, users can accurately divide their deposits into several parts. Here, users can add only partial deposits rather than entire deposits to a new virtual channel, thus utilizing the remnant assets to build other virtual channels with others simultaneously. Meanwhile, the initial and final balances of the users in the virtual channels can be preserved from the hub. Lastly, a user can send a request with a proof stating the validity of the redemption request to the hub to redeem his/her assets.

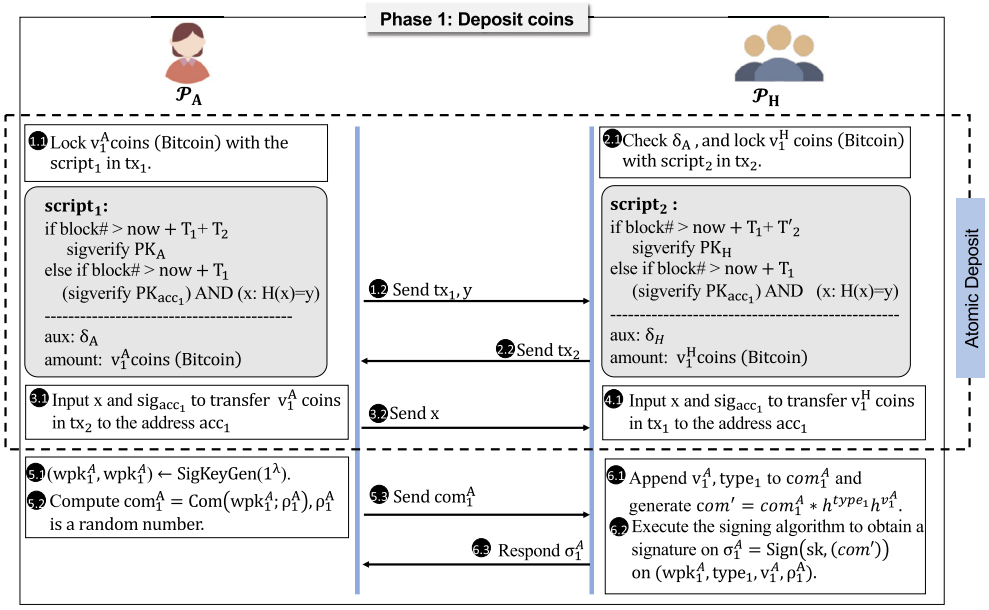


Fig. 2. The process of depositing coins.

5.2. Atomic deposit

Here, we introduce *atomic deposit* protocol (cf. Fig. 2). In PRI, the hub and user lock equivalent assets on a jointly controlled account to create a channel over a blockchain but do not make a timelock to their deposit. That is, once one of the parties transfers his/her assets to the jointly controlled account, the deposit only can be redeemed via the cooperation of both parties. Since this restriction, the atomic deposit protocol should provide a mechanism to prevent the (dishonest) hub from refusing to provide services for users. Under this requirement, we design the atomic deposit protocol which enforces the hub and user deposit equivalent assets into an account acc_1 jointly controlled by them.

As shown in Fig. 2, the protocol starts with the user $\mathcal{P}_A$ constructing a deposit transaction tx_1 with the value v_1^A. This transaction is controlled by a specific script $script_1$. Intuitively, the main role of the script $script_1$ is as follows: (1) enabling $\mathcal{P}_A$ to redeem her assets when $\mathcal{P}_A$ locks certain assets in the transaction tx_1 but $\mathcal{P}_H$ does not respond to these actions before time $\mathtt{T}_1 + \mathtt{T}_2$; (2) enabling $\mathcal{P}_H$ to transfer assets to the account acc if $\mathcal{P}_H$ provides a multi-signature generated by $\mathcal{P}_A$ and $\mathcal{P}_H$ and a value x meeting the formula: H(x)=y where x is selected by $\mathcal{P}_A$. Additionally, $\mathcal{P}_A$ utilizes an extra field aux (e.g., OP_RETURN in Bitcoin) to record the partial multi-signature δ_A which can be used to spend v_1^A in the transaction tx_1. After constructing the transaction tx_1, $\mathcal{P}_A$ sends the transaction tx_1 and the hash value y to the hub $\mathcal{P}_H$ and waits for the response from $\mathcal{P}_H$ (steps 1.1-1.2 in Fig. 2).

Next, the hub $\mathcal{P}_H$ constructs and submits a transaction tx_2 controlled by a similar script $script_2$ to the blockchain, which also defines a due time $\mathtt{T}_1 + \mathtt{T}'_2$ and a hash lock of the hash value y (steps 2.1-2.2 in Fig. 2). At that time, the transactions tx_1 and tx_2 are submitted to the blockchain. Then, $\mathcal{P}_A$ sends the value x to $\mathcal{P}_H$ for transferring assets to the jointly controlled account. Two ways are considered to transfer the value in tx_1 and tx_2 (steps 3.1-4.2 in Fig. 2). The first is that $\mathcal{P}_A$ and $\mathcal{P}_H$ input the value x and the multi-signature to spend the transactions tx_1 and tx_2 respectively. The second is that each party can utilize the value x and the partial multi-signature appended in the extra field aux to spend the transaction of the other party. This way aims to ensure all parties lock their assets when one of two parties has transferred his/her assets but the other one refuses to transfer his/her transaction to the controlled address acc. Note that since the time $\mathtt{T}_2$ in $script_1$ is longer than the time $\mathtt{T}'_2$ in $script_2$, the hub can transfer the assets of $\mathcal{P}_A$ to the account acc before $\mathcal{P}_A$ redeem his/her assets. After that, both $\mathcal{P}_H$ and $\mathcal{P}_A$ deposit equivalent assets to the account acc_1.

Remark. The atomic deposit protocol is a general proposed protocol that can be utilized in other protocols when they require fair deposit among mutually distrustful parties. Further, the hash lock in the script can be substituted by the adaptor signature [14], which is out of the scope of our work. Thus, we omit it here.

5.3. Formal protocol specification

Here, we present the formal protocol specification of PRI. The construction includes the instructions for the Bitcoin user $\mathcal{P}_A$, Ethereum user $\mathcal{P}_B$, and hub $\mathcal{P}_H$.[1] We consider the standard scenario that $\mathcal{P}_A$ and $\mathcal{P}_B$ exchange bitcoins with ethereums with the help of $\mathcal{P}_H$. Fig. 1 shows the overview of the PRI protocol, including five phases: *deposit coins, create virtual channels, establish offline payment, close virtual channels, and redeem coins*. Users create payment channels with hubs in the depositing coins phase. Based on the payment channels, the users divide their deposits and utilize their divided deposits to establish virtual channels. Next, the users can pay for each other offline over the virtual channel. After multi offline payments, the users can close the virtual channel and finally redeem their remained deposits. In the following, we will give a detail of these phases.

5.3.1. Deposit coins

Fig. 2 shows the specific steps of depositing coins, which depend on the atomic deposit protocol and the PS signature scheme [23]. The phase starts with $\mathcal{P}_A$ and $\mathcal{P}_B$ executing the atomic deposit protocol with $\mathcal{P}_H$ to deposit assets on shared accounts. The deposit operations can be regarded as that $\mathcal{P}_A$ and $\mathcal{P}_B$ create wallets with the values v_1^A and v_1^B (steps 1.1-4.1 in Fig. 2). Next, since the operation of $\mathcal{P}_A$ is consistent with the operation of $\mathcal{P}_H$, we take the user $\mathcal{P}_A$ as an example. $\mathcal{P}_A$ generates a key pair (wpk_1^A, wsk_1^A) to identify the wallet with v_1^A coins. $\mathcal{P}_A$ commits the wallet

[1] Note that our protocol can be expanded to other blockchains with little change.

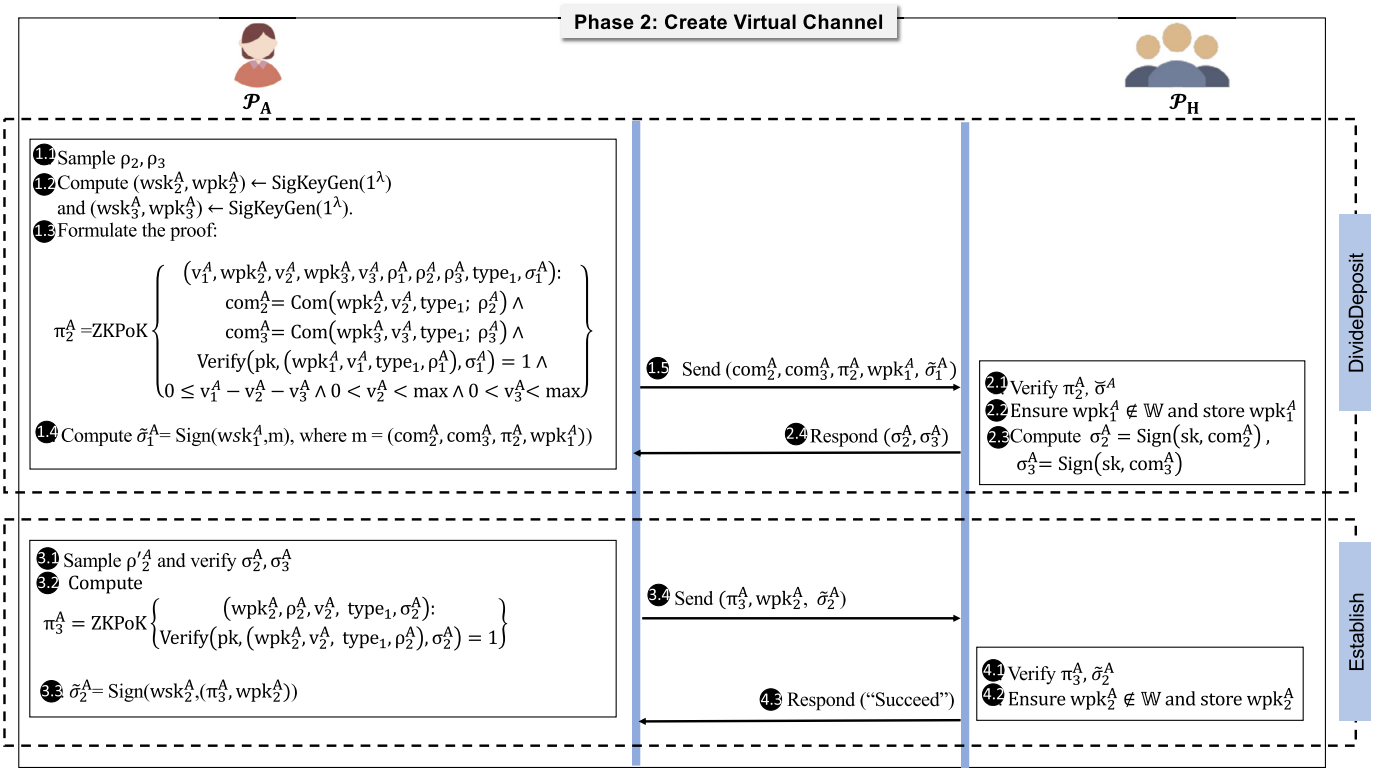


Fig. 3. The process of creating virtual channel.

identification wpk_1^A and sends the commitment com_1^A=Com(wpk_1^A; r_A) to $\mathcal{P}_H$ (steps 5.1-5.3 in Fig. 2). After receiving the commitment, $\mathcal{P}_H$ appends the wallet value v_1^A and the coin type $type_1$ to the receiving commitment com_1^A for generating a new commitment com'. Since the user $\mathcal{P}_A$ may commit an incorrect wallet value, PRI requires $\mathcal{P}_H$ rather than $\mathcal{P}_A$ to commit the wallet value. Further, $\mathcal{P}_H$ generates a PS signature σ_1^A on the commitment com', which can be regarded as a certification of the wallet (steps 6.1-6.3 in Fig. 2). Since the identification is committed, $\mathcal{P}_H$ can not trivially link $\mathcal{P}_A$ and $\mathcal{P}_B$ when they reveal the identifications to $\mathcal{P}_H$ in the create channel phase. Once $\mathcal{P}_A$ is convinced of the validity of σ_1^A, he/she can move to the next phase.

5.3.2. Create virtual channel

Here, we discuss how the users $\mathcal{P}_A$ and $\mathcal{P}_B$ create a virtual channel via the hub $\mathcal{P}_H$. To simplify, Fig. 3 only shows the interaction between the user $\mathcal{P}_A$ and the hub $\mathcal{P}_H$ since the interaction of $\mathcal{P}_B$ and $\mathcal{P}_H$ is analogous. Next, we detail the instruction between $\mathcal{P}_A$ and $\mathcal{P}_H$ as an example.

This phase begins with the users dividing their wallets into two parts based on their needs to avoid the deposit lock-in issue [10]. Concretely, the user $\mathcal{P}_A$ divides her wallet w_1^A=(v_1^A, wpk_1^A, wsk_1^A, $type_A$, σ_1^A) with the value v_1^A into two new wallets. Particularly, the new wallets respectively own the values v_2^A and v_3^A, where $v_1^A \ge v_2^A + v_3^A$. To identify the new wallets, $\mathcal{P}_A$ computes two new key pairs (wpk_2^A, wpk_2^A) and (wpk_3^A, wsk_3^A). Thus, the two new wallets can be written as w_2^A=(v_2^A, wpk_2^A, wsk_2^A, $type_A$, *) and w_3^A=(v_3^A, wpk_3^A, wsk_3^A, $type_A$, *). Then, $\mathcal{P}_A$ commits the detail of the new wallets on the commitments com_2^A and com_3^A, including wallet identifications, wallet values, and coin types (steps 1.1-1.2 in Fig. 3).

Further, to avoid the user $\mathcal{P}_A$ stealing coins more than his/her owns, $\mathcal{P}_A$ produces a proof π_2^A to state: 1) the commitments com_2^A and com_3^A are correct, 2) the old wallet certification σ_1^A is valid, and 3) the new wallet values are valid (step 1.3 in Fig. 3). Specifically, π_2^A proves the following statements: (i) The commitments com_2^A and com_3^A are computed by executing the Π_{cm}.Com algorithm; (ii) The deposit certification σ_1^A is a valid signature on the identification wpk_1^A, the initial value v_1^A, and the coin type $type_1$; (iii) The new wallet values v_2^A and v_3^A are legal, i.e. the new values are positive and the sum of them is no more than the initial value.[2]

Next, $\mathcal{P}_A$ utilizes the secret key wsk_1^A to compute a signature $\tilde{\sigma}_1^A$, which signs on the messages sent to $\mathcal{P}_H$. At this point, $\mathcal{P}_A$ can not claim an identification that does not belong to herself, because she can not generate a valid signature without the corresponding secret key. Further, the identification wpk_1^A is also revealed to $\mathcal{P}_H$ to avoid $\mathcal{P}_A$ spending the same wallet for twice (steps 1.4-1.5 in Fig. 3). Once $\mathcal{P}_H$ is convinced of the validity of π_2^A and $\tilde{\sigma}_1^A$, $\mathcal{P}_H$ gives the new certifications of the new wallets to $\mathcal{P}_A$ (steps 2.1-2.4 in Fig. 3). Note that the identifications and certifications of the new wallets are unrevealed to the hub $\mathcal{P}_H$. Therefore, $\mathcal{P}_H$ is unable to link the new wallets to the initial wallet and also can not obtain the balances and identifications of the new wallets.

Then, the users $\mathcal{P}_A$ and $\mathcal{P}_B$ respectively choose one of the new wallets to establish a virtual channel, i.e., the wallets w_2^A and w_2^B. Concretely, the users $\mathcal{P}_A$ and $\mathcal{P}_B$ respectively submit the identifications wpk_2^A and wpk_2^B of the selected wallets w_2^A and w_2^B to $\mathcal{P}_H$ for creating a virtual channel. To prove the validity of the selected wallet, $\mathcal{P}_A$ and $\mathcal{P}_B$ respectively produce proofs (π_3^A, π_3^B) for the statements that the selected wallets w_2^A and w_2^B include the valid certifications (σ_2^A, σ_2^B) (steps 3.1-3.2 in Fig. 3). Similarly, to avoid $\mathcal{P}_A$ and $\mathcal{P}_B$ using a wallet owned by other users, $\mathcal{P}_A$ and

[2] The equal sign in $v_1^A \ge v_2^A + v_3^A$ is established when the payment fee is added, i.e., $v_1^A = v_2^A + v_3^A + fee$.

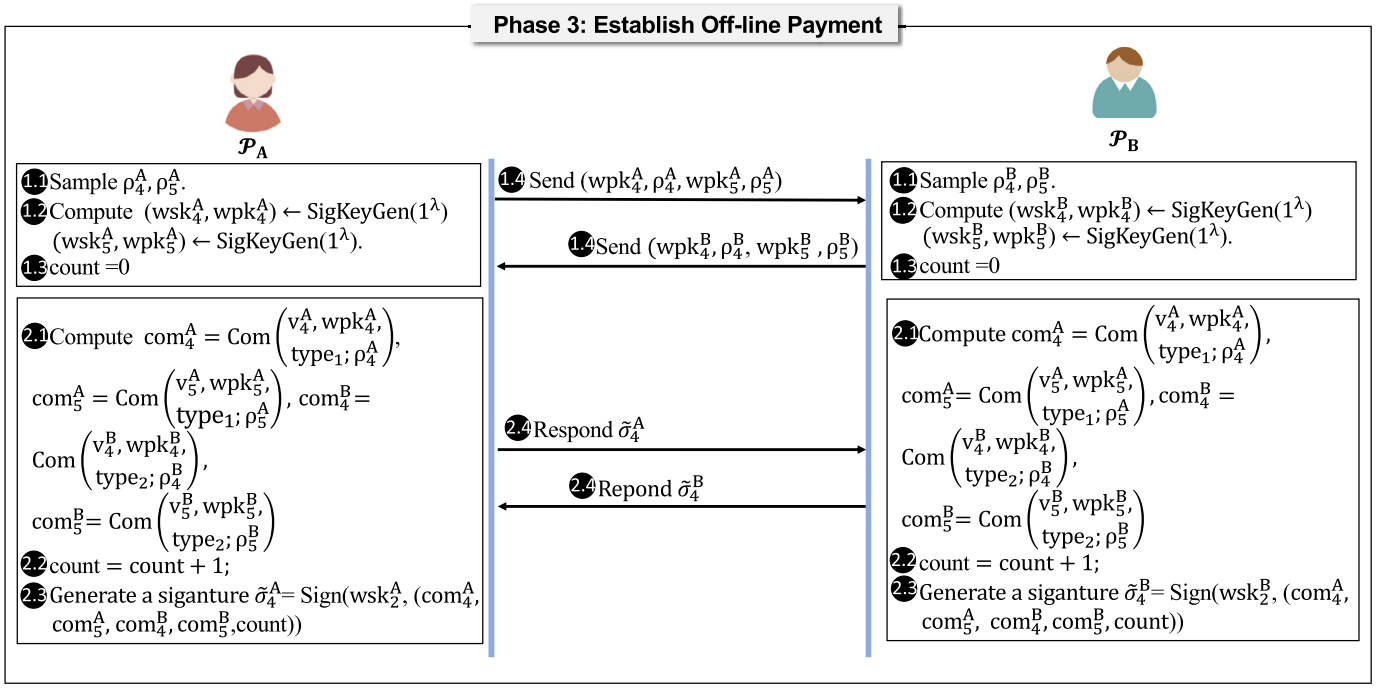


Fig. 4. The process of offline payment.

$\mathcal{P}_B$ respectively use the secret keys wsk_2^A and wsk_2^B to give signatures $\tilde{\sigma}_2^A$ and $\tilde{\sigma}_2^B$ on their identifications and proofs (steps 3.3 in Fig. 3).

Finally, $\mathcal{P}_H$ verifies the signatures and proofs from the users and checks if the identifications wpk_2^A and wpk_2^B are unspent before. If the verification succeeds, $\mathcal{P}_H$ will store the identification pairs (wpk_2^A, wpk_2^B) and respond a message "Succeed" to $\mathcal{P}_A$ (steps 4.1-4.3 in Fig. 3). Once the virtual channel is built, $\mathcal{P}_A$ and $\mathcal{P}_B$ can perform offline payments without $\mathcal{P}_H$.

5.3.3. Establish offline payment

Fig. 4 details an offline payment between $\mathcal{P}_A$ and $\mathcal{P}_B$. To clarify our description, we define the current balances of users in the channel as follows:

$$((v_4^A, v_5^B), (v_4^B, v_5^A)), \tag{1}$$

where (v_4^A, v_5^B) is the current balances of $\mathcal{P}_A$ and $\mathcal{P}_B$ in Bitcoin as well as (v_4^B, v_5^A) is the current balances of $\mathcal{P}_B$ and $\mathcal{P}_A$ in Ethereum. Since the virtual channel used is built over the wallets with v_2^A and v_3^A values, the current balance can be initialized by the following formula:

$$\begin{cases} (v_4^A, v_5^B) = (v_2^A, 0), \\ (v_4^B, v_5^A) = (v_2^B, 0) \end{cases} \tag{2}$$

In the offline payment phase, $\mathcal{P}_A$ and $\mathcal{P}_B$ firstly create four new wallets with the current balances and produces the key pairs to identify the new wallets. For instance, $\mathcal{P}_A$ generates wpk_4^A to identify the wallet with the v_4^A value. To commit the identifications, $\mathcal{P}_A$ and $\mathcal{P}_B$ sample some random numbers, i.e., ρ_4^A, ρ_5^A, ρ_4^B, ρ_5^B (steps 1.1-1.2 in Fig. 4). Further, to prevent the user from submitting outdated balances to the hub, a counter *count* is initialized to record the number of offline payments. Next, $\mathcal{P}_A$ and $\mathcal{P}_B$ exchange the sampled random numbers and the new identifications (steps 1.3-1.4 in Fig. 4). Then, $\mathcal{P}_A$ and $\mathcal{P}_B$ make a commitment on the current balances, the coin types, and the new wallet identifications (step 2.1 in Fig. 4). $\mathcal{P}_A$ and $\mathcal{P}_B$ simultaneously update the counter *count* and respectively makes use of the secret keys (wsk_2^A, wsk_2^B) of the initial wallets to sign on the commitments and the counters (steps 2.2-2.3 in Fig. 4). The signatures generated here indicate that $\mathcal{P}_A$ and $\mathcal{P}_B$ agree on the new balances of the new wallets (step 2.4 in Fig. 4). If $\mathcal{P}_A$ and $\mathcal{P}_B$ have multiple offline payments, they repeat the above steps and finally submit the latest commitments and signatures to the hub $\mathcal{P}_H$ while closing their virtual channel.

5.3.4. Close virtual channel

In this phase, $\mathcal{P}_A$ and $\mathcal{P}_B$ submit the latest states of the virtual channel to $\mathcal{P}_H$ to close the channel. Here, the latest states consist of the latest balance, coin type, and wallet identification. After establishing multiple offline payments, the latest states of the virtual channel composed of two tuples can be denoted as follows:

$$\begin{cases} state_1 = (v_4^A, type_1, wpk_4^A), state_2 = (v_5^A, type_2, wpk_5^A) \\ state_3 = (v_4^B, type_2, wpk_4^B), state_4 = (v_5^B, type_1, wpk_5^B) \end{cases} \tag{3}$$

More concretely, $state_1$ and $state_3$ represent the wallet details in Bitcoin as well as $state_3$ and $state_4$ denote the wallet details in Ethereum. As shown in Fig. 5, $\mathcal{P}_A$ and $\mathcal{P}_B$ have to ensure that the users can not claim the sum of the committed balances over the initial balance of the virtual channel. Thus, $\mathcal{P}_A$ and $\mathcal{P}_B$ respectively start with generating zk-proofs to prove that the latest states are committed correctly and the latest balances in the state are legal. Take the zk-proof π_4^A generated by $\mathcal{P}_A$ as an example, π_4^A proves the following statements: (i) the commitments com_4^A and com_4^B are generated over $state_1$ and $state_3$ by calling the algorithm Π_{cm}.Com and (ii) the sum of v_4^A and v_5^B in Bitcoin is not greater than the initial balance v_2^A as well as v_4^A and v_5^B are positive numbers. Similarly, $\mathcal{P}_B$ constructs the proof π_4^B (step 1.1 in Fig. 5). Further, to state the initial wallets' ownership and authenticate the sent message, the users utilize the secret keys wsk_2^A and wsk_2^B to provide signatures $\tilde{\sigma}_5^A$ and $\tilde{\sigma}_5^B$ on the commitments and proofs (steps 1.2-1.3 in Fig. 5).

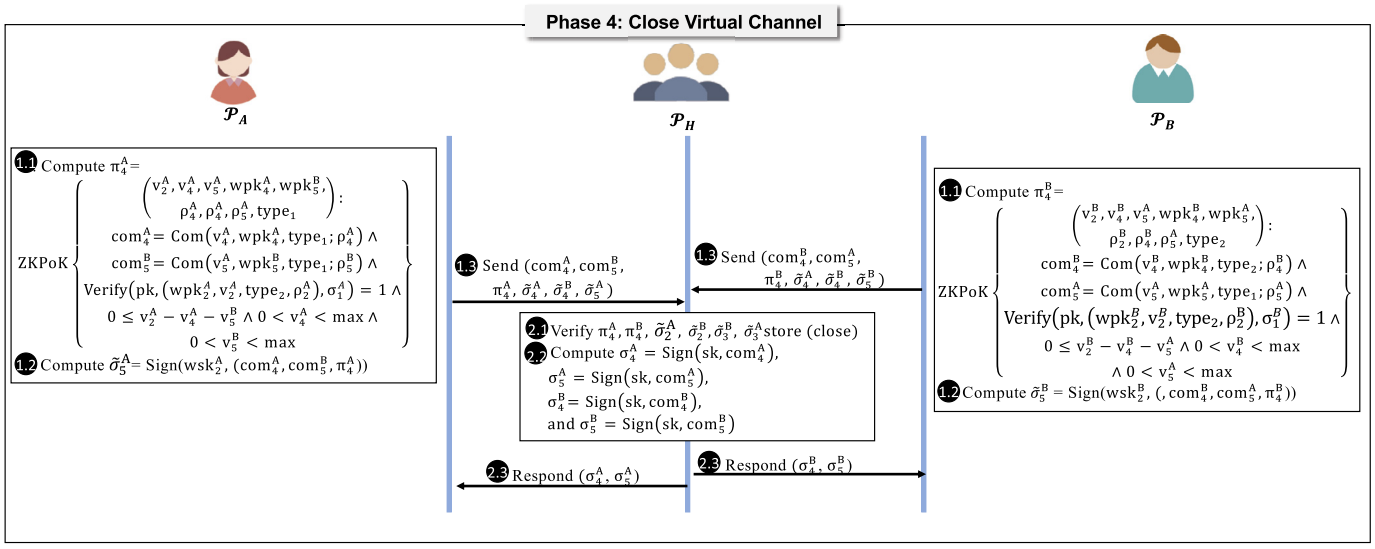


Fig. 5. The process of closing virtual channels.

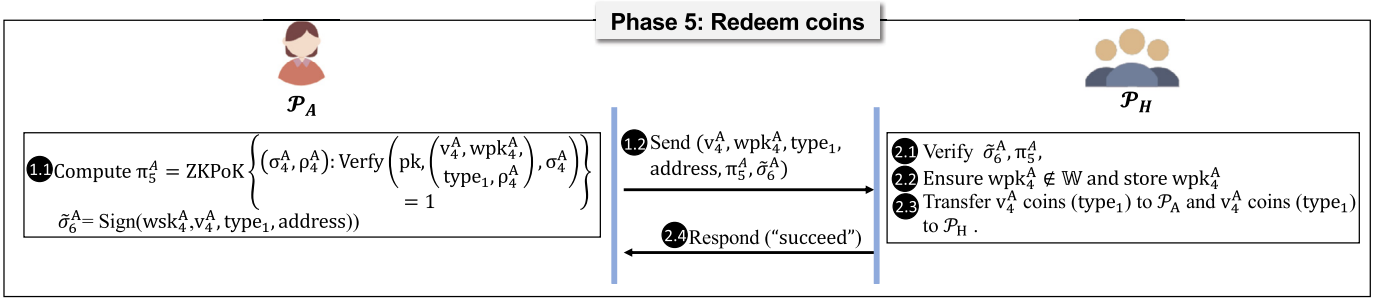


Fig. 6. The process of redeeming coins.

Upon receiving the request from the user, $\mathcal{P}_H$ first checks if the zk-proofs and signatures are valid. Then, $\mathcal{P}_H$ gives the new certifications (σ_4^A, σ_5^A) to $\mathcal{P}_A$ and (σ_4^B, σ_5^B) to $\mathcal{P}_B$ (steps 2.1-2.3 in Fig. 5). The new certifications represent the new wallets of the users which are indistinguishable from the certification generated in the deposit coins phase. Thus, $\mathcal{P}_A$ or $\mathcal{P}_B$ can use the new wallets to build a payment channel with other users. Note that since the initial wallet identifications are spent, the two users are willing to submit new states to generate the new wallets for avoiding losing their assets. Besides, considering that the close request may not be submitted on time by two users, we set a timer to remind the users to close the channel with the latest state. If none of the users submits the latest state in due time, the hub will remove the usage record of the initial identification to close the channel.

5.3.5. Redeem coins

In the redeem coins phase, the users and hub jointly create an on-chain transaction that transfers coins from the capital pool back to a user who sends a request to redeem his/her coins. Fig. 6 shows an illustrative example that $\mathcal{P}_A$ intends to redeem v_4^A coins (Bitcoin) in his/her unspent wallet. To ensure the correctness of the redeeming request from $\mathcal{P}_A$, the hub $\mathcal{P}_H$ requires $\mathcal{P}_A$ to provide a zk-proof. Next, $\mathcal{P}_A$ constructs the zk-proof π_5^A to state that the certification is correct. To avoid users double redeeming coins, $\mathcal{P}_H$ asks $\mathcal{P}_A$ to give the unspent wallet identification wpk_4^A. Thus, to state the ownership of the wallet identification, $\mathcal{P}_A$ computes a signature $\tilde{\sigma}_4^A$ over the sent messages with her secret key wsk_4^A to $\mathcal{P}_H$ (steps 1.1-1.2 in Fig. 6).

Upon receiving the redeem coins request, $\mathcal{P}_H$ checks the validity of the zk-proof π_5^A and the signature $\tilde{\sigma}_4^A$. If the verification succeeds and the identification wpk_4^A has not been spent before, $\mathcal{P}_H$ will store wpk_4^A (steps 2.1-2.2 in Fig. 6). Lastly, $\mathcal{P}_H$ broadcasts a request for encouraging the users in the capital pool to construct a transaction that transfers v_4^A assets from their jointly controlled account to $\mathcal{P}_A$. Note that the participating users will be rewarded with a certain fee which can motivate users actively respond to the request (steps 2.3-2.4 in Fig. 6).

6. Theoretical analysis

Here, we formal the security discussion and provide the functionality analysis for a PPCH.

6.1. Informal security discussion

6.1.1. Security definitions

We first give the formal security definitions about three properties: *unlinkability*, *confidentiality*, and *balance*.

1) Unlinkability. We denote $\mathcal{A}$ as an adversary playing the role of the hub. There is an experiment involving "users" interacting with the hub. Let $pp \leftarrow \mathsf{Setup}(1^\lambda)$ and $(sk_H, pk_H) \leftarrow \mathsf{SigKeyGen}(pp)$. We give pp and sk_H to $\mathcal{A}$. Then, $\mathcal{A}$ issues the following queries in any order:

Deposit $\mathcal{A}$ executes the Deposit protocol with $\mathcal{P}_i$ as:

$$\mathsf{Deposit}(\{\mathcal{P}_i(v_1^i, pp, acc_i)\}, \{\mathcal{A}(state)\}) \tag{4}$$

Where *state* is the adversary's state. Also, we denote the user's output as w^i, where w^i might be $\perp$.

DivideDeposit $\mathcal{A}$ executes the DivideDeposit protocol for dividing a wallet w^i with $\mathcal{P}_i$ as:

$$\text{DivideDeposit}(\{\mathcal{P}_i(pp, w, v_2, v_3)\}, \{\mathcal{A}(state)\}) \quad (5)$$

Where *state* is the adversary's state. (w_1^i, w_2^i) is denoted as the user's output, where (w_1^i, w_2^i) might be $(\perp, \perp)$. Besides, $\mathcal{A}$ obtains the wallet identification wpk^i.

Establish In this query, $\mathcal{A}$ executes the Establish protocol with $\mathcal{P}_i$ and $\mathcal{P}_j$ as:

$$\text{Establish}(\{\mathcal{P}_i(w^i, pp)\}, \{\mathcal{P}_j(w^j, pp)\}, \{\mathcal{A}(state)\}) \quad (6)$$

When $\mathcal{A}$ makes this query, it obtains the wallet identifications (wpk_1^i, wpk_2^j) or $\perp$.

CloseChannelb In this query, $\mathcal{A}$ recalls the CloseChannel protocol with $\mathcal{P}_i$ and $\mathcal{P}_j$ as:

$$\text{CloseChannel}(\{\mathcal{P}_i(\widetilde{\sigma}^i, w_3^i, w_4^i, pp)\}, \{\mathcal{P}_j(\widetilde{\sigma}^j, w_3^j, w_4^j, pp)\}, \{\mathcal{A}(state)\}) \quad (7)$$

The outputs of users are denoted as (w_3^i, w_4^i) and (w_3^j, w_4^j), or $\perp$.

Redeem $\mathcal{A}$ executes the Redeem protocol with $\mathcal{P}_i$ as:

$$\text{Redeem}(\{\mathcal{P}_i(w, addr)\}, \{\mathcal{A}(state)\}) \quad (8)$$

Where *state* is the adversary's state. The user's output is denoted as Redeemed or $\perp$.

We state that $\mathcal{A}$ is legal if it never builds channels from an illegal wallet where w is equal to $\perp$ or undefined. There is a simulator $\mathcal{S}(pp, aux,.)$ interacting with $\mathcal{A}$ in the Ideal experiment. Concretely, *aux* is an auxiliary trapdoor not available to the parties of the Real experiment. Due to the existence of the simulator, $\mathcal{A}$ can not distinguish the Real experiment and the Ideal experiment in non-negligible advantage:

- **Real experiment.** In this experiment, all responses are computed as described above.
- **Ideal experiment.** In this experiment, $\mathcal{A}$ queries the Deposit, Establish, CloseChannel, and Redeem protocols using the process mentioned above. Nevertheless, as for the DivideDeposit protocol, $\mathcal{A}$ interacts with $\mathcal{P}_i$ and $\mathcal{P}_j$ but instead interacts with $\mathcal{S}(pp, aux,.)$.

The unlinkability is reflected in the fact that the hub cannot distinguish which specific wallet is being used by users. Further, we only discuss wallets generated in the Deposit protocol, which cannot be linked to a specific wallet when used in the DivideDeposit protocol. Other protocols in the scheme also use similar mechanisms to guarantee unlinkability, and thus, we omit the security analysis here.

2) Confidentiality. The adversary $\mathcal{A}$ interacts with a collection of honest users $\mathcal{P}_1$,..., $\mathcal{P}_N$. Initialize $pp \leftarrow \text{Setup}(1^\lambda)$ and $(sk_H, pk_H) \leftarrow \text{SigKeyGen}(pp)$. We give pp and sk_H to $\mathcal{A}$. Next, $\mathcal{A}$ issues the queries as the same as the game in the definition of unlinkability property. Note that the queries can be issued in any order.

Similarly, we convey that $\mathcal{A}$ is legal if $\mathcal{A}$ never establishes a channel with a wallet where $w = \perp$ or where w is undefined. We say that $\mathcal{A}$ wins the game if it can distinguish the following experiments.

- **Real experiment.** All responses are computed as described above in this experiment.
- **Ideal experiment.** In this experiment, the Deposit and Redeem queries are addressed by the procedure described in the unlinkability property. In the remaining queries, $\mathcal{A}$ does not interact with $\mathcal{P}_i$ but instead interacts with the simulator $\mathcal{S}(pp, aux)$, where *aux* is an auxiliary trapdoor not accessible in the Real experiment.

3) Balance. $\mathcal{A}$ interacts with a party P, which can be a collection of honest users $\mathcal{P}_1$,..., $\mathcal{P}_N$ or an honest hub $\mathcal{P}_H$. Let $pp \leftarrow$ and $(pk_H, sk_H) \leftarrow \text{SigKeyGen}(pp)$. Set the balance counters $count_{\mathcal{A}} \leftarrow 0$, $final_{\mathcal{A}} \leftarrow 0$. For each user, it is given with pp and (pk_H, sk_H). Now $\mathcal{A}$ may issue the following queries in any order:

Deposit $\mathcal{A}$ executes the Deposit protocol with $\mathcal{P}$:
- If the party $\mathcal{P}$ is a user: Deposit$(\{\mathcal{P}_i(v_1^i, pp, acc_i)\}, \{\mathcal{A}(state)\}) \rightarrow w_i$ (or $\perp$). Then $count_{\mathcal{A}} \leftarrow v_1^A + count_{\mathcal{A}}$.
- If the party $\mathcal{P}$ is a hub: Deposit$(\{\mathcal{A}(state)\}, \{\mathcal{P}_i(v_1^i, pp, acc_i)\}) \rightarrow w_i$ (or $\perp$). Then $count_{\mathcal{A}} \leftarrow v_1^A + count_{\mathcal{A}}$.

Where *state* is denoted as the state of the adversary.

DivideDeposit When $\mathcal{A}$ has deposited several coins with party $\mathcal{P}$, $\mathcal{A}$ executes the DivideDeposit protocol with $\mathcal{P}_i$ (resp.$\mathcal{P}_H$) and obtains the closure message as:
- If the party $\mathcal{P}$ is a user: DivideDeposit$(\{\mathcal{P}_i(pp, w, v_2, v_3)\}, \{\mathcal{A}(state)\}) \leftarrow (w_1^i, w_2^i)$ (or $(\perp, \perp)$).
- If the party $\mathcal{P}$ is a hub: DivideDeposit$(\{\mathcal{A}(state)\}, \{\mathcal{P}_H(sk_H, pp)\}) \leftarrow (w_1^i, w_2^i)$ (or $(\perp, \perp)$).

Where *state* is denoted as the state of the adversary.

Establish $\mathcal{A}$ executes the Esatblish protocol with $\mathcal{P}$:
- If the party $\mathcal{P}$ is a pair of users ($\mathcal{P}_i$ and $\mathcal{P}_j$): Establish$(\{\mathcal{P}_i(w^i, pp)\}, \{\mathcal{P}_j(w^j, pp)\}, \{\mathcal{A}(state)\}) \rightarrow (wpk^i, wpk^j)$ (or $\perp$).
- If $\mathcal{A}$ plays the role of a user $\mathcal{P}_i$ (or $\mathcal{P}_j$): Establish$(\{\mathcal{A}(state)\}, \{\mathcal{P}_j(w^j, pp)\}, \{\mathcal{P}_H(\mathbb{W})\}) \rightarrow (wpk^i, wpk^j)$ (or $\perp$).

Where *state* is denoted as the state of the adversary.

CloseChannel When $\mathcal{A}$ has established a channel with party $\mathcal{P}_i$, it executes the CloseChannel with $\mathcal{P}$:
- If the party $\mathcal{P}$ is a pair of users ($\mathcal{P}_i$ and $\mathcal{P}_j$): CloseChannel$(\{\mathcal{P}_i(\widetilde{\sigma}^i, w_3^i, w_4^i, pp)\}, \{\mathcal{P}_j(\widetilde{\sigma}^j, w_3^j, w_4^j, pp)\}, \{\mathcal{A}(state)\}) \rightarrow ((w_3^i, w_4^i), (w_3^j, w_4^j))$ (or $\perp$).
- If $\mathcal{A}$ plays the role of a user $\mathcal{P}_i$ (or $\mathcal{P}_j$): CloseChannel$(\{\mathcal{A}(state)\}, \{\mathcal{P}_j(\widetilde{\sigma}^j, w_3^j, w_4^j, pp)\}, \{\mathcal{P}_H(sk_H, pp)\}) \rightarrow ((w_3^i, w_4^i), (w_3^j, w_4^j))$ (or $\perp$). Then $count_{\mathcal{A}} \leftarrow count_{\mathcal{A}} - w_3^i.v + w_4^i.v$.

Where *state* is denoted as the state of the adversary.

Redeem $\mathcal{A}$ executes the Redeem protocol with $\mathcal{P}$:
- If $\mathcal{A}$ plays the roles of a hub and the party $\mathcal{P}$ is a user: Redeem$(\{\mathcal{P}_i(w, addr)\}, \{\mathcal{A}(state)\}) \rightarrow$ Redeemed. Then update $final_{\mathcal{A}} \leftarrow final_{\mathcal{A}} + w.v$.
- If $\mathcal{A}$ plays the role of a user and the party $\mathcal{P}$ is a hub: Redeem$(\{\mathcal{A}(state)\}, \{\mathcal{P}_H(sk, acc)\}) \rightarrow$ Redeemed. Then update $final_{\mathcal{A}} \leftarrow final_{\mathcal{A}} + w.v$.

Where *state* is denoted as the state of the adversary.

As the same as the above property, we say that $\mathcal{A}$ is legal if $\mathcal{A}$ never establishes a channel with a wallet where $w_i = \perp$ or where w_i is undefined. We further restrict that the adversary $\mathcal{A}$ can not corrupt most of the users and the honest hub at the same time. $\mathcal{A}$ wins the games if $final_{\mathcal{A}} > count_{\mathcal{A}}$ after the queries of Redeem protocol.

6.1.2. Security analysis

We sketch the security proof of PRI based on the underlying cryptographic primitives. The NIZK proofs [32,23,33] are derived from the standard sigma protocol, which can be transferred to no-interactive protocols utilizing the Fiat-Shamir [36] in the random

oracle. Further, the security of the NIZK proof relies on the q-Strong Diffie-Hellman (q-SDH) assumption [31]. The PRI protocol inherits the security assumptions: Lysyanskaya-Rivest-Sahai-Wolf (LRSW) and Discrete Logarithm (DL) [37] from the signature and commitment schemes.

Theorem 6.1. *The PPCH satisfies the properties of unlinkability, confidentiality, and balance security under the assumptions that (1) the commitment scheme is secure, (2) the zero-knowledge system satisfies soundness and zero-knowledge, (3) the signature scheme satisfies existential unforgeability under chosen message attack.*

Proof. *1) Unlinkability.* To prove that our scheme matches this property, we describe a simulator $\mathcal{S}(pp, aux,.)$ so that with non-negligible advantage, an adversary $\mathcal{A}$ can not distinguish the Real experiment from the Ideal experiment. In the Real experiment, the adversary runs and responds following the above definition. In the Ideal experiment, the simulator will take the place of users to interact with $\mathcal{A}$. The simulator $\mathcal{S}$ takes the following operation while interacting with the adversary $\mathcal{A}$. Firstly, due to the requirement of the zk-proof system in our scheme, we need to initialize a simulation common reference string (CRS) and embed it in pp. Secondly, the simulator $\mathcal{S}$ takes the place of $\mathcal{P}_i$ in the DivideDeposit protocol with the following differences against the Real experiment: 1) the ZK simulation algorithm simulates the zk-proofs, 2) the commitments com_2^A and com_3^A are substituted by random messages, and 3) wpk_1^A is generated by the simulator. Further, the generation of wpk and signatures are the same as in the Real experiment.

To prove the distinguishability between the Real experiment and the Ideal experiment, we will start with the Real experiment and replace elements through a series of games until we arrive at the Ideal experiment where the adversary interacts with the simulator $\mathcal{S}$. We denote negl_1, negl_2 as negligible functions. Also, we use **Adv[Game i]** to represent the advantage of the adversary in distinguishing the distribution of the output of the **Game i** from the distribution of the output of Real experiments.

Game 0. This is the Real experiment.

Game 1. This is the same as Game 1 except that each zk-proofs generated by users in the DivideDeposit is simulated by the ZK simulation algorithm. If the zk-proof system satisfies the zero-knowledge property, **Adv[Game 1]** $\leq \mathsf{negl}_1(\lambda)$.

Game 2. This is the same as Game 1 except that the simulator substitutes the commitments com_2^A and com_3^A with random messages. If the commitment scheme in here satisfies the hiding property, then **Adv[Game 2]** - **Adv[Game 1]** $\leq \mathsf{negl}_2(\lambda)$.

Game 3. This is the same as Game 2 except that a random key substitutes the original value wpk from running the SigKeyGen algorithm. **Adv[Game 3]** - **Adv[Game 2]** = 0, because the distribution of the random key is consistent with the distribution of the original value.

Based on the above games, we obtain that **Adv[Game 3]** is negligible. Since the distribution of Game 3 is the same as the Ideal experiment, the adversary $\mathcal{A}$ can not distinguish the Ideal experiment from the Real experiment so our scheme satisfies the property of unlinkability.

2) Confidentiality. To prove property, we construct a simulator $\mathcal{S}(pp, aux,.)$ to interact with the adversary $\mathcal{A}$ in the Ideal experiment so that $\mathcal{A}$ can not distinguish the Ideal experiment from the Real experiment in negligible advantage. The operations of the simulator $\mathcal{S}$ are similar to the simulator of the proof of *unlinkability*. That is, $\mathcal{S}$ substitutes the commitments, zk-proofs, and the identifications of the wallet (i.e., the public key generated through the SigKeyGen algorithm) using the similar rules as above. Then, based on the operations of the simulator $\mathcal{S}$, we design a series of games to prove the indistinguishability between the Real experiment and the Ideal experiment. To simplify, we omit the description of the games here, which are the same as the games of unlinkability. Since the first game is identical to the Real experiment and the last game is identical to the Ideal experiment, we can state that the adversary $\mathcal{A}$ can not distinguish between the Ideal experiment and the Real experiment. Therefore, the property of confidentiality is guaranteed in our scheme.

3) Balance Security. We prove that our scheme satisfies this property by proving the indistinguishability between the Real experiment and the simulated experiment. Firstly, we design a series of games and denote that the adversary $\mathcal{A}$ wins in a game when $\mathcal{A}$ succeeds in claiming his/her balance more than his/her own. Based on those games, we can prove that the adversary in the Ideal experiment wins the game with negligible advantage, and the simulated experiment is identical to the Real experiment so that our scheme satisfies the balance property. Here, the simulator utilizes the following assumption to respond to the adversary in the simulated experiment: 1) the zk-proof system satisfies soundness and zero-knowledge, 2) the commitment scheme is secure, and 3) the signature schemes are existential unforgeability under chosen message attack (EU-CMA). Here, we start with the Real experiment to design a series of games as follows:

Game 0. This is the Real experiment.

Game 1. This is the same as Game 0 except that the witnesses of zk-proofs generated in the DivideDeposit, Establish, Close, and Redeem protocol are extracted. If the extraction is failed, the adversary can win this game. Under the soundness property of the zk-proof system, **Adv[Game 1]** $\leq \mathsf{negl}_1(\lambda)$.

Game 2. This is the same as Game 1 except that the adversary can win the game if $\mathcal{A}$ find a collision in the commitments generated in the DivideDeposit, Close, and Redeem. Since the commitment scheme is secure, the commitment satisfies the binding property so that the adversary can not change the values committed in the commitment. Therefore, **Adv[Game 2]**-**Adv[Game 1]** $\leq \mathsf{negl}_2(\lambda)$.

Game 3. This is the same as Game 2 except that the adversary wins the game if the adversary can replace $\mathcal{P}_H$ to construct signatures on the commitments made in the DivideDeposit, Close, and Redeem. The adversary succeeds in forging the signature with negligible possibility because The signature scheme used in signing the commitments is EU-CMA secure. Thus, **Adv[Game 3]**-**Adv[Game 2]** $\leq \mathsf{negl}_3(\lambda)$.

Game 4. This is the same as Game 3 except that the adversary wins the game if the adversary can substitute $\mathcal{P}_i$ (or $\mathcal{P}_j$) to forge signatures on the message sent from $\mathcal{P}_i$ (or $\mathcal{P}_j$) to $\mathcal{P}_H$. Due to the signature scheme satisfying EU-CMA security, the adversary can not forge these signatures with a non-negligible advantage. Under this assumption, **Adv[Game 4]**-**Adv[Game 3]** $\leq \mathsf{negl}_4(\lambda)$.

Based on the above analysis, we can obtain that the advantage of the adversary wins in the Game 4 is negligible, which indicates that the adversary can not perform the following events.

- The adversary forges zk-proofs to state that it owns a wallet authenticated by the hub $\mathcal{P}_H$ but not. Each valid zk-proof can extract a valid witness so that this situation happens in the Game 4 with negligible advantage.
- The adversary forges the signatures generated by $\mathcal{P}_i$ (or $\mathcal{P}_j$) while sending the message to $\mathcal{P}_H$. The adversary can not win the Game 4 under the signature scheme is EU-CMA secure, implying that it is unable to generate such signatures.
- The adversary forges the signature generated by $\mathcal{P}_H$ while authenticating the wallet of $\mathcal{P}_i$ (or $\mathcal{P}_j$). Similarly, this can not

Table 3
The performance of two-party payments in PRI.

Role	Cost	Deposit	Create Virtual Channel	Offline Payment	Close Virtual Channel	Redeem Coins	Total
Hub	Computation time (ms)	8.04	83.01	-	76.98	30.04	198.07
	Communication overhead (KB)	0.10	0.19	-	0.19	-	0.48
	Latency (ms)	119.79	204.08	-	198.05	-	521.92
User	Computation time (ms)	7.04	96.25	30.10	50.38	24.78	208.55
	Communication overhead (KB)	0.09	4.11	0.85	3.17	0.95	9.18
	Latency (ms)	192.13	475.24	215.19	322.57	231.94	1467.07

Table 4
The computation cost of a hub that establishes payments with users ranging from 10 to 100. Note that the payments here are two-party payments.

The number of users	10	20	30	40	50
Computation cost (s)	1.98	3.96	5.94	7.92	9.90
Communication cost (KB)	4.8	9.60	14.40	19.20	9.00
The number of users	**60**	**70**	**80**	**90**	**100**
Computation cost (s)	11.88	13.89	15.85	18.91	21.01
Communication cost (KB)	28.80	33.60	38.40	17.83	19.81

occur based on the assumption that the signature scheme is EU-CMA secure.

- Via making a collision in the commitments, the adversary represents $\mathcal{P}_i$ (or $\mathcal{P}_j$) to modify the balance of its wallets to obtain more tokens than they possess. Because of the binding property, the adversary can not win the Game 4 which implies that it can not occur.

Since the above events can not happen and the distribution of the Real experiment is identical to the Game 4, our scheme satisfies the balance security property. □

6.2. Functionality analysis

The PRI protocol satisfies functionality by realizing interoperability and reusability for the following reasons.

Interoperability. Since the deposit certifications of different assets are indistinguishable, the users can use their deposit certification to create a virtual channel with other users, no matter what kind of assets they possess. Thus, the users can exchange assets across different blockchains based on their willingness in PRI.

Reusability. The user is allowed to reuse his/her remnant assets to build a new virtual channel after closing the virtual channel. Also, he/she can divide his/her deposit into several parts to establish multiple channels simultaneously. This is ensured by the fact that 1) channels are unified in the deposit coins phase by the certification mechanism and 2) a deposit certification generated in the close virtual channel phase is identical to a deposit certification generated in the deposit coins phase.

7. Implementation and experiment

This section presents an evaluation of the performance and functionality of PRI.

The objective of this experiment is to demonstrate the versatility of PRI by implementing not only two-party payments but also multi-party payments. Specifically, the two-party payment is involved in two roles, i.e. (two) users and (one) hub; thus, we analyze the computation time, communication overhead, and latency of each role in different phases to state the efficiency and functionality of two-party payments. Similarly, we also evaluate the same metrics on multi-party payments which are initialized as three-party payments in our experiment. In order to conduct our experiments, we develop the PRI protocol in the Rust Language using the Arkworks framework [38]. Over the curve *bls12381*, we instantiate the PS signature [23] as our signature scheme and the Pedersen commitment scheme as our commitment scheme. Also, we implement our proof of zero-knowledge system as the proof system in [32] and set the range of asset values as $[0, 2^{16})$. We establish our evaluation on a ThinkStation P310 laptop with AMD Ryzen 53600 6-core processors × 12 and 8 GB of memory using Ubuntu Desktop-20.04.1-LTS operating system with Linux kernel 5.4.

7.1. Computation time

Here, we analyze each phase's computation time of two-party and multi-party payments.

Computation time in two-party payment. - We first measure the situation that only two users and one hub in the payment system and discuss the computation cost of each party in each phase. Based on the computation time of each party, we can demonstrate the efficiency of PRI in the real life. As shown in Table 3, the total computation cost of the hub is acceptable (only ~ 198.07 ms), and the cost of the user is ~ 208.55 ms. In detail, each user generates one commitment and a pair of keys to the hub for obtaining his/her deposit certification in the "create virtual channel" phase. The hub executes the key generation algorithm one time and computes one signature in the deposit phase. In the create virtual channel phase, each user generates two zk-proofs, two commitments, and two signatures on the message for building a new channel. The hub spends most of its time verifying the signatures and the zk-proofs while computing two signatures as deposit certifications. The hub is not involved during the offline payment phase and thus has no computation cost. As for a user, he/she needs to compute four commitments and one signature as the agreement of the user's balances in the channel. When closing the virtual channel, a user generates one zk-proof and one signature to certify the user's balances in the channel. Correspondingly, the hub is responsible for generating four signatures as new deposit certifications and verifying the user's proofs and signatures. To redeem coins, the user generates one signature and one zk-proof. The computation costs for the hub and the user in the create virtual channel and close virtual channel phases are relatively high because NIZK proofs are involved.

Further, to demonstrate the efficiency of PRI while multiple payments happening concurrently, we measure the total computation cost of a hub when the hub interacts with users ranging from 10 to 100. That is, a hub interacts with users pair ranging from 5 to 50. Table 4 shows that the cost is linear with the num-

Table 5
The performance of multi-party payments.

Cost	Deposit	Create Virtual Channel	Offline Payment	Close Virtual Channel	Redeem Coins	Total
Computation time (s)	0.04	0.53	0.05	0.46	0.16	1.24
Communication overhead (KB)	0.57	12.52	2.55	8.58	2.85	27.07
Latency (s)	0.10	0.79	0.23	0.75	0.25	2.12

ber of users. The cost here includes all five phases. Note that the computation cost can be optimized by parallel computation. That is, a hub can utilize multiple computation devices to deal with payments from different user pairs in order to decrease the computation time.

Computation time in multi-party payment. To demonstrate the functionality of PRI, we also measure the total computation cost of the multi-party payments happening among three users and one hub. Here, we only experiment with the computation cost of the situation where only three users and one hub establish one three-party payment. As shown in Table 5), the main computation cost happens in the create and close virtual channel ($\sim 0.53s$ and $\sim 0.46s$) phases since the users are required to generate a series of NIZK proofs to state the correctness of the deposit certifications. Further, each offline payment only costs $\sim 0.05s$ and consumes the computation resource of the three users but not the hub. Compared with each on-chain payment that requires a few minutes or even an hour of confirmation time, the total computation cost of the three users and the hub is acceptable ($\sim 2s$).

7.2. Communication overhead

To discuss the extra bandwidth of PRI, the experiment measures the communication overhead in the two-party and multi-party payments.

Communication overhead in two-party payment. We evaluate the communication overhead of PRI by measuring the size of the messages exchanged between a hub and a user in each phase. Note that there is one user pair and one hub in this experiment, and further, only one off-chain payment happening between the user pair. As shown in Table 3, the total communication overhead of the hub is ~ 0.48 KB. The communication overhead of each user is ~ 9.18 KB. Specifically, there is one round of offline communication between each user and hub for exchanging a deposit certification and a commitment in the deposit phase. To create a new virtual channel, a user communicates with the hub to divide his/her deposit into multiple parts at first. Then, in the second round, the involved users exchange messages with the hub in registering a new channel. As for the offline payment phase, the communication overhead between the users is ~ 0.85 KB. In the close virtual channel phase, each user interacts with the hub only once to send one proof, three signatures, and two commitments and receives two new deposit certifications. In the redeem coins phase, the user sends a request to redeem assets with a proof of the validity of the request and information about the redeemed assets.

Communication overhead in multi-party payment. Similarly, we also measure the communication cost of each party in a three-party payment in order to experiment with the extra bandwidth while applying PRI in the multi-party payment. The communication overhead of each phase is recorded in Table 5. Note that there are three users and one hub participating in the multi-party payments. Further, the three users only establish one off-chain payment during the experiment. As shown in Table 5, the total communication cost of three users and a hub is $\sim 27KB$. The main communication overhead happens in the create and close virtual channel phases since the users need to submit a series of zk-proofs to state the correctness of the submitted information.

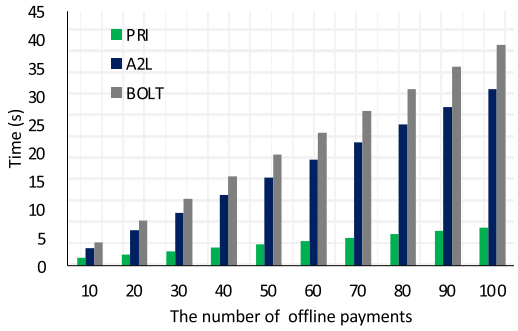


Fig. 7. The comparison among BOLT, A2L, and PRI on computation time.

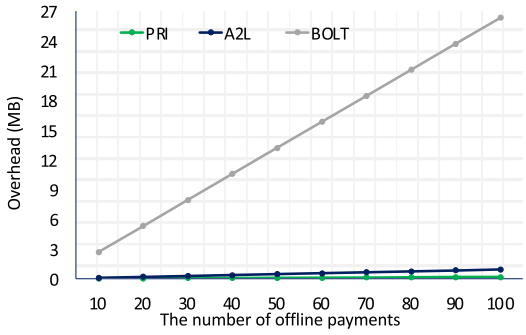


Fig. 8. The comparison among BOLT, A2L and PRI on communication cost.

7.3. Latency

To clarify the efficiency of PRI, we further discuss the latency of PRI in two-party and multi-party payments. The latency is computed by the formula: Latency = computation time + communication time. Specifically, we measure the communication time between two laptops connecting in the same LAN. The experiment results in Table 3 show the total latency of each user and hub who participate in two-party payments is no more than $2s$. Further, Table 5 reflects that the total latency in multi-party is $\sim 2.12s$ which is evaluated among three users and one hub. By these experiments, we can conclude that the latency of our solution is mainly caused by network latency as well as generating and verifying zk-proofs and signatures.

7.4. Discussion

We compare PRI with A2L [14] and BOLT [24] in Fig. 7 and Fig. 8 in terms of computation time and communication overhead in two-party payments. In terms of total computation time, PRI achieves ~ 2.3X speedup than A2L and ~ 3X speedup than BOLT over 10 offline payments occurring. The speedups would be more obvious as the number of offline payments grows. In terms of total communication overhead, PRI is 2.8X lower than A2L and 89X lower than BOLT. This is achieved because each offline payment

only occurs between users and does not involve NIZK proofs. The speedup will be more conspicuous if the number of offline payments becomes larger. Furthermore, our solution can be applied to multi-party payments, which are not realized in A2L and BOLT. In contrast to building multiple channels, users can build one channel for exchanging their assets.

8. Conclusion

In this paper, we presented PRI which simultaneously achieves relationship unlinkability, value confidentiality, deposit reusability, and blockchain interoperability, significantly alleviating the scalability issue in the blockchain. PRI has no restriction on payment amounts and supports multi-party payments in PCH. Further, we proved that PRI is secure and enjoys significant performance advantages over state-of-the-art PCH solutions.

Declaration of competing interest

The author(s) declare(s) that there are no conflicts of interest regarding the publication of this article.

Data availability

Data will be made available on request.

Acknowledgment

This work was supported by National Key Research and Development Plan of China under Grant No. 2020YFB1005600, National Natural Science Foundation of China under Grant Nos. 61825203 and U22B2028, Major Program of Guangdong Basic and Applied Research Project under Grant No. 2019B030302008, Guangdong Provincial Science and Technology Project under Grant No. 2021A0505030033, Science and Technology Major Project of Tibetan Autonomous Region of China under Grant No. XZ202201ZD0006G, National Joint Engineering Research Center of Network Security Detection and Protection Technology, Guangdong Key Laboratory of Data Security and Privacy Preserving, and Guangdong Hong Kong Joint Laboratory for Data Security and Privacy Protection.

References

[1] S. Nakamoto, Bitcoin: a peer-to-peer electronic cash system, in: Decentralized Business Review, 2008, pp. 21260–21268.
[2] G. Wood, Ethereum: a secure decentralized transaction ledger, 2014.
[3] B. Parno, J. Howell, C. Gentry, M. Raykova, Pinocchio: nearly practical verifiable computation, in: 2013 IEEE Symposium on Security and Privacy (SP), IEEE, 2013, pp. 238–252.
[4] V. Buterin, Chain Interoperability, R3 Research Paper, 2016.
[5] S. Tikhomirov, P. Moreno-Sanchez, M. Maffei, A quantitative analysis of security, anonymity and scalability for the lightning network, in: 2020 IEEE European Symposium on Security and Privacy Workshops (EuroSPW), IEEE, 2020, pp. 387–396.
[6] Transaction rate, https://en.bitcoin.it/wiki/. Accessed on 2021.
[7] A. Zamyatin, M. Al-Bassam, D. Zindros, E. Kokoris-Kogias, P. Moreno-Sanchez, A. Kiayias, W.J. Knottenbelt, Sok: communication across distributed ledgers, in: Financial Cryptography and Data Security: 25th International Conference (FC), Springer, 2021, pp. 3–36.
[8] J. Poon, T. Dryja, The bitcoin lightning network. Scalable o-chain instant payments, 2015.
[9] R. Network, What is the raiden network, 2019.
[10] J. Lind, O. Naor, I. Eyal, F. Kelbert, E.G. Sirer, P. Pietzuch, Teechain: a secure payment network with asynchronous blockchain access, in: Proceedings of the 27th ACM Symposium on Operating Systems Principles (SOSP), ACM, 2019, pp. 63–79.
[11] S. Dziembowski, S. Faust, K. Hostáková, General state channel networks, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS), ACM, 2018, pp. 949–966.
[12] G. Malavolta, P. Moreno-Sanchez, A. Kate, M. Maffei, S. Ravi, Concurrency and privacy with payment channel networks, in: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS), ACM, 2017, pp. 455–471.
[13] G. Malavolta, P. Moreno-Sanchez, C. Schneidewind, A. Kate, M. Maffei, Anonymous multi-hop locks for blockchain scalability and interoperability, in: 26th Annual Network and Distributed System Security Symposium (NDSS), NDSS, 2019, pp. 1–30.
[14] E. Tairi, P. Moreno-Sanchez, M. Maffei, A2l: anonymous atomic locks for scalability in payment channel hubs, in: 2021 IEEE Symposium on Security and Privacy (SP), IEEE, 2021, pp. 1834–1851.
[15] S. Dziembowski, L. Eckey, S. Faust, D. Malinowski, Perun: virtual payment hubs over cryptocurrencies, in: 2019 IEEE Symposium on Security and Privacy (SP), IEEE, 2019, pp. 106–123.
[16] L. Aumayr, P. Moreno-Sanchez, A. Kate, M. Maffei, Blitz: secure multi-hop payments without two-phase commits, in: 30th USENIX Security Symposium (USENIX Security). USENIX Association, 2021, pp. 4043–4060.
[17] E. Heilman, L. Alshenibr, F. Baldimtsi, A. Scafuro, S. Goldberg, Tumblebit: an untrusted bitcoin-compatible anonymous payment hub, in: Network and Distributed System Security Symposium (NDSS), NDSS, 2017, pp. 1–14.
[18] S. Dziembowski, L. Eckey, S. Faust, J. Hesse, K. Hostáková, Multi-party virtual state channels, in: Advances in Cryptology – EUROCRYPT 2019, Springer, 2019, pp. 625–656.
[19] P. Prihodko, S. Zhigulin, M. Sahno, A. Ostrovskiy, O. Osuntokun, Flare: an approach to routing in lightning network, White Pap. (2016) 144.
[20] G. Avarikioti, R. Scheuner, R. Wattenhofer, Payment networks as creation games, in: ESORICS 2019 International Workshops, Springer, 2019, pp. 195–210.
[21] Y. Shen, H. Tian, Y. Chen, K. Chen, R. Wang, Y. Xu, Y. Xia, S. Yan, Occlum: secure and efficient multitasking inside a single enclave of intel sgx, in: Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), ACM, 2020, pp. 955–970.
[22] T.P. Pedersen, Non-interactive and information-theoretic secure verifiable secret sharing, in: Advances in Cryptology CRYPTO, Springer, 1991, pp. 129–140.
[23] D. Pointcheval, O. Sanders, Short randomizable signatures, in: Cryptographers' Track at the RSA Conference (CT-RSA), Springer, 2016, pp. 111–126.
[24] M. Green, Miers I. Bolt, Anonymous payment channels for decentralized currencies, in: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS), ACM, 2017, pp. 473–489.
[25] C. Egger, P. Moreno-Sanchez, M. Maffei, Atomic multi-channel updates with constant collateral in bitcoin compatible payment channel networks, in: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS), ACM, 2019, pp. 801–815.
[26] L. Aumayr, M. Maffei, O. Ersoy, A. Erwig, S. Faust, S. Riahi, K. Hostáková, P. Moreno-Sanchez, Bitcoin-compatible virtual channels, in: 2021 IEEE Symposium on Security and Privacy (SP), IEEE, 2021, pp. 901–918.
[27] L. Aumayr, K. Abbaszadeh, M. Maffei, Thora: Atomic and privacy-preserving multi-channel updates, IACR Cryptol ePrint Arch, 2022.
[28] I. Miers, C. Garman, M. Green, A.D. Rubin, Zerocoin: anonymous distributed e-cash from bitcoin, in: 2013 IEEE Symposium on Security and Privacy (SP), IEEE, 2013, pp. 397–411.
[29] N. Zhang, K. Sun, D. Shands, W. Lou, Y.T. Hou, Trusense: information leakage from trustzone, in: IEEE Conference on Computer Communications (INFOCOM), IEEE, 2018, pp. 1097–1105.
[30] X. Zhang, Y. Wang, Y. Zou, Reconsidering leakage prevention in mapreduce, in: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE, 2020, pp. 1350–1351.
[31] D. Boneh, X. Boyen, Short signatures without random oracles, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2004, pp. 56–73.
[32] J. Camenisch, R. Chaabouni, et al., Efficient protocols for set membership and range proofs, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, 2008, pp. 234–252.
[33] M. Blum, P. Feldman, S. Micali, Non-interactive zero-knowledge and its applications, in: Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali, 2019, pp. 329–349.
[34] G.G. Dagher, B. Bünz, J. Bonneau, J. Clark, Boneh D. Provisions, Privacy-preserving proofs of solvency for bitcoin exchanges, in: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS), ACM, 2015, pp. 720–731.
[35] E. Cecchetti, F. Zhang, Y. Ji, A. Kosba, A. Juels, E. Shi, Solidus: confidential distributed ledger transactions via PVORM, in: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS), ACM, 2017, pp. 701–717.
[36] A. Fiat, A. Shamir, How to prove yourself: practical solutions to identification and signature problems, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 1986, pp. 186–194.

[37] A. Lysyanskaya, R.L. Rivest, A. Sahai, S. Wolf, Pseudonym systems, in: International Workshop on Selected Areas in Cryptography, Springer, 1999, pp. 184–199.
[38] Arkworks, https://github.com/arkworks-rs. Accessed on 2023.

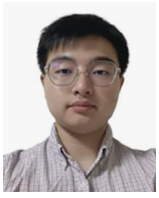
Yuxian Li obtained the B.S degree in Computer Science and Technology from Jinan University in June 2018. Currently, she is a Ph.D. student with School of Information Science and Technology in Jinan University. Her research interests include applied cryptography and blockchain.

Jian Weng is a professor and the Executive Dean with College of Information Science and Technology in Jinan University. He received B.S. degree and M.S. degree from South China University of Technology in 2001 and 2004 respectively, and Ph.D. degree at Shanghai Jiao Tong University in 2008. His research areas include public key cryptography, cloud security, blockchain, etc. He has published 80 papers in international conferences and journals such as CRYPTO, EUROCRYPT, ASIACRYPT, TCC, PKC, CT-RSA, IEEE TPAMI, IEEE TDSC, etc.

Wei Wu received his B.S. degree from the college of Mathematics and Informatics, South China Agricultural University, in 2018. And he is currently pursuing the M.S. degree with the College of Cyber Security, Jinan University. His research interests include cryptography and blockchain.

Ming Li received his B.S. in electronic information engineering from University of South China in 2009, and M.S. in information processing from Northwestern Polytechnical University in 2012. From 2016, he started his Ph.D. at Jinan University. His research interests include blockchain, crowdsourcing and its privacy and security.

Yingjiu Li is currently a Ripple Professor in the Computer and Information Science Department at the University of Oregon. His research interests include IoT Security and Privacy, Mobile and System Security, Applied Cryptography and Cloud Security, and Data Application Security and Privacy. He has published over 140 technical papers in international conferences and journals, and served in the program committees for over 80 international conferences and workshops, including top-tier cybersecurity conferences and journals.



Haoxin Tu received his B.S. degree in electronic information engineering from Northeast Forestry University in 2017, and his M.S. degree in software engineering from the Dalian University of Technology in 2019. In 2019 and 2020, he started his Dual-degree Ph.D. study in software engineering at the Dalian University of Technology and in computer science at Singapore Management University, respectively. His research focuses on developing practical techniques and tools that can help improve the reliability and security of software systems.

Yongdong Wu Received the B.Eng and M.S. from Beijing University of Aeronautics and Astronautics, the Ph.D degree from Institute of Automation, Chinese Academy of Science, and the Master for Management of Technology from National University of Singapore. He is a Professor of Jinan University, China, Adjunct Professor of Wuhan University, China and CTO of Mirai Electronics Pte Ltd Singapore. He was Head of System Security Lab, Institute for Infocomm Research, Singapore.

Robert H. Deng has been an AXA Chair Professor of cybersecurity and a Professor of information systems with the School of Information Systems, Singapore Management University, Singapore, since 2004. His current research interests include data security and privacy, multimedia security, and network and system security. He was a recipient of the Distinguished Paper Award from NDSS in 2012, the Best Paper Award from CMS in 2012, and the Best Journal Paper Award from IEEE Communications Society in 2017.