10-2023

# PriRPT: Practical blockchain-based privacy-preserving reporting system with rewards

Rui. SHI

Yang YANG
*Singapore Management University*, yyang@smu.edu.sg

Huamin. FENG

Feng. YUAN

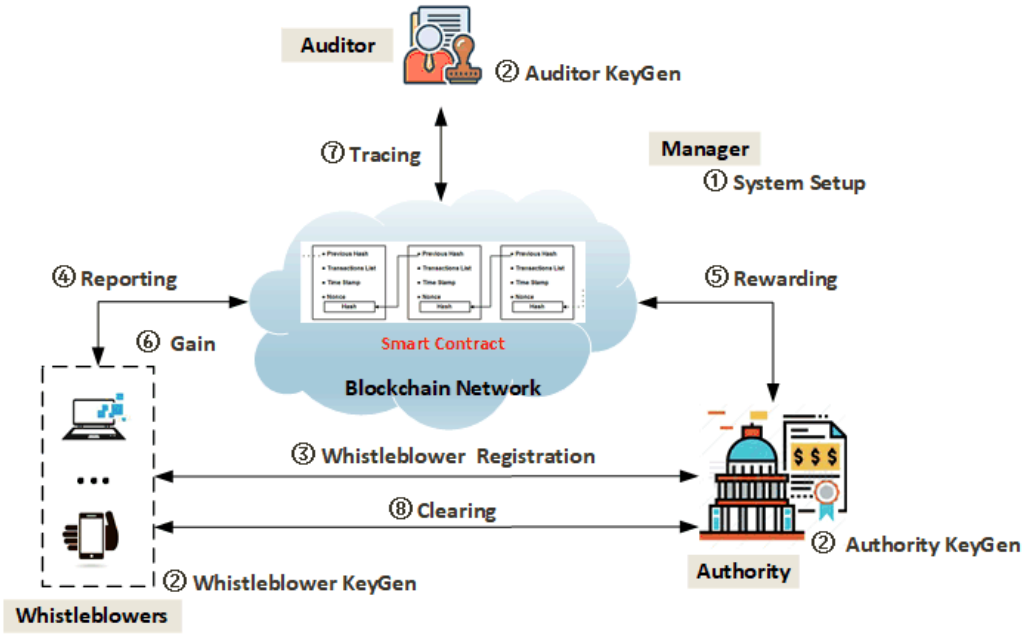
Huiqin. XIE

*See next page for additional authors*

# Graphical Abstract

**PriRPT: Practical Blockchain-Based Privacy-Preserving Reporting System with Rewards**

Rui Shi, Yang Yang, Huamin Feng, Feng Yuan, Huiqin Xie, Jianyi Zhang



# Highlights

**PriRPT: Practical Blockchain-Based Privacy-Preserving Reporting System with Rewards**

Rui Shi, Yang Yang, Huamin Feng, Feng Yuan, Huiqin Xie, Jianyi Zhang

- Reward whistleblowers with prizes
- Integrity and immutability of report messages
- Anonymous reporting and anonymous rewarding
- Low communication and computation overhead
- Formal system model and security proof

# PriRPT: Practical Blockchain-Based Privacy-Preserving Reporting System with Rewards

Rui Shi[a], Yang Yang[b], Huamin Feng[a,c], Feng Yuan[d], Huiqin Xie[c], Jianyi Zhang[c]

[a]*School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, 100876, China*
[b]*School of Computing and Information Systems, Singapore Management University, 188065, Singapore*
[c]*Beijing Electronic Science and Technology Institute, Beijing, 100070, China*
[d]*Institute 706, Second Academy of CASIC, Beijing, 100854, China*

**Abstract**

In order to obtain evidence of a crime timely, most authorities encourage whistleblowers to provide valuable reports by rewarding them with prizes. However, criminals will try their best to delete or tamper with the reports and even threaten and revenge the whistleblowers to escape punishment. Hence, to make the reporting system work, it is essential to ensure the integrity of reported messages and the anonymity of the reporting and rewarding procedures in the reporting system. Most existing schemes for this problem are generally based on ring signatures, which incur high computational overhead and imperfect anonymity. In this paper, we introduce a novel practical blockchain-based privacy-preserving reporting system with rewards dubbed as PriRPT. Specifically, the proposed scheme integrates the permissioned blockchain system, keyed-verification anonymous credential (KVAC), and structure-preserving signatures on equivalence classes (SPS-EQ) to provide reliable auditing of reports, and support anonymous reporting and anonymous rewarding simultaneously. In addition, we achieve higher efficiency in the reporting and rewarding protocol by replacing costly zero-knowledge proofs with KVAC and SPS-EQ. We also formalize the scheme along with security proof and provide rigorous evaluations on an open blockchain platform

*Corresponding author.
*Email address:* ruishi_mail@126.com. (Rui Shi)

(JUICE) and a personal laptop to demonstrate its practicability.

*Keywords:*
Anonymous credential, blockchain, structure-preserving signatures, reporting system

---

## 1. Introduction

Kindness and evil always go hand in hand in this world. Some government officials and business executives try to satisfy their desire for money and power through illegal means; what is more, some criminals gain financial benefits by infringing on the interests of others. In order to promote justice and punish crimes, authorities have tried to utilize various methods to find clues to solve cases, among which the quickest and most effective way to get evidence is by rewarding whistleblowers with incentive payment. To this end, the United States, China, the United Kingdom, Singapore, and other countries have launched online reporting systems to create a convenient environment for reporting crimes.

**Integrity of reports.** Ensuring the integrity and reliable auditing of the reported messages is critical to the function of the reporting system effectively. A reported message must go through multiple government officials in the actual setting. If malicious officials modify or delete the evidence, the value of reporting will be significantly reduced, and even the direction of the criminal investigation will be misleading. According to the news report of "People Daily" [1] in China, almost all criminal gangs are backed by corrupt officials who may have access to reporting systems and tamper with evidence. The phenomenon is not unique to China; Italy, Japan, and the United States have all reported officials jailed for colluding with criminals. Because there is corrupt personnel in the reporting system, the authenticity of the reported messages will be difficult to discern.

**Anonymous of reports.** Another important reason to discourage people from reporting is revenge from criminals. Revenge is a well-known motive for crime and violence, and according to a Finnish crime survey [2], about half of identity violations are motivated by revenge. In addition, nearly 70% whistleblowers have been targeted for revenge, according to "Legal Daily" in China. The high incidence of revenge in the reporting field significantly weakens people's enthusiasm to participate in the submission of evidence. Therefore, it is necessary to protect the real identity of the whistleblower

from being identified by any authority in the process of reporting and rewarding, and anonymity is the most basic security requirement in the reporting system.

**Drawbacks of existing schemes.** Some current reporting systems are constructed based on ring signatures [3, 4], resulting in high computational costs and flawed privacy-preserving, some have no formal definition and security proof [4, 5], and some have no convincing performance analysis [5]. We aim to develop a reporting system that satisfies security requirements, with formal security proof and superior performance.

### *1.1. Our Contributions*

We proposed a practical blockchain-based privacy-preserving reporting system with rewards called PriRPT to address the above security requirements. Our main contributions are listed below.

**Integrity of reports**. We use the smart contract to record reports and reward procedures on the blockchain to protect reported messages from being tampered with or deleted by malicious authorities. The immutability of the blockchain ensures the integrity of reports. Naturally, the integrity of reports can also be achieved at a lower cost by introducing a trusted third party to maintain the reporting information database. However, finding an authority to act as a trusted third party in the reporting system is problematic. The integrity of reporting can be achieved with blockchain without relying on a trusted third party. However, the downside of using blockchain is that smart contracts and consensus protocols must be executed each time a report is submitted, increasing reporting latency. As tested on JUICE [6], an open blockchain platform, the time cost of smart contracts and consensus protocols is on the order of seconds, and this delay is acceptable in the reporting system.

**Anonymity of reporting**. In a reporting system, the issuer of the whistleblowers' credentials is usually the same authority as the entity receiving the reports. Based on this fact, we create the report requests by using the keyed-verification anonymous credential (KVAC) [7, 8] where the credential issuer and verifier are the same entity, that is, both have credential issuing keys, thus avoiding complex bilinear map operations and improving the efficiency of reports verification.

**Anonymity of rewarding**. To improve the efficiency of the anonymous rewarding protocol, we use a structure-preserving signature on equivalence classes (SPS-EQ) [9, 10] to reduce the computational overhead of the system. In our scheme, the whistleblowers token consists of a Pedersen commitment

[11] of the reward value and an SPS-EQ signature. Since the SPS-EQ scheme can randomize the message and the corresponding signature, the whistleblower only requires three groups' exponentiation operations to realize the token's unlinkability.

Moreover, we formalize the system and security models of PriRPT and prove that it satisfies the security requirements of integrity, anonymity, and unforgeability. We implement the concrete scheme and compare its efficiency to state-of-the-art solutions on a personal laptop.

### *1.2. Organization*

The rest of the paper is organized as follows. Sec.2: Related work. Sec. 3: System model and security requirements of our PriRPT scheme. Sec. 4: Cryptographic primitives. Sec. 5: Construction of our PriRPT scheme. Sec. 6: Security analysis of our PriRPT scheme. Sec. 7: Performance analysis of our PriRPT scheme. Sec. 8: Conclusion.

## 2. Related Work

### *2.1. Reporting System*

Wang et al. [3] proposed a blockchain-based anonymous reporting scheme from ring signature [12], Monero [13], and batch verification [14, 15], and their scheme is a state-of-the-art solution. For the first time, they ensured the anonymity of whistleblowers both in the reporting and rewarding procedures. However, since they use the ring signature to achieve anonymous reporting and utilize the Monero, which also integrates the ring signature, to achieve anonymous rewarding, the computational cost of rewarding and reporting operations is linear with the number of whistleblowers in the group. In addition, their scheme does not achieve the integrity of reporting. Zou et al. [4] proposed a novel blockchain-based incentive anonymous reporting system, called ReportCoin, for management in the smart city. For the first time, they achieved both anonymous reporting and rewarding, and the reliability of reported messages via blockchain. Unfortunately, the ring signatures are used to design anonymous reporting, which leads to the computational cost of reporting being linear with the number of whistleblowers in the group; In addition, they still require building a self-designed IPFS-based blockchain system to implement anonymous incentives but have not given a formal definition and analysis of its security. Zou et al. [5] introduced a decentralized electronic reporting scheme from proxy signature [16] and blockchain. They

innovatively designed a post-quantum proxy signature based on modular lattices [17] and used it as the main block to achieve anonymity and unforgeability. Their scheme, however, does not support rewards for whistleblowers and has neither formal proof of security nor experimental results to evaluate performance.

Shi et al. [18] designed a privacy-preserving single sign-on system for cloud environments using attribute-based credentials [19], inner-product functional encryption [20], and structure-preserving signatures [21]. Their scheme can be used for anonymous reporting if the reported message is taken as an input message when computing the zero-knowledge proof of the credential-showing algorithm. However, the scheme does not support the integrity of reporting and the anonymous reward, and the function encryption is used to hide the authentication information, resulting in significant computational overhead.

Recently, Huang et al. [22] proposed a trustworthy mobile crowdsensing scheme that implements similar functions to the reporting system. They designed a blockchain-based MCS platform to provide a trustworthy platform and data integrity and designed a homomorphism data perturbation scheme to preserve the privacy of sensory data. Wang et al. [23] introduced a data reporting protocol with revocable anonymous authentication for edge-assisted intelligent transport systems by Pointcheval–Sanders signature [19] and the bivariate polynomial function [24]. Their data reporting protocol satisfies source authentication, traceability, revocability, non-frameability, and non-repudiation. These schemes implement anonymous reporting, which is the same as our reporting system; unfortunately, none allow for anonymous rewards.

Table 1 summarizes a detailed comparison between PriRPT and related works [3, 4, 5, 18, 22, 23]. The comparisons are conducted in terms of formal security proof, anonymous reporting, anonymous rewarding, and integrity of reports. *Formal security proof* means that a reporting scheme is formally proven to be secure. *Anonymous reporting* means that the authority cannot identify the real whistleblower in the reporting process. *Anonymous rewarding* means that the authority cannot identify the real whistleblower in the rewarding process. *Integrity of reports* means that no adversary can tamper with the reported messages. The reporting scheme in [4] and [5] are not formally proven secure. The scheme in [5] does not support anonymous rewarding. The state-of-the-art scheme [3] has been proven to be secure and support both anonymous reporting and anonymous rewarding, but it does

not ensure the integrity of reports. The schemes in [18], [22], and [23] are not strictly reporting systems, although they all have formal proofs and achieve anonymous reporting. Nevertheless, none of them [18, 22, 23] support anonymous rewards, and the schemes in [18] and [23] do not consider the integrity of reports. In comparison, only our scheme supports full functionality.

Table 1: Function Comparison with Related Works

| Scheme | Formal security proof | Anonymous Reporting | Anonymous Rewarding | Integrity of Reports |
|---|---|---|---|---|
| [3] | √ | √ | √ | × |
| [4] | × | √ | √ | √ |
| [5] | × | √ | × | √ |
| [18] | √ | √ | − | − |
| [22] | √ | √ | − | √ |
| [23] | √ | √ | − | − |
| PriRPT | √ | √ | √ | √ |

√: supported feature; ×: unsupported feature −: not applicable

### *2.2. Blockchain-Enabled Privacy-Preserving Authentication Mechanism*

Authentication mechanism that combines a blockchain system with an authentication protocol to realize privacy protection has many application values. In order to reduce the computational complexity of authentication mechanisms while protecting privacy, many schemes have been proposed recently.

Lu et al. [25] proposed a blockchain-based privacy-preserving authentication scheme for VANETs with a novel data structure named the Merkel Patricia tree (MPT). Unfortunately, their scheme uses multiple credentials to gain user anonymity, which results in a high computing overhead.

Fan et al. [26] proposed a secure and efficient authentication and data-sharing scheme for the Internet of Things based on blockchain. They designed an ID-based signature authentication for IoT to enable mutual authentication; however, this protocol does not achieve the anonymity of authentication.

Lin et al. [27] designed a conditional privacy-preserving authentication protocol for vehicular ad hoc networks based on the blockchain and key

derivation algorithm. Chhikara et al. [28] proposed a blockchain-based authenticated access control framework for medical database systems. However, the privacy protection of their schemes is based on the inherent properties of the blockchain.

Mei et al. [29] proposed a blockchain-enabled privacy-preserving authentication mechanism for transportation CPS with cloud-edge computing using identity-based ring signatures and elliptic curve cryptography. However, because their scheme uses the ring signature to design authentication protocol, the computation cost increases linearly with the increase in the number of public keys in the ring signature.

Li et al. [30] proposed a blockchain-based device authentication framework to ensure the security of data sources. They achieve decentralized device authentication using SRAM physical unclonable function (PUF) and Arbiter PUF; however, this strong assumption limits its application scenarios.

Table 2 summarizes a detailed comparison between PriRPT and existing blockchain-based authentication constructions [25, 26, 27, 28, 29, 30]. The comparisons are conducted regarding *authentication mechanism* and *anonymity*. *Anonymity* means that a scheme protects the privacy of the user's identity. *Authentication mechanism* means that the primary technique for obtaining anonymous usage. Unlike any of the above schemes, our system uses the KVAC [8] to implement anonymous authentication. It, therefore, does not rely on the inherent properties of blockchain, PUF, or multiple public keys/credentials to mix user identities.

Table 2: Comparison of authentication mechanism with existing blockchain-based authentication constructions

| Scheme | authentication mechanism | anonymity |
|---|---|---|
| [25] | multiple credentials | $\surd$ |
| [26] | – | $\times$ |
| [27, 28] | inherent properties of the blockchain | $\surd$ |
| [29] | identity-based ring signatures | $\surd$ |
| [30] | PUFs | $\surd$ |
| PriRPT | KVAC | $\surd$ |

–: not applicable

### *2.3. Keyed-Verification Anonymous Credential*

The anonymous attribute-based credentials (ABCs) were first proposed by Chaum [31] and Brands [32]. Subsequent schemes have been improved both in terms of functionality and efficiency through various variants of ABCs such as updatable credentials [33], decentralized credentials [34], delegatable credentials [35], and keyed-verification credentials [7].

Chase et al. [7] argue that in many scenarios where anonymous credentials can be deployed, the issuer of the credentials will also act as a verifier, meaning that the verifier has the issuing key. They formally define these so-called keyed-verification anonymous credentials (KVAC) and propose two concrete constructions. Subsequently, Barki et al. [36] proposed a more efficient KVAC scheme. Couteau et al. [37] constructed a KVAC scheme in the standard model. Camenisch et al. [38] introduced a fast KVAC scheme on standard smart cards. In the above schemes, the attributes of credentials are integers of a prime field, which limits the application of KVAC until Chase et al. [8] construct a new KVAC scheme in which attributes may be elements in the group. Since issuing credentials and receiving reports in the reporting system tend to be the same entity, we use the KVAC scheme [8] to let authority create credentials for the whistleblower's public key.

### *2.4. Structure-Preserving Signatures on Equivalence Classes*

Hanser et al. [9] introduced a notation of structure-preserving signatures on equivalence classes (SPS-EQ), which can randomize both signed messages and corresponding signatures simultaneously. Given a prime order group $\mathbb{G}$ and a projective space $(\mathbb{G}^*)^l$, they defined projective equivalence classes of messages $[\vec{M}]_{\mathcal{R}}$ based on the equivalence relation: $\mathcal{R}_{\vec{M}} = \{(\vec{M}, \vec{M}') \in (\mathbb{G}^*)^l \times (\mathbb{G}^*)^l | \exists s \in \mathbb{Z}_p^* : \vec{M}' = \vec{M}^s\}$. They formalized the security of SPS-EQ, defined as *signature adaptation*, such that randomized signatures are distributed like fresh signatures on any new representative of equivalence classes. Subsequently, Fuchsbauer et al. [10] proposed a more streamlined SPS-EQ scheme in the generic group model (GGM) [39] and constructed a constant-size anonymous credential based on it. The signature size of their scheme is only three group elements, and only two bilinear pairings are required for signature verification, which is the most efficient SPS-EQ scheme to date. In this paper, we use the SPS-EQ scheme [10] to create and update the signature of the reward value to achieve privacy-preserving rewards.

## 3. System Model and Security Requirements

### *3.1. System Model*

As shown in Fig. 1, the architecture of our PriRPT scheme consists of three types of parties: system manager ($\mathcal{M}$), authority ($\mathcal{A}$), and whistleblower ($\mathcal{W}$). In addition, the system also includes a permissioned blockchain [40] network that supports smart contracts. The specific role of each party is described as follows.

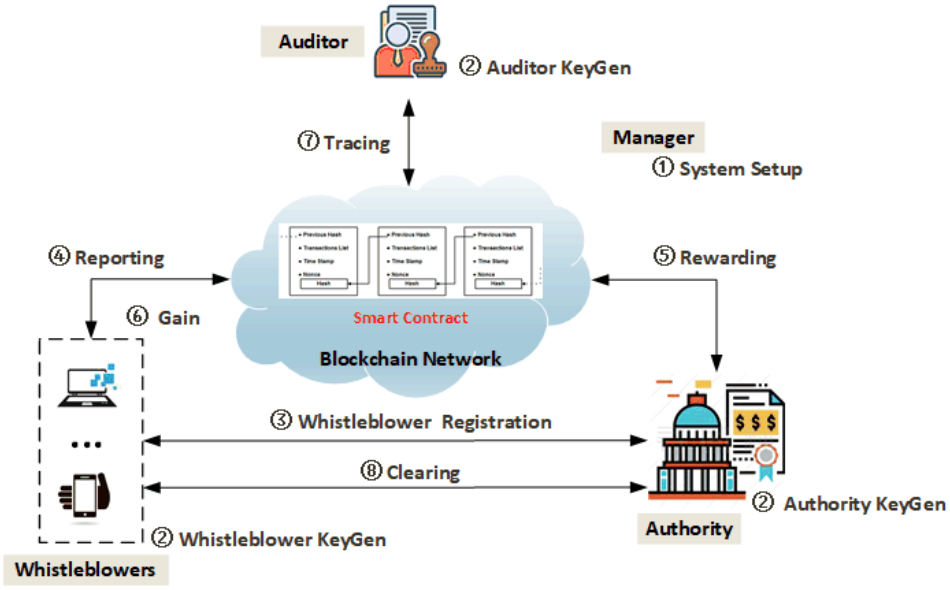



Figure 1: Architecture of PriRPT

- Blockchain: To guard the integrity and reliable auditing of reported messages, the information of reporting and rewarding will be chained into the blockchain with the help of a smart contract.
- Manager: $\mathcal{M}$ is a temporary and trusted party responsible for setting up the system (step ①) and creating the smart contract. Manager does not participate in any other system protocols, so it will be removed after initialization.
- Authority: $\mathcal{A}$ is an authority that is tasked to issue credentials and tokens for all whistleblowers (step ③), accept whistleblowers' reports, and reward whistleblowers who submit valuable reports (step ⑤).

• Whistleblower: $\mathcal{W}$ should register to $\mathcal{A}$ (step ③). To report the crime, $\mathcal{W}$ anonymously submits the evidence of a crime to $\mathcal{A}$ (step ④). If the report is valuable, $\mathcal{W}$ will gain the corresponding reward from $\mathcal{A}$ (step ⑥).

*3.2. Formal Definition*

The PriRPT scheme, which consists of a tuple (Setup, KeyGen, WbReg, Report, Reward, Gain) of probabilistic polynomial time (PPT) algorithms, is formally defined below.

① Setup: This algorithm is operated by the manager. It inputs a security parameter, then outputs the system parameters used in the algorithms KeyGen, WbReg, Report, Reward, and Gain. It also creates a smart contract to record the operations of Report and Reward.

② KeyGen: On input system parameters, authority and whistleblowers generate their private/public key pairs, respectively, where the public keys of the authority are public and known to other entities.

③ WbReg: This algorithm is operated by interacting between a whistleblower and the authority to issue a credential and a token for the whistleblower. The whistleblower and authority take their private and public keys as inputs. At the end of this algorithm, it outputs a credential and a token for the whistleblower.

④ Report: This algorithm is operated by the whistleblower to submit criminal evidence anonymously. It takes the whistleblower's credential, token, private/public key pair, and the reported message as inputs, then computes an anonymous proof of the credential and token and uploads it to the blockchain via the smart contract.

⑤ Reward: This algorithm is operated by the authority to reward valuable reporting. It inputs the authority's private and public keys, then gets a current reported message and the corresponding proof via the smart contract. If the proof is verified and the reported message is valuable for detecting the crime, the authority will create a reward and upload it to the blockchain via the smart contract.

⑥ Gain: This algorithm is operated by the whistleblower to gain the reward. It inputs the whistleblower's token and private/public key pair, then gets the corresponding reward via the smart contract. If the reward is verified, the whistleblower will update his token.

### *3.3. Overflow of PriRPT*

As shown in Fig. 1, the working flow of PriRPT is described as follows. The manager initializes the system, outputs the system parameter, and creates a smart contract (Setup, step ①). To join the system, the authority and whistleblowers generate their private/public key pairs and become a node of the permissioned blockchain, respectively (KeyGen, step ②). To make a legitimate party, each whistleblower must register with the authority and obtain a credential and a token (WbReg, step ③). When a whistleblower submits evidence of a crime, he computes proof of the credential and token and then uploads the proof and the reported message to the blockchain (Report, step ④). When the authority gets a new reported message from the blockchain, if the proof is verified and the reported message is valuable, he creates a reward and returns it to the blockchain. For an honest report, the whistleblower will get a reward and update his token (Gain, step ⑥).

### *3.4. Security Assumptions*

We assume that the manager and blockchain are fully trusted parties in the reporting system, where the manager is only responsible for initializing the system and does not participate in any other system protocols. The authority is assumed to be semi-honest because it honestly issues credentials for whistleblowers, verifies whistleblowers' reports, and creates rewards for honest whistleblowers. However, it is curious about the whistleblowers' real identities and may tamper with or delete the reporting messages. The whistleblowers are malicious; they may impersonate legitimate whistleblowers to submit reports or illegally obtain rewards that do not belong to them.

### *3.5. Security Requirements*

A privacy-preserving reporting scheme must satisfy the following security requirements in the practical environment.

**Integrity for Reports**. To prevent malicious deletion or tampering with the reported messages, the system should ensure the integrity of the reported messages.

**Unforgeability of Credentials**. The system should ensure that credential presentations are unforgeable to prevent impersonation of real whistleblowers when submitting the reported messages.

**Unforgeability of Tokens**. To prevent fraudulent claims of being a real whistleblower to obtain the authority's reward, the system should ensure the unforgeability of tokens.

**Anonymity of Reporting**. To ensure privacy, the system should guarantee anonymous reporting. In this process, the authority can verify the whistleblower's legitimacy and the authenticity of the reported message, while preventing identification of the real whistleblower by any adversary.

**Anonymity of Rewarding**. For privacy-preserving, the system should guarantee the anonymity of rewarding. The adversary cannot identify the real whistleblower from the rewarding process.

## 4. Cryptographic Primitives

### *4.1. Bilinear Pairing*

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups of prime order $p$. Let $G$ and $\widetilde{G}$ be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. The mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map if it has three properties: (1) *bilinearity*: $\forall G \in \mathbb{G}_1, \widetilde{G} \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, we have $e(G^a, \widetilde{G}^b) = e(G, \widetilde{G})^{ab}$; (2) *non-degeneracy*: $e(G, \widetilde{G}) \neq 1_{\mathbb{G}_T}$; (3) *computability*: $e$ can be efficiently computed. We denote a bilinear group $\mathcal{BL} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, p, G, \widetilde{G})$. Our scheme is based on the Type-III pairing [41], which means that there is no efficiently computable homomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$.

### *4.2. Zero-Knowledge Signature of Knowledge*

Zero-knowledge signature of knowledge ($\mathsf{ZKSoK}$) [42] for a NP-relation $\mathcal{R}$ with the language $L_{\mathcal{R}} = \{y : \exists x, (x, y) \in \mathcal{R}\}$ consists of the following algorithms.

– $\mathsf{Gen}(1^\lambda) \to pp$. On input a security parameter $\lambda$, outputs a public parameter $pp$.

– $\mathsf{Prove}(y, x, m) \to \pi$. On input a message $m$ and a relation $(x, y) \in \mathcal{R}$, outputs a $\mathsf{ZKSoK}$: $\pi = \mathsf{ZKSoK}\{x|(x, y) \in \mathcal{R}\}(m)$.

– $\mathsf{Verify}(y, \pi, m) \to 0/1$. On input a message $m$, a $\mathsf{ZKSoK}$ $\pi$ and a statement $y$. If $\pi$ is valid, return 1; otherwise, return 0.

The ZKSoK can be instantiated by Fiat-Shamir paradigm [43] incorporated with zero-knowledge protocols in [44]. A ZKSoK is *SimExt-secure* [42] if it satisfies correctness, simulatability and extractability.

### *4.3. Keyed-Verification Anonymous Credential*

Chase et al.'s keyed-verification anonymous credential (KVAC) [8] scheme consists of the following PPT algorithms:

– $\mathsf{KVAC.Setup}(\mathbb{G}) \rightarrow pp$: On input a cyclic group $\mathbb{G}$ of prime order $p$ , this algorithm selects public parameters $pp = (H_w, H_{w'}, H_{x_0}, H_{x_1}, H_y, H_V) \xleftarrow{R} \mathbb{G}$, so that the relative discrete logarithms are unknown.

– $\mathsf{KVAC.KeyGen}(pp) \rightarrow (sk, pk)$: On input public parameters $pp$, this algorithm selects $(w, w', x_0, x_1, y) \xleftarrow{R} \mathbb{Z}_p^*$, then computes $W = H_w^w, C_W = WH_{w'}^{w'}, I = \dfrac{H_V}{H_{x_0}^{x_0} H_{x_1}^{x_1} H_y^y}$. Finally, it outputs a secret key $sk = (w, w', x_0, x_1, y, W)$ and a public key $pk = (C_W, I)$.

– $\mathsf{KVAC.Issue}(sk, M) \rightarrow cred$: On input a secret key $sk$ and an attribute $M \in \mathbb{G}$, this algorithm selects $\nu \xleftarrow{R} \mathbb{Z}_p^*$ and $U \xleftarrow{R} \mathbb{G}$, and computes $V = WU^{x_0 + x_1 \nu} M^y$. Finally, it outputs a credential $cred = (\nu, U, V)$.

– $\mathsf{KVAC.Show}(cred, M) \rightarrow pst$: On input a credential $cred$ and an attribute $M \in \mathbb{G}$, this algorithm selects $t \xleftarrow{R} \mathbb{Z}_p^*$, then computes $Z = I^t, C_{x_0} = H_{x_0}^t U, C_{x_1} = H_{x_1}^t U^\nu, C_y = H_y^t M, C_V = H_V^t V$ and $\pi = \mathsf{ZKSoK}\{(\nu, t) : Z = I^t, C_{x_0} = H_{x_0}^t U, C_{x_1} = H_{x_1}^z U^\nu\}$. Finally, it outputs a credential presentation $pst = (Z, C_{x_0}, C_{x_1}, C_y, C_V, \pi)$.

– $\mathsf{KVAC.Vfy}(sk, pst) \rightarrow 0/1$: On input a secret key $sk$ and a credential presentation $pst$, this algorithm verifies $\pi$ and checks the equation $Z = \dfrac{C_V}{WC_{x_0}{}^{x_0} C_{x_1}{}^{x_1} C_y{}^y}$. If all of the above are verified, it outputs 1, otherwise, it outputs 0.

Chase et al.'s KVAC scheme [8] supports attribute that is group element and satisfies unforgeability, anonymity and key-parameter consistency.

### *4.4. Structure-Preserving Signatures on Equivalence Classes*

Given a bilinear group $\mathcal{BL} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, p, G, \widetilde{G})$ and a message equivalence relation : $\mathcal{R}_{\vec{M}} = \{(\vec{M}, \vec{M'}) \in (\mathbb{G}_1^*)^l \times (\mathbb{G}_1^*)^l : s \in \mathbb{Z}_p^* : \vec{M'} = \vec{M}^s\}$, Fuchsbauer et al. 's structure-preserving signatures scheme on equivalence classes (SPS-EQ) [10] over $\mathcal{R}_{\vec{M}}$ consists of the following PPT algorithms:

– $\mathsf{SPS\text{-}EQ.KeyGen}(\mathcal{BL}, l) \rightarrow (sk, pk)$: On input a bilinear group $\mathcal{BL}$ and a positive integer $l$, this algorithm selects $(y_1, \cdots, y_l) \xleftarrow{R} \mathbb{Z}_p^*$, and for all $i \in [1, l]$ computes $\widetilde{Y}_i = \widetilde{G}^{y_i}$. Finally, it outputs a secret key $sk = (y_1, \cdots, y_l)$ and a public key $pk = (\widetilde{Y}_1, \cdots, \widetilde{Y}_l)$.

– $\mathsf{SPS\text{-}EQ.Sign}(sk, \vec{M}) \rightarrow \sigma$: On input a secret key $sk$ and a message vector $\vec{M} = (M_1, \cdots, M_l)$, this algorithm selects $\kappa \xleftarrow{R} \mathbb{Z}_p^*$, and then computes $A = (\prod_{i=1}^l M_i^{y_i})^\kappa, \widetilde{B} = \widetilde{G}^{1/\kappa}, B = G^{1/\kappa}$. Finally, it outputs a signature

$\sigma = (A, \widetilde{B}, B)$.

– $\mathsf{SPS\text{-}EQ.ChgRep}(\vec{M}, \sigma, \mu, pk) \rightarrow (\vec{M}', \sigma')$: On input a message vector $\vec{M}$, a signature $\sigma$ on the message $\vec{M}$, a message converter $\mu \in \mathbb{Z}_p^*$, and the public key $pk$, this algorithm selects $\phi \xleftarrow{R} \mathbb{Z}_p^*$, and then computes $\vec{M}' = \vec{M}^{\mu} \leftarrow (M_1^{\mu}, \cdots, M_l^{\mu})$ and $\sigma' = (A', \widetilde{B}', B') \leftarrow (A^{\phi \cdot \mu}, \widetilde{B}^{1/\phi}, B^{1/\phi})$.

– $\mathsf{SPS\text{-}EQ.Verify}(\vec{M}, \sigma, pk) \rightarrow 0/1$: On input a message vector $\vec{M}$, a signature $\sigma$ on the message $\vec{M}$, and the public key $pk$, outputs 1 if $\prod_{i=1}^{l} e(M_i, \widetilde{Y}_i) = e(A, \widetilde{B})$ and $e(B, \widetilde{G}) = e(G, \widetilde{B})$, and 0 otherwise.

Fuchsbauer el al. 's SPS-EQ [10] scheme is existentially unforgeable under chosen message attacks (EUF-CMA) and has perfect adaptation of signatures.

## 5. Construction of PriRPT

### *5.1. High-Level Overview*

This section presents a concrete construction of PriRPT based on building blocks, including the smart contract, keyed-verification anonymous credential, structure-preserving signatures on equivalence classes, and zero-knowledge signature of knowledge. Our fundamental ideas consist of the following three points. (1) We treat the credential issuer and the authority receiving the reports as one entity, and the authority computes a KVAC [8] credential on the whistleblower's public key and outputs it as the whistleblower's credential. The KVAC scheme provides anonymous authentication while avoiding the time-consuming bilinear map operations required to verify credentials. (2) Each whistleblower independently generates a commitment [11] of rewards and obtains an SPS-EQ [10] signature on the commitment, and the commitment and its signature together form the whistleblower's token. The whistleblower can create an updated token with low computational cost by utilizing the feature that SPS-EQ randomizes the commitment and signature. (3) We use ZKSoK [42] to prove that both the credential and token are well-formed and use smart contracts to chain the intermediate processes to the blockchain.

Before presenting the system design, we first introduce the smart contract design in our system.

**Algorithm 1**: Smart Contract on PriRPT

```
address public Manager; % the system Manager.
structure Rpt
% Define the structure of report from the whistleblower.
    uint256[] report; % presentations and reported message.
    uint256[] reward; % reward.
mapping (address => Rpt) public rptList;
function PriRpt()
% Constructor that is invoked automatically when deployed.
    Manager = msg.sender;
function uploadReport(address _authority, uint256[] _report)
% Invoked by the whistleblower to upload a reported message.
    require(rptList[_authority].report == Null)
    rptList[_authority].report = _report;
function getReport(address _authority)
% Invoked by the authority to obtain a reported message.
    require(msg.sender == _authority);
    return rptList[_authority].report;
function uploadReward(address _authority, uint256[] _reward)
% Invoked by the authority to reward the valuable report.
    require(msg.sender == _authority);
    rptList[_authority].reward = _reward;
function getReward(address _authority)
% Invoked by the whistleblower to obtain the reward.
    return rptList[_authority].reward;
function deleteReport(address _authority)
% Invoked by the authority to clear the report structure.
    require(msg.sender == _authority);
    rptList[_authority].report = Null;
    rptList[_authority].reward = Null;
```

### *5.2. Smart Contract*

Smart contracts, as the fundamental building block of decentralized applications, are computer programs stored on the blockchain system (e.g.,

Ethereum [45] and Hyperledger [46]) that execute automatically when predetermined conditions are met. A smart contract defines some executable logic in a high-level language (e.g., *Solidity* language supported by Ethereum and *go* language supported by Hyperledger) and is deployed to the blockchain after being compiled into byte-code. It contains multiple data structures and functions that can be triggered by initiating a transaction and calls from other smart contracts, and all changes to the data structures are chained into the blockchain.

In PriRPT system, to provide integrity of reports and reliable auditing, we employ smart contracts to record the processes of reporting and rewarding. As shown in Algorithm 1, the manager deploys the smart contract (PriRpt). A whistleblower must upload his report (including the reported message and its prepared proof of the credential and token) into the smart contract via invoking uploadReport function. Then, the authority is responsible for monitoring the smart contract for a new report (getReport) and responds to it if the report is verified and valuable via invoking uploadReward function. Finally, honest whistleblowers can be rewarded with authority via invoking getReward function. In order to release the storage cost of smart contracts, the authority periodically clears the list of reports via invoking deleteReport function.

Note that functions uploadReport and uploadReward change the storage of smart contracts, so the consensus nodes will chain all reports and rewards into the blockchain for integrity and auditing. However, getReport and getReward are view-type functions, meaning their invocation does not require transaction confirmation.

### *5.3. System Design*

The notations used in the scheme are summarized in Table 3.

We now will describe the Setup, KeyGen, WbReg, Report, Reward and Gain algorithms in our system.

• Setup: On input a security parameter $\lambda$, the manager performs the following procedures to set up the system.

1. Generate a Type-III bilinear group $\mathcal{BL} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, p, g, \widetilde{g})$, and the basic operations of all algorithms in the system are defined on $\mathcal{BL}$.
2. For setting up the KVAC scheme, generate the public parameters by invoking $pp_{\mathsf{KVAC}} \leftarrow \mathsf{KVAC.Setup}(\mathbb{G}_1)$.
3. Select two different collision-resistant hash functions: $\mathsf{HASH}_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$ and $\mathsf{HASH}_2 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$.

Table 3: Summary of Notations

| Notation | Description |
|---|---|
| $\lambda/\epsilon(\lambda)$ | security number/negligible function |
| $\mathsf{HASH}_1$ | hash functions: $\{0,1\}^* \to \mathbb{G}_1$ |
| $\mathsf{HASH}_2$ | hash functions: $\{0,1\}^* \to \mathbb{Z}_p^*$ |
| $pp$ | system parameters |
| $\mathcal{BL}$ | bilinear group |
| $ask/apk$ | private/public key of authority |
| $wsk/wpk$ | private/public key of whistleblower |
| $cred$ | credential of whistleblower |
| $tok$ | token of whistleblower |
| $v$ | accumulated reward value |
| $m$ | reported message |
| $report$ | a report from whistleblower |
| $reward$ | a reward from authority |
| $v'$ | reward value of a reporting |

4. Output the system parameters $pp = (\mathcal{BL}, pp_{\mathsf{KVAC}}, \mathsf{HASH}_1, \mathsf{HASH}_2)$.
5. Deploy the smart contract described in Algorithm 1 to the blockchain system.

• KeyGen: On input the system parameters $pp$, each participating entity generates their private and public keys by performing the following procedures.

– Authority:
1) To issue credentials for whistleblowers, generate the issuing key by invoking $(sk_{\mathsf{KVAC}}, pk_{\mathsf{KVAC}}) \leftarrow \mathsf{KVAC.KeyGen}(pp_{\mathsf{KVAC}})$.
2) To create a reward for whistleblowers, generate the signing key by invoking $(sk_{\mathsf{SPS\text{-}EQ}}, pk_{\mathsf{SPS\text{-}EQ}}) \leftarrow \mathsf{SPS\text{-}EQ.KeyGen}(\mathcal{BL}, 2)$.
3) Choose $(z_1, z_2, z_3) \xleftarrow{R} \mathbb{Z}_p^*$, then compute the parameters of commitment: $Z_i = g^{z_i}$ for $i = 1, 2, 3$.
4) Output the authority's private key $ask = (sk_{\mathsf{KVAC}}, sk_{\mathsf{SPS\text{-}EQ}}, z_1, z_2, z_3)$ and public key $apk = (pk_{\mathsf{KVAC}}, pk_{\mathsf{SPS\text{-}EQ}}, Z_1, Z_2, Z_3)$.

– Whistleblower:
Choose the whistleblower's private key $wsk \xleftarrow{R} \mathbb{Z}_p^*$, then compute his public key $wpk = g^{wsk}$.

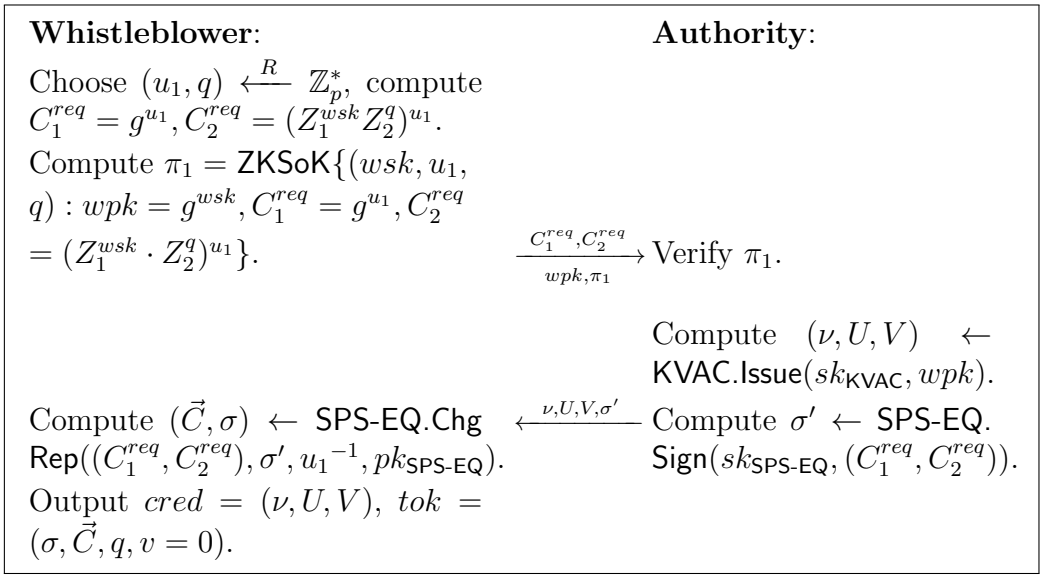



Figure 2: Whistleblower Registration Algorithm

• WbReg: As shown in Fig. 2, to join the reporting system, a new whistleblower must apply for a credential *cred* and a token *tok* from the authority. This is an interactive protocol detailed as follows:

– The whistleblower performs the following procedures to send a registration request to the authority and provide evidence that he is allowed to operate as a whistleblower.
  1) Choose $(u_1, q) \xleftarrow{R} \mathbb{Z}_p^*$, then compute $C_1^{req} = g^{u_1}, C_2^{req} = (Z_1^{wsk} Z_2^q)^{u_1}$;
  2) Compute $\pi_1 = \mathsf{ZKSoK}\{(wsk, u_1, q) : wpk = g^{wsk}, C_1^{req} = g^{u_1}, C_2^{req} = (Z_1^{wsk} Z_2^q)^{u_1}\}$;
  3) Send $(wpk, C_1^{req}, C_2^{req}, \pi_1)$ to the authority.
– The authority performs the following procedures to verify the registration request and returns a credential and a token for the whistleblower.
  1) Verify $\pi_1$;
  2) Create a credential by invoking $(\nu, U, V) \leftarrow \mathsf{KVAC.Issue}(sk_{\mathsf{KVAC}}, wpk)$;
  3) Compute $\sigma' \leftarrow \mathsf{SPS\text{-}EQ.Sign}(sk_{\mathsf{SPS\text{-}EQ}}, (C_1^{req}, C_2^{req}))$;
  4) Return $(\nu, U, V, \sigma')$ to the whistleblower.
– The whistleblower performs the following procedures to obtain their credential and token.
  1) Compute $(\vec{C}, \sigma) \leftarrow \mathsf{SPS\text{-}EQ.ChgRep}((C_1^{req}, C_2^{req}), \sigma', {u_1}^{-1}, pk_{\mathsf{SPS\text{-}EQ}})$;
  2) Set $cred = (\nu, U, V)$, $tok = (\sigma, \vec{C}, q, v = 0)$.

*Remark 1:* A whistleblower's credential is a KVAC credential $cred = (\nu, U, V)$ on the public key $wpk$. A whistleblower's token is a Pedersen commitment $\vec{C} = (C_1, C_2) = (g, Z_1^{wsk} Z_2^q Z_3^v)$ and an SPS-EQ signature $\sigma$ on $\vec{C}$, where $wsk$ is the whistleblower's private key, $q$ is a randomization value for the commitment, $v$ is the accumulated reward value of the whistleblower (the initial value is 0).

• Report: In order to anonymously submit a reported message $m \in \{0,1\}^*$, a whistleblower with private key $wsk$, public key $wpk$, credential $cred$, and token $tok$ performs the following procedures.

1. To prove his legitimacy to authority, the whistleblower chooses $t \xleftarrow{R} \mathbb{Z}_p^*$, then computes a credential presentation of KVAC: $Z = I^t, C_{x_0} = H_{x_0}^t U, C_{x_1} = H_{x_1}^t U^\nu, C_y = H_y^t g^{wsk}, C_V = H_V^t V$;
2. To obtain rewards anonymously, the whistleblower chooses $u_2 \xleftarrow{R} \mathbb{Z}_p^*$, then randomizes his token: $(\vec{C}', \sigma') \leftarrow \mathsf{SPS\text{-}EQ.ChgRep}(\vec{C}, \sigma, u_2, pk_{\mathsf{SPS\text{-}EQ}})$;
3. Compute $\pi_2 = \mathsf{ZKSoK}\{(\nu, wsk, t, q, u_2, U) : Z = I^t, C_{x_0} = H_{x_0}^t U, C_{x_1} = H_{x_1}^t U^\nu, C_y = H_y^t g^{wsk}, C_2' = (Z_1^{wsk} \cdot Z_2^q)^{u_2}\}(m)$ to prove that all the sent values are well-formed;
4. Upload $report = (m, Z, C_{x_0}, C_{x_1}, C_y, C_V, \vec{C}', \sigma', \pi_2)$ to the smart contract via invoking uploadReport function.

• Reward: Once the authority receives a new reporting, he performs the following procedures to verify $report$ and create $reward$ for the valuable reporting.

1. Verify $\pi_2$;
2. Check the credential presentation by verifying the equation $Z = \dfrac{C_V}{W C_{x_0}^{x_0} C_{x_1}^{x_1} C_y^y}$;
3. Check the authenticity of the reported message $m$;
4. Check the token by invoking $\mathsf{SPS\text{-}EQ.Verify}(\vec{C}', \sigma', pk_{\mathsf{SPS\text{-}EQ}})$;
5. Set the reward value $v' \in \mathbb{Z}_p^*$, then compute $\sigma'' \leftarrow \mathsf{SPS\text{-}EQ.Sign}(sk_{\mathsf{SPS\text{-}EQ}}, (C_1', (C_1')^{z_3 \cdot v'} \cdot C_2'))$;
6. Upload $reward = (v', \sigma'')$ to the smart contract via invoking uploadReward function.

*Remark 2:* A reward is also a Pedersen commitment $\vec{C}' = (C_1', C_2') = (g^{u_2}, (Z_1^{wsk} Z_2^q Z_3^{v+v'})^{u_2})$ and an SPS-EQ signature $\sigma''$ on $C'$, where $v'$ is the reward value of this reporting.

• Gain: When the whistleblower obtains the reward $(v', \sigma'')$ from smart contract, he performs the following procedures to update his token.

1. Compute $(\vec{C}^*, \sigma^*) \leftarrow \mathsf{SPS\text{-}EQ.ChgRep}((C'_1, Z_3^{u_2 \cdot v'} \cdot C'_2), \sigma'', {u_2}^{-1}, pk_{\mathsf{SPS\text{-}EQ}})$;
2. Update the token $tok \leftarrow (\sigma^*, \vec{C}^*, q, v^* \leftarrow v + v')$.

*5.4. System Analysis*

In this section, we explain how the PriRPT scheme satisfies all the security requirements as mentioned in Section 3.5.

**Integrity for Reports**. Due to the immutability of blockchain, the records of reporting and rewarding are not easy to be tampered with or deleted, hence providing integrity of reports and reliable auditing.

**Unforgeability of Credentials**. Since the credentials offered by the KVAC scheme used in the system cannot be forged, only the whistleblower with the private key and the corresponding credentials can compute the credential representation. If any adversary tries to perform authentication by pretending to be a real whistleblower, he needs to break the unforgeability of the KVAC scheme.

**Unforgeability of Tokens**. The token in the system consists of a Pedersen commitment and the SPS-EQ signature; since the commitment is binding and the SPS-EQ signature is unforgeable, only the authority with the private key can create tokens. If any adversary tries to obtain rewards by pretending to be a real whistleblower, he needs to break the binding ability of Pedersen commitment and the unforgeability of the SPS-EQ scheme.

**Anonymity of Reporting**. As proved in *Theorem 1* of Sec. 6, the anonymity of the KVAC scheme, the signature adaption of the SPS-EQ scheme, the hiding of Pedersen commitment, and the simulatability of ZK-SoK can prevent the whistleblowers' real identities from being revealed.

**Anonymity of Rewarding**. The token in the system consists of a Pedersen commitment and an SPS-EQ signature. Since the commitment provides hiding and the SPS-EQ scheme perfectly adapts signatures, no adversary can identify the real identity of the whistleblower from the rewarding.

## 6. Security Analysis

In this section, we define the security model for PriRPT based on the game executed between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$. The security definition uses the following global oracles. $\mathcal{C}$ controls all the oracles, and $\mathcal{A}$ interacts with $\mathcal{C}$ by querying them.

(1) *Setup-Oracle*: The $\mathcal{C}$ runs Setup to generate public parameters $pp$, and KeyGen to obtain the authority's private/public key pair $(ask, apk)$. Then $\mathcal{C}$ sends $(pp, apk)$ to $\mathcal{A}$. In addition, the $\mathcal{C}$ performs registration protocol for $n$ whistleblowers by running KeyGen and WbReg algorithms, where $n$ is the number of honest whistleblowers in the game.

(2) *Reg-Oracle*: The $\mathcal{A}$ can choose any whistleblower to register. Specifically, $\mathcal{A}$ prepares a fresh private/public key pair $(wsk, wpk)$ and sends the registration request $(wpk, C_1^{req}, C_2^{req}, \pi_1)$ to $\mathcal{C}$. After receiving this request, $\mathcal{C}$ verifies it and returns a credential $(\nu, U, V)$ and an SPS-EQ signature $\sigma'$ to the whistleblower.

(3) *Report-Oracle*: The $\mathcal{A}$ can query any reporting of the honest whistleblower. In this oracle, $\mathcal{C}$ servers as a whistleblower, and replies a new report by running Report algorithm.

(4) *Reward-Oracle*: For any *report* generated by the registered whistleblower, the $\mathcal{C}$ replies a *reward* for it by running Reward algorithm.

(5) *Corrupt-Oracle*: The $\mathcal{A}$ can corrupt any honest whistleblower $i \in [1, n]$, where $i$ is the index of the whistleblower. Specifically, the $\mathcal{C}$ discloses the keys $(wsk_i, wpk_i)$, credential $cred_i$ and token $tok_i$ to $\mathcal{A}$.

(6) *Test-Oracle*: After $\mathcal{A}$ has queried the above oracles sufficient times, it queries this oracle. Specifically, $\mathcal{A}$ adaptively chooses two uncorrupted whistleblowers $i_0$ and $i_1$, that is, $\mathcal{A}$ has never queried them in *Corrupt-Oracle*. Then, $\mathcal{C}$ randomly chooses $b \in \{0, 1\}$, and compute a new report $report_b$ of the whistleblower $i_b$ by running Report algorithm. After obtaining $report_b$, $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. If $b = b'$, then $\mathcal{A}$ wins this game; otherwise, it fails.

Let's define the anonymity of PriRPT using the above game.

***Definition 1: Anonymity***. A PriRPT scheme is anonymous, if for any PPT adversary $\mathcal{A}$, there is a negligible function $\epsilon(\lambda)$ such that:

$$Adv^{anon} = \left| \Pr(b = b') - \frac{1}{2} \right| \leqslant \epsilon(\lambda)$$

***Theorem 1*:** Our PriRPT scheme is anonymous if the underlying KVAC scheme satisfies anonymity, SPS-EQ scheme provides perfect adaption, Pedersen commitment is perfectly hiding, and ZKSoK satisfies simulatability.

**Proof**: As shown in Report algorithm, a report is consisted of a credential presentation of KVAC $(Z, C_{x_0}, C_{x_1}, C_y, C_V)$, a Pedersen commitment $\vec{C}'$, a

randomization signature of SPS-EQ $\sigma'$, and a proof of ZKSoK $\pi_2$. Since Pedersen commitment is perfectly hiding and ZKSoK satisfies simulatability, the probability that any PPT adversary $\mathcal{A}$ wins the game by commitment $\vec{C}'$ and ZKSoK $\pi_2$ is negligible. Here, we prove that it is impossible to win the game with the credential presentation or the randomization signatures by defining two types of adversaries.

Type-1: $\mathcal{A}$ wins the anonymity game by the credential presentation of KVAC.

Type-2: $\mathcal{A}$ wins the anonymity game by the randomization signature of SPS-EQ.

***Lemma 1*:** If there is a Type-1 adversary $\mathcal{A}$ that breaks the anonymity of PriRPT scheme with the probability $\epsilon$, then there is another adversary $\mathcal{B}$ that breaks the anonymity of KVAC scheme with the same probability.

**Proof**: Given parameters of the KVAC scheme $pp_{\mathsf{KVAC}}$, an oracle of the credential presentation $\mathcal{O}_{\mathsf{KVAC}}(M, \cdot)$, a public key of the KVAC scheme $pk_{\mathsf{KVAC}}$, and a test oracle $\mathcal{O}_{\mathsf{Test}}(cred_0, cred_1)$, then $\mathcal{B}$ answers oracles queries as follows:

(1) *Setup-Oracle*: $\mathcal{B}$ runs $\mathsf{Setup}$ to generate public parameters $pp$, and runs $\mathsf{SPS\text{-}EQ.KeyGen}$ to obtain $(sk_{\mathsf{SPS\text{-}EQ}}, pk_{\mathsf{SPS\text{-}EQ}})$. Then, $\mathcal{B}$ sets $apk = (pk_{\mathsf{KVAC}}, pk_{\mathsf{SPS\text{-}EQ}})$ and sends $(pp, apk)$ to $\mathcal{A}$.

(2) *Reg-Oracle*: For any whistleblower with a registration request $(wpk, C_1^{req}, C_2^{req}, \pi_1)$, $\mathcal{B}$ queries $\mathcal{O}_{\mathsf{KVAC}}(wpk, \cdot)$ to obtain $cred$. Then $\mathcal{B}$ runs $\mathsf{SPS\text{-}EQ.Sign}$ to generate $\sigma'$ and replies the credential $cred$ and signature $\sigma'$ to the whistleblower.

(3) *Report-Oracle*, *Reward-Oracle*, and *Corrupt-Oracle*: $\mathcal{B}$ knows $sk_{\mathsf{SPS\text{-}EQ}}$ and the honest whistleblowers' secret keys, and so perfectly simulates these oracles.

(4) *Test-Oracle*: When $\mathcal{A}$ challenges two uncorrupted whistleblowers $i_0$ and $i_1$, $\mathcal{B}$ queries $(cred_{i_0}, cred_{i_1})$ to $\mathcal{O}_{\mathsf{Test}}$ and will get a response $pst_b$. Then $\mathcal{B}$ computes $(\vec{C}', \sigma')$ and simulates $\pi_2$. Finally, $\mathcal{B}$ sends $report = (m, pst_b, \vec{C}', \sigma', \pi_2)$ to obtain the guess $b'$ from $\mathcal{A}$ and uses the $b'$ as its answer.

The game is simulated perfectly and never aborts. Therefore, the advantage of $\mathcal{B}$ breaking the anonymity of KVAC is equal to $Adv^{anon}$. □

***Lemma 2*:** If there is a Type-2 adversary $\mathcal{A}$ that breaks the anonymity of PriRPT scheme with the probability $\epsilon$, then there is another adversary

$\mathcal{B}$ that breaks the signature adaptation of SPS-EQ scheme with the same probability.

**Proof**: Given parameters of the SPS-EQ scheme $pp_{\mathsf{SPS\text{-}EQ}}$, an oracle of the signature $\mathcal{O}_{\mathsf{SPS\text{-}EQ}}(\cdot, \vec{M})$, a public key of the SPS-EQ scheme $pk_{\mathsf{SPS\text{-}EQ}}$, and a test oracle $\mathcal{O}_{\mathsf{Test}}(\sigma_0, \sigma_1)$, then $\mathcal{B}$ answers oracles queries as follows:

(1) *Setup-Oracle*: $\mathcal{B}$ runs $\mathsf{Setup}$ to generate public parameters $pp$, and runs $\mathsf{KVAC.KeyGen}$ to obtain ($sk_{\mathsf{KVAC}}$, $pk_{\mathsf{KVAC}}$). Then, $\mathcal{B}$ sets $apk = (pk_{\mathsf{KVAC}}, pk_{\mathsf{SPS\text{-}EQ}})$ and sends $(pp, apk)$ to $\mathcal{A}$.

(2) *Reg-Oracle*: For any whistleblower with a registration request $(wpk, C_1^{req}, C_2^{req}, \pi_1)$, $\mathcal{B}$ queries $\mathcal{O}_{\mathsf{SPS\text{-}EQ}}(\cdot, (C_1^{req}, C_2^{req}))$ to obtain $\sigma'$. Then $\mathcal{B}$ runs $\mathsf{KVAC.Issue}$ to generate *cred* and replies the signature $\sigma'$ and credential *cred* to the whistleblower.

(3) *Reward-Oracle*: For a $report = (m, Z, C_{x_0}, C_{x_1}, C_y, C_V, \vec{C'}, \sigma', \pi_2)$ generated by the registered whistleblower, the $\mathcal{B}$ sets the reward value $v' \in \mathbb{Z}_p^*$ and queries $\mathcal{O}_{\mathsf{SPS\text{-}EQ}}(\cdot, (C_1', (C_1')^{z_3 \cdot v'} \cdot C_2'))$ to obtain $\sigma''$. Finally, $\mathcal{B}$ performs the remaining operations and replies *reward* for $\mathcal{A}$.

(4) *Report-Oracle* and *Corrupt-Oracle*: $\mathcal{B}$ knows $sk_{\mathsf{KVAC}}$ and the honest whistleblowers' secret keys and so perfectly simulates these oracles.

(5) *Test-Oracle*: When $\mathcal{A}$ challenges two uncorrupted whistleblowers $i_0$ and $i_1$, $\mathcal{B}$ queries $(\sigma_{i_0}, \vec{C}_{i_0}, \sigma_{i_1}, \vec{C}_{i_1})$ to $\mathcal{O}_{\mathsf{Test}}$ and will get a response $(\vec{C}_b', \sigma_b')$. Then $\mathcal{B}$ computes $(\nu, U, V)$ and simulates $\pi_2$. $\mathcal{B}$ sends $report = (m, Z, C_{x_0}, C_{x_1}, C_y, C_V, \vec{C}_b', \sigma_b', \pi_2)$ to obtain the guess $b'$ from $\mathcal{A}$. Finally, $\mathcal{B}$ uses the $b'$ as its answer.

The game is simulated perfectly and never aborts. Therefore, the advantage of $\mathcal{B}$ breaking the signature adaptation of SPS-EQ is equal to $Adv^{anon}$. □

## 7. Implementation and Evaluations

### *7.1. Theoretical Analysis and Comparison*

In Table 4 and Table 5, our scheme is compared with the existing schemes [3, 4] that support reporting and rewarding in terms of computation overheads and communication overheads. Due to the schemes in [5, 18, 22, 23] not supporting anonymous rewarding, it is not included in the comparison. We denote the exponentiation in $\mathbb{G}_i$ by $\mathsf{e}_i$ ($i \in \{1, 2\}$) and denote the bilinear map in pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ by $\mathsf{e}_p$, and $|\mathbb{Z}_p|, |\mathbb{G}_1|, |\mathbb{G}_2|, |\mathbb{G}_T|$ are the element's sizes in the group $\mathbb{Z}_p, \mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$, respectively.

Table 4: Computation comparison of our scheme with existing reporting constructions

| Scheme | Reporting | Rewarding |
|---|---|---|
| [3] | $(n+3)\mathsf{e}_1$ | $(2n+3)\mathsf{e}_1 + (n+1)\mathsf{e}_T + (2n+1)\mathsf{e}_p$ |
| [4] | $(3n+2)\mathsf{e}_1$ | $2n\mathsf{e}_1$ |
| PriRPT | $17\mathsf{e}_1 + 1\mathsf{e}_2$ | $20\mathsf{e}_1 + 1\mathsf{e}_2 + 5\mathsf{e}_p$ |

$n$ is the number of input public keys in the group of ring signature

Table 5: Communication comparison of our scheme with existing reporting constructions

| Scheme | Reporting | Rewarding |
|---|---|---|
| [3] | $(2n+3)\lvert\mathbb{G}_1\rvert$ | $10\lvert\mathbb{G}_1\rvert + 2\lvert\mathbb{G}_2\rvert + 28\lvert\mathbb{Z}_p\rvert$ |
| [4] | $(n+2)\lvert\mathbb{G}_1\rvert + 3\lvert\mathbb{Z}_p\rvert$ | $1\lvert\mathbb{G}_1\rvert + 6\lvert\mathbb{Z}_p\rvert$ |
| PriRPT | $9\lvert\mathbb{G}_1\rvert + 1\lvert\mathbb{G}_2\rvert + 8\lvert\mathbb{Z}_p\rvert$ | $2\lvert\mathbb{G}_1\rvert + 1\lvert\mathbb{G}_2\rvert + 1\lvert\mathbb{Z}_p\rvert$ |

$n$ is the number of input public keys in the group of ring signature

In the Report algorithm, to generate a credential presentation, randomize the token, and generate the ZKSoK on the reported message, the whistleblower's computational overhead is about $17\mathsf{e}_1 + 1\mathsf{e}_2$, and the communication overhead is about $9|\mathbb{G}_1| + 1|\mathbb{G}_2| + 8|\mathbb{Z}_p|$. In the Reward algorithm, to verify credential presentation and generate an SPS-EQ signature on a new reward, the authority's computational overhead is about $20\mathsf{e}_1 + 1\mathsf{e}_2 + 5\mathsf{e}_p$, and the communication overhead is about $2|\mathbb{G}_1| + 1|\mathbb{G}_2| + 1|\mathbb{Z}_p|$. Since the reporting schemes in [3, 4] use the ring signature [12] for anonymous reporting, the computational complexity of their schemes is $O(n)$. In particular, the computation of the Reward algorithm in [3] is about $(2n+3)\mathsf{e}_1 + (n+1)\mathsf{e}_T + (2n+1)\mathsf{e}_p$, and that of the Report algorithm in [4] is about $(3n+2)\mathsf{e}_1$, which makes it difficult to use the two schemes in practical applications. We see that PriRPT significantly reduces computational and communication costs.

In Table 6, our scheme is compared with the latest blockchain-based anonymous authentication schemes [29, 25] in terms of computational overheads of authentication protocols. Since the schemes in [27, 28] and [30] rely on the inherent properties of the blockchain and PUFs to achieve anonymous authentication, they are not included in the comparison. As shown in Table 6, only our scheme achieves the constant size computation overhead.

Table 6: Comparison of our scheme with existing blockchain-based authentication constructions

| Scheme | Computation overhead of authentication |
|---|---|
| [29] | $(4n+1)\mathsf{e}_1$ |
| [25] | $2n\mathsf{e}_1$ |
| PriRPT | $18\mathsf{e}_1$ |

$n$ is the number of public keys or credentials

### *7.2. Experimental Analysis*

In this section, we evaluate the performance of smart contract invocation and local computation, respectively. The overheads of invocation algorithms of the smart contract are transaction confirmation and consensus protocol reaching, and the overheads of local computation are computing operations in the bilinear group.

#### *7.2.1. Evaluations of Smart Contract*

Inspired by [47], we implement and evaluate the smart contract described in Algorithm 1 on JUCE [6] to verify our architecture. JUICE is an open blockchain service platform with homomorphic encryption, zero-knowledge proof, and other privacy protection features integrated with rich service interfaces. It contains efficient and reliable plug-in consensus mechanisms PBFT and RAFT, supports the design of smart contracts using *Solidity*, and provides contract templates and functional components.

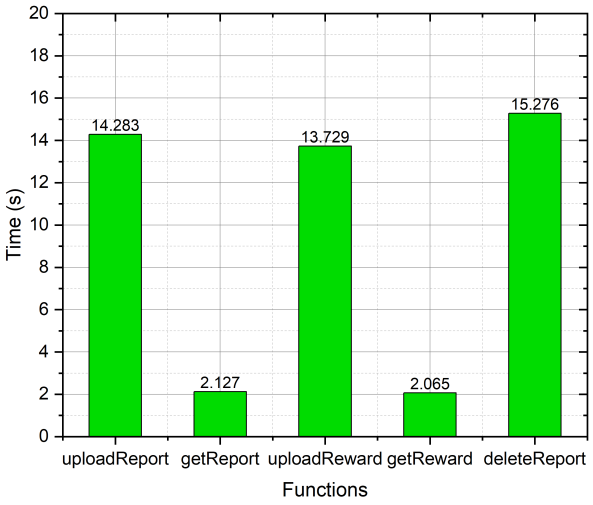



Figure 3: System latency of functions in the smart contract

Specifically, we first use JIDE, an online development tool of matrix elements, to develop the smart contract and deploy it on the JUCE blockchain with four master nodes. Then, we used Web3j [48] to test the functionality of the smart contract, including uploadReport, getReport, uploadReward, getReward, and deleteReport functions.

The results are shown in Fig. 3. Note that the system latency of a smart contract call is on the order of seconds, and this delay is acceptable for the reporting system. uploadReport, uploadReward, and deleteReport will change the internal variables of the smart contract, so they need to consume *gas* and transaction confirmation, and their execution time is about 13 s - 16 s. getReport and getReward are *view* type functions, so their time consumption is mainly communication delay, and their execution time is about 2 s.

*7.2.2. Evaluation of Local Computation*

We implement the PriRPT scheme using MIRACL [49], Type-III pairing [41], and Barreto-Naehrig curve (BN-256) [50] and test the system's performance at AES-100 bit security level. We run our implementation on a personal laptop (HUAWEI Matebook 14) with an AMD Ryzen-5 4600H with Radeon Graphics 3.00 GHz CPU, 16GB RAM, 512GB SSD running Ubuntu

Kylin 16.04 Operating system.

The detailed parameters of the pairing and their values used in the experiment are shown in Table 7, where the values are expressed in hexadecimal. The parameters include elliptic curve equation $equ$, embedding number $k$, order $p$ of $\mathbb{G}_1$ and $\mathbb{G}_2$, cofactor $cof$, trace $tra$, generator $g$ of $\mathbb{G}_1$, and generator $\widetilde{g}$ of $\mathbb{G}_2$.

Table 7: Value of the pairing parameter in the experiment

| Parameter | Value (hexadecimal) |
|---|---|
| $equ$ | $y^2 = x^3 + 2$ |
| $k$ | $C$ |
| $p$ | 2523648240000001BA344D8000000007<br>FF9F800000000010A10000000000000D |
| $cof$ | 2523648240000001BA344D8000000008<br>C2A2800000000016AD00000000000019 |
| $tra$ | 6181800000000003060000000000000007 |
| $g$ | (1B370C02969140F4804BBC211C85D9BC<br>57BF40523F004C58140A33D7A07B0C08,<br>1F6C6191736503BFEAE3A400822B7052<br>04E4C891A60FB89F434DC7CE94C1CA7A) |
| $\widetilde{g}$ | ([1217487F6CDD75AC9A927D03E54D8C49<br>A9026898D23A3B10B33948CB312A3067,<br>187FBE43E5BE26D0708C4946395DE1D8<br>722924DBA90272F7F82D5E69F934DC68],<br>[1013E332300F04B1D1200174BC67725A<br>E7BB6D65BAB49B0D25A95C782CFE3FE9,<br>2437C7FE36BFBC05E60A83C4400C4A71<br>A34E670BA9F15406BF1E5328A776B41B]) |

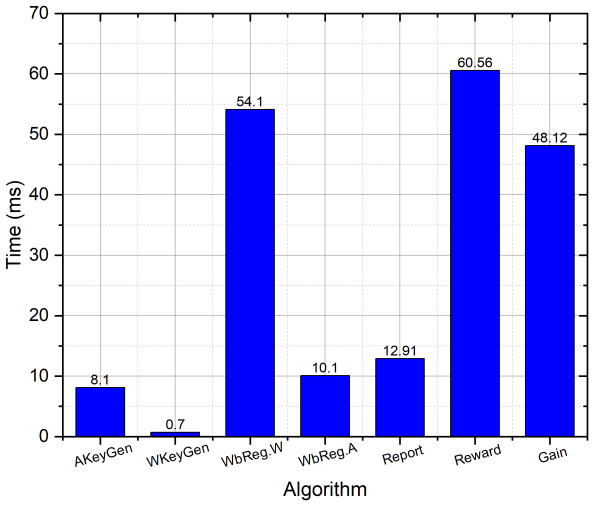



Figure 4: Execution Time of Algorithms

As shown in Fig. 4, we show the execution times of each algorithm of PriRPT, including KeyGen, WbReg, Report, Reward, and Gain. The authority and whistleblower cost 8.1 ms and 0.7 ms for the key generation, respectively. For the whistleblower registration, the whistleblower and authority cost 54.1 ms and 10.1 ms, respectively. The most frequently used in PriRPT are the Report, Reward, and Gain algorithms. When reporting a crime, the whistleblower costs 12.91 ms; when rewarding a report, the authority costs 60.56 ms; when updating the token, the whistleblower costs 48.12 ms.

In addition, to further demonstrate the performance of PriRPT (excluding invoking smart contracts and consensus algorithms), we compare PriRPT with the schemes in [3] and [4] in terms of computation and communication overheads, respectively. Since the schemes of [3] and [4] use the ring signatures as the main component, we set the number of public keys $n$ in the group from 10 to 40 during the evaluation process, respectively.

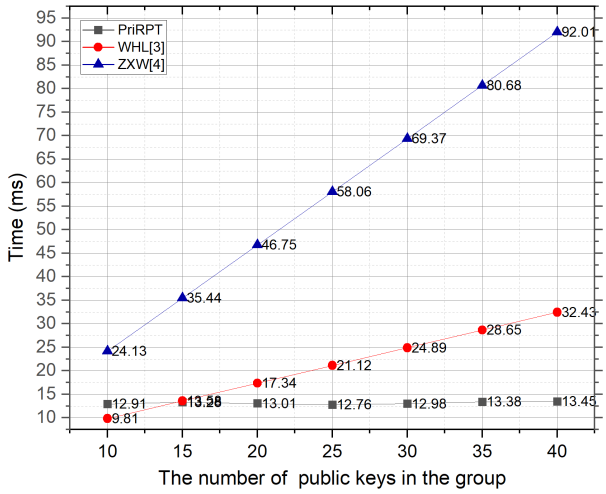



Figure 5: Computation cost of Reporting

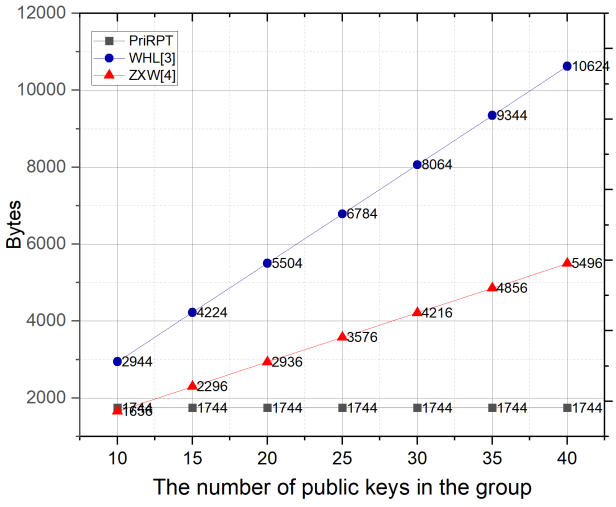



Figure 6: Communication cost of Reporting

In Fig. 5, we compare the computation cost of reporting. When $n$ equals 10, our Report algorithm takes only about 13 ms, which is 84% faster than the scheme in [4]. When $n$ is less than 15, the scheme's performance in [3] is better than ours, but the computational overhead of the scheme increases linearly with the increase of $n$. In Fig. 6, we compare the communication cost of reporting. Our Report algorithm achieves constant communication overhead, while the communication overhead of the schemes in [4] and [3] increases linearly with the increase of $n$.

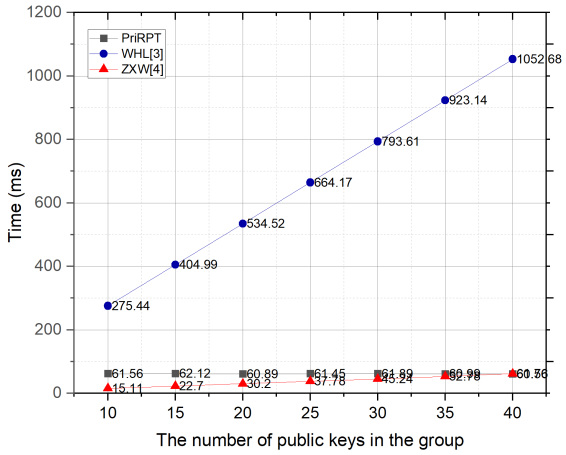



Figure 7: Computation cost of Rewarding

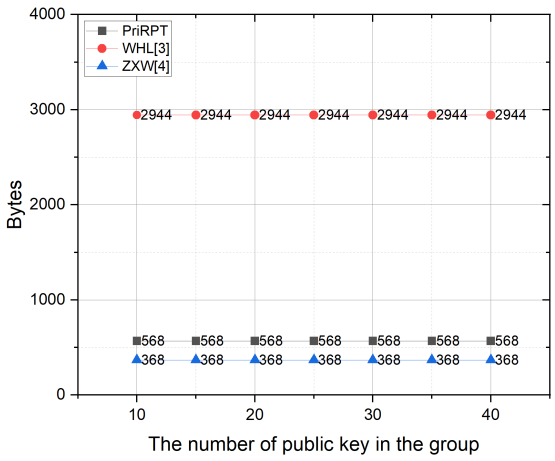



Figure 8: Communication cost of Rewarding

In Fig. 7, we compare the computation cost of rewarding. When $n$ equals 10, our Reward algorithm takes only 61 ms, which is 350% faster than the scheme in [3]. When $n$ is less than 40, the scheme in [4] is better than our scheme, but its running time increases linearly as $n$ increases, and there is no formal security proof for this scheme. In Fig. 8, we compare the communication cost of rewarding. Our scheme achieves constant complexity communication consumption as those in [3] and [4]. However, our scheme requires 568 bytes, comparable to the 368 bytes in [4], but 80% less than the 2944 bytes in [3].

The above analysis and comparison indicate that PriRPT enjoys significantly low communication and computation overheads compared to state-of-the-art schemes.

## 8. Conclusion

In this paper, we proposed a novel practical blockchain-based privacy-preserving reporting system with rewards and proved its security features, including integrity, unforgeability, the anonymity of reporting, and the anonymity of rewarding. Our scheme significantly improved the reporting and rewarding protocol efficiency using the KVAC and SPS-EQ schemes. We presented the

implementation of our scheme on PC and compared it with existing schemes, which shows that our scheme is well-suited for practical uses.

In the PriRPT system, we assume that the authority that issues credentials for whistleblowers and receives the reporting messages is the same entity, which is a reasonable assumption and allows the system to be designed with significantly lower computational and communication overhead. However, when reporting systems use the same credentials as other electronic systems, which is happening now that electronic identities are ubiquitous, PriRPT is no longer applicable. In the future, we will explore techniques for designing reporting systems that do not rely on KVAC schemes without this assumption.

## Acknowledgment

This work was supported in part by National Natural Science Foundation of China under Grant No.61872091; National Key Research and Development Program of China under Grant 2018YFB0803600; Beijing Natural Science Foundation under Grant No.4234084.

## References

[1] People daily, `https://wap.peopleapp.com/article/6170571/6074259.` (2021).

[2] J. Kivivuori, J. Savolainen, M. Aaltonen, The revenge motive in delinquency: Prevalence and predictors, Acta sociologica 59 (1) (2016) 69–84.

[3] H. Wang, D. He, Z. Liu, R. Guo, Blockchain-based anonymous reporting scheme with anonymous rewarding, IEEE Transactions on Engineering Management 67 (4) (2019) 1514–1524.

[4] S. Zou, J. Xi, S. Wang, Y. Lu, G. Xu, Reportcoin: A novel blockchain-based incentive anonymous reporting system, IEEE access 7 (2019) 65544–65559.

[5] H. Zou, X. Liu, W. Ren, T. Zhu, A decentralized electronic reporting scheme with privacy protection based on proxy signature and blockchain, Secur. Commun. Networks 2022 (2022) 5424395:1–5424395:8.

[6] Juice ltd., `https://open.juzix.net/.` (2023).

[7] M. Chase, S. Meiklejohn, G. Zaverucha, Algebraic macs and keyed-verification anonymous credentials, in: Proceedings of the 2014 acm sigsac conference on computer and communications security, 2014, pp. 1205–1216.

[8] M. Chase, T. Perrin, G. Zaverucha, The signal private group system and anonymous credentials supporting efficient verifiable encryption, in: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, 2020, pp. 1445–1459.

[9] C. Hanser, D. Slamanig, Structure-preserving signatures on equivalence classes and their application to anonymous credentials, in: Advances in Cryptology–ASIACRYPT 2014: 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, ROC, December 7-11, 2014. Proceedings, Part I 20, Springer, 2014, pp. 491–511.

[10] G. Fuchsbauer, C. Hanser, D. Slamanig, Structure-preserving signatures on equivalence classes and constant-size anonymous credentials, Journal of Cryptology 32 (2019) 498–546.

[11] T. P. Pedersen, Non-interactive and information-theoretic secure verifiable secret sharing, in: Advances in Cryptology—CRYPTO'91: Proceedings, Springer, 2001, pp. 129–140.

[12] A. Bender, J. Katz, R. Morselli, Ring signatures: Stronger definitions, and constructions without random oracles, in: Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3, Springer, 2006, pp. 60–79.

[13] S.-F. Sun, M. H. Au, J. K. Liu, T. H. Yuen, Ringct 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero, in: Computer Security–ESORICS 2017: 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II 22, Springer, 2017, pp. 456–474.

[14] C. Zhang, P.-H. Ho, J. Tapolcai, On batch verification with group testing for vehicular communications, Wireless Networks 17 (2011) 1851–1865.

[15] N. Saxena, H. Shen, N. Komninos, K.-K. R. Choo, N. S. Chaudhari, Bvpsms: A batch verification protocol for end-to-end secure sms for mobile users, IEEE Transactions on Dependable and secure Computing 17 (3) (2018) 550–565.

[16] M. Mambo, K. Usuda, E. Okamoto, Proxy signatures for delegating signing operation, in: Proceedings of the 3rd ACM Conference on Computer and Communications Security, 1996, pp. 48–57.

[17] M. Ajtai, C. Dwork, A public-key cryptosystem with worst-case/average-case equivalence, in: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, 1997, pp. 284–293.

[18] R. Shi, Y. Yang, H. Xie, H. Feng, G. Shi, J. Zhang, Prisign, a privacy-preserving single sign-on system for cloud environments, Applied Sciences 13 (2) (2023) 727.

[19] D. Pointcheval, O. Sanders, Short randomizable signatures, in: Topics in Cryptology-CT-RSA 2016: The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29-March 4, 2016, Proceedings, Springer, 2016, pp. 111–126.

[20] X. T. Do, D. H. Phan, D. Pointcheval, Traceable inner product functional encryption, in: Topics in Cryptology–CT-RSA 2020: The Cryptographers' Track at the RSA Conference 2020, San Francisco, CA, USA, February 24–28, 2020, Proceedings, Springer, 2020, pp. 564–585.

[21] J. Groth, Efficient fully structure-preserving signatures for large messages, in: Advances in Cryptology–ASIACRYPT 2015: 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29–December 3, 2015, Proceedings, Part I, Springer, 2016, pp. 239–259.

[22] J. Huang, L. Kong, L. Cheng, H.-N. Dai, M. Qiu, G. Chen, X. Liu, G. Huang, Blocksense: Towards trustworthy mobile crowdsensing via proof-of-data blockchain, IEEE Transactions on Mobile Computing (2022).

[23] Y. Wang, X. Wang, H.-N. Dai, X. Zhang, M. Imran, A data reporting protocol with revocable anonymous authentication for edge-assisted in-

telligent transport systems, IEEE Transactions on Industrial Informatics (2022).

[24] R. Jiang, R. Lu, J. Luo, C. Lai, X. Shen, Efficient self-healing group key management with dynamic revocation and collusion resistance for scada in smart grid, Security and communication networks 8 (6) (2015) 1026–1039.

[25] Z. Lu, Q. Wang, G. Qu, H. Zhang, Z. Liu, A blockchain-based privacy-preserving authentication scheme for vanets, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 27 (12) (2019) 2792–2801.

[26] Q. Fan, J. Chen, L. J. Deborah, M. Luo, A secure and efficient authentication and data sharing scheme for internet of things based on blockchain, Journal of Systems Architecture 117 (2021) 102112.

[27] C. Lin, D. He, X. Huang, N. Kumar, K.-K. R. Choo, Bcppa: A blockchain-based conditional privacy-preserving authentication protocol for vehicular ad hoc networks, IEEE Transactions on Intelligent Transportation Systems 22 (12) (2020) 7408–7420.

[28] D. Chhikara, S. Rana, A. Mishra, D. Mishra, Blockchain-driven authorized data access mechanism for digital healthcare, Journal of Systems Architecture 131 (2022) 102714.

[29] Q. Mei, H. Xiong, Y.-C. Chen, C.-M. Chen, Blockchain-enabled privacy-preserving authentication mechanism for transportation cps with cloud-edge computing, IEEE Transactions on Engineering Management (2022).

[30] D. Li, R. Chen, D. Liu, Y. Song, Y. Ren, Z. Guan, Y. Sun, J. Liu, Blockchain-based authentication for iiot devices with puf, Journal of Systems Architecture 130 (2022) 102638.

[31] D. Chaum, Security without identification: Transaction systems to make big brother obsolete, Communications of the ACM 28 (10) (1985) 1030–1044.

[32] S. Brands, Rethinking public key infrastructures and digital certificates: building in privacy, Mit Press, 2000.

[33] J. Blömer, J. Bobolz, D. Diemert, F. Eidens, Updatable anonymous credentials and applications to incentive systems, in: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019, pp. 1671–1685.

[34] A. Sonnino, M. Al-Bassam, S. Bano, S. Meiklejohn, G. Danezis, Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers, arXiv preprint arXiv:1802.07344 (2018).

[35] J. Blömer, J. Bobolz, Delegatable attribute-based anonymous credentials from dynamically malleable signatures, in: Applied Cryptography and Network Security: 16th International Conference, ACNS 2018, Leuven, Belgium, July 2-4, 2018, Proceedings, Springer, 2018, pp. 221–239.

[36] A. Barki, S. Brunet, N. Desmoulins, S. Gambs, S. Gharout, J. Traoré, Private ecash in practice (short paper), in: Financial Cryptography and Data Security: 20th International Conference, FC 2016, Christ Church, Barbados, February 22–26, 2016, Revised Selected Papers 20, Springer, 2017, pp. 99–109.

[37] G. Couteau, M. Reichle, Non-interactive keyed-verification anonymous credentials, in: Public-Key Cryptography–PKC 2019: 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, April 14-17, 2019, Proceedings, Part I, Springer, 2019, pp. 66–96.

[38] J. Camenisch, M. Drijvers, P. Dzurenda, J. Hajny, Fast keyed-verification anonymous credentials on standard smart cards, in: ICT Systems Security and Privacy Protection: 34th IFIP TC 11 International Conference, SEC 2019, Lisbon, Portugal, June 25-27, 2019, Proceedings 34, Springer, 2019, pp. 286–298.

[39] V. Shoup, Lower bounds for discrete logarithms and related problems, in: Advances in Cryptology—EUROCRYPT'97: International Conference on the Theory and Application of Cryptographic Techniques Konstanz, Germany, May 11–15, 1997 Proceedings 16, Springer, 1997, pp. 256–266.

[40] K. Wüst, A. Gervais, Do you need a blockchain?, in: 2018 crypto valley conference on blockchain technology (CVCBT), IEEE, 2018, pp. 45–54.

[41] S. D. Galbraith, K. G. Paterson, N. P. Smart, Pairings for cryptographers, Discrete Applied Mathematics 156 (16) (2008) 3113–3121.

[42] M. Chase, A. Lysyanskaya, On signatures of knowledge, in: Advances in Cryptology-CRYPTO 2006: 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006. Proceedings 26, Springer, 2006, pp. 78–96.

[43] A. Fiat, A. Shamir, How to prove yourself: Practical solutions to identification and signature problems., in: Crypto, Vol. 86, Springer, 1986, pp. 186–194.

[44] J. Camenisch, M. Stadler, Efficient group signature schemes for large groups, in: Advances in Cryptology—CRYPTO'97: 17th Annual International Cryptology Conference Santa Barbara, California, USA August 17–21, 1997 Proceedings 17, Springer, 1997, pp. 410–424.

[45] G. Wood, et al., Ethereum: A secure decentralised generalised transaction ledger, Ethereum project yellow paper 151 (2014) (2014) 1–32.

[46] C. Cachin, et al., Architecture of the hyperledger blockchain fabric, in: Workshop on distributed cryptocurrencies and consensus ledgers, Vol. 310, Chicago, IL, 2016, pp. 1–4.

[47] C. Lin, D. He, N. Kumar, X. Huang, P. Vijayakumar, K.-K. R. Choo, Homechain: A blockchain-based secure mutual authentication system for smart homes, IEEE Internet of Things Journal 7 (2) (2019) 818–829.

[48] Web3j, `https://docs.web3j.io/4.8.7/.` (2023).

[49] Miracl ltd., `https://github.com/miracl /MIRACL.` (2023).

[50] J. Fan, F. Vercauteren, I. Verbauwhede, Faster fp-arithmetic for cryptographic pairings on barreto-naehrig curves, in: CHES, Vol. 9, Springer, 2009, pp. 240–253.