Division of Regulatory Power: Collaborative Regulation for Privacy-Preserving Blockchains

Tianyu Zhaolu, Zhiguo Wan, Huaqun Wang,

***Abstract*—Recently, fast advances in decentralized blockchains have led to the rapid development of decentralized payment systems and finance. In decentralized anonymous payment systems such as Monero, Zerocash and Zether, payment amounts and traders' addresses are confidential to other users. Therefore, cryptocurrency may be used for illegal activities such as money laundering, bribery and blackmails. To solve this problem, some decentralized anonymous payment schemes supporting regulation have been proposed. Unfortunately, most solutions have no restriction on the regulator's power, which may lead to abuse of power and disclosure of privacy. In this paper, we propose a decentralized anonymous payment scheme supporting collaborative regulation. Different from existing solutions, our scheme prevents abuse of power by dividing the regulatory power into two regulatory authorities. These two regulatory authorities, namely Filter and Supervisor, can cooperate to recover payment amounts and traders' addresses from suspicious transactions. However, neither Filter nor Supervisor alone can decode transactions to obtain transaction privacy. Our scheme enjoys three major advantages over others: 1) We design a generic transaction structure using zk-SNARK, 2) divide regulatory power using the regulation label, 3) and achieve aggregability of transaction amounts using the amount label. The experimental result shows that the time cost of generating a transaction is about 11 s and the transaction fee is about 12,183k gas, which is acceptable.**

***Index Terms*—Blockchain, Cryptocurrency, Privacy Protection, Regulation, Decentralized Finance.**

I. Introduction

In recent years, the blockchain technology has brought significant economic and social impacts to the real world. With its intruding features of decentralization, transparency and immutability, blockchain has been applied in finance, IoT, reputation systems and social services [1]. The most popular application of blockchain is the decentralized payment system, also known as cryptocurrency. In 2021 the total volume of cryptocurrency transactions has grown to $\$15.8$ trillion.

The privacy problem has been a major issue with public blockchains due to their transparency. In decentralized payment systems such as Bitcoin [2] and Ethereum [3], the transactions are completely public. Users trade with each other with pseudonyms, and payment amounts of transactions is available to each user. Unfortunately, it is possible to establish a mapping between pseudonyms and real identities of users by analyzing the transactions. In other words, Bitcoin and Ethereum fail to support privacy protection of transactions compared with traditional payment systems that rely on trusted centers. In order to solve this privacy problem, decentralized anonymous payment systems (DAP) such as Monero [4], Zerocash [5] and Zether [6] have been proposed. Traders' addresses and the payment amount of each transaction are confidential to other users in DAP.

However, unconditional privacy may turn DAP into a hotbed of crimes. Cryptocurrency may be used for bribery, blackmail, terrorist financing and money laundering [7]. According to statistics, cyber criminals made use of cryptocurrency to laundered $\$8.6$ billion in 2021. The Draft European Parliament Legislative Resolution[1] in 2021 proposed to develop regulation and supervision of crypto-asset transactions in relation to money laundering and terrorist financing. Furthermore, it is mentioned in the executive order[2] issued by the U.S. government in March 2022 that digital assets may create additional economic and financial risks requiring a regulatory approach that adequately addresses those risks.

In order to crack down crimes on DAP, it is necessary to realize effective regulation for each user's total payment amount in a period rather than the amount of a single transaction. For example, to transfer $100,000$ tokens to Bob, Alice posts $10,000$ transactions on the blockchain and the payment amount in each transaction is 10 tokens. If the regulator only focuses on the transaction with a large amount, each transaction with a small amount between Alice and Bob will be ignored. If the regulation objective is Alice's total payment amount during a period, a large number of tokens transferred between Alice and Bob will attract the attention of the regulator.

Ideally, for accurate and effective regulation, it is required that: 1) the regulator should only know the total transaction amount of a user during a specific period; 2) only if the total transaction amount exceeds a threshold can the regulator to identify the real identity of the corresponding user; 3) at least

Corresponding authors: Huaqun Wang.

T. Zhaolu is with the School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (e-mail: zhaolty@aliyun.com).

Z. Wan is with the Zhejiang Lab, Hangzhou 310000, China (e-mail: wanzhiguo@zhejianglab.com).

H. Wang is with the Jiangsu Key Laboratory of Big Data Security & Intelligent Processing, College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China, and also with the Guangxi Key Laboratory of Cryptography and Information Security, Guilin 541004, China (e-mail: wanghuaqun@aliyun.com).

[1]https://www.europarl.europa.eu/doceo/document/CJ12-PR-704888_EN.pdf

[2]https://www.whitehouse.gov/briefing-room/presidential-actions/2022/03/09/

two regulators should collaborate to identify the real identity and the transaction amount.

Challenges. There are several challenges in constructing an effective regulatable decentralized anonymous payment scheme for blockchains.

First of all, it is not trivial to realize *restrictive regulation* for regulators, which requires to identify potentially illegal transactions without exposing additional transaction privacy. One possible strategy is to divide the regulatory power among multiple regulators, similar to the separation of powers in modern governments, and different regulators have different responsibilities. To hold illegal users accountable, multiple regulators must cooperate to obtain the sender's address and the payment amount of an illegal transaction.

Secondly, it is necessary to achieve *consistency* between regulatory information and transactions. To realize transaction regulation, an additional field with regulatory information (e.g. the payment amount) should be appended to an anonymous transaction. From the additional regulatory field the regulator can extract regulatory information. However, a malicious user may generate the regulatory field containing incorrect information to bypass regulation, from which the regulators cannot identify illegal transactions at all. To ensure consistency of regulatory information and transactions is critical for the security of a regulatory scheme, but it is rather difficult since both the regulatory field and the original transaction are anonymous.

Finally, it is desirable to achieve *aggregability* for transactions, which means the regulator can aggregable transactions from the same user to monitor his total payment amounts. Considering privacy protection and efficient regulation, the regulator cannot directly extract payment amounts from transactions, but aggregate transactions from the same user and determine whether his total payment amount exceeds the preset upper limit.

In this paper, we propose a decentralized anonymous payment scheme with collaborative regulation (DAPCR). The advantages of DAPCR are as follows.

- DAPCR is a highly generic solution to achieve the regulation of any DAP scheme. It is compatible not only with UTXO blockchains but also with account-based blockchains. To realize generality of DAPCR, we extend the original transaction in the DAP scheme using zk-SNARK. The regulator makes use of the additional field appended to the transaction for regulation, regardless of the structure of the original transaction.
- We design the relationship between the regulation label and the user's pseudonym to achieve restrictive regulation. The regulatory power is divided to two regulators called *Filter* and *Supervisor*. Filter can extract the regulation label from each transaction. The regulation label consists of a sender label and an amount label, and the sender label is equal to the sender's pseudonym. The mapping between pseudonyms and addresses is only known to Supervisor. These two regulators must cooperate to regulate transactions.
- We design the amount label to achieve the aggregability of transaction amounts. Filter can extract the amount label from each transaction and aggregates amount labels to determine whether the total amount exceeds the preset upper limit. During this process, no additional privacy is exposed.

TABLE I
PROPERTIES OF DAPCR AND RELATED WORKS

Scheme	Restricted Regulation	Non-interaction	Aggregability	Generality
DAPCR	✓	✓	✓	✓
[8]	✗	✓	✗	✗
[9]	✗	✓	✗	✗
[10]	✓	✓	✗	✗
[11]	✗	✗	✗	✗
[12]	✗	✗	✗	✗

To clarify the advantages of our scheme, the properties of DAPCR and related works are shown in Table I.

Paper Contributions. In summary, our contributions in this paper are as follows.

1. We propose the DAPCR scheme which can be applied to most decentralized anonymous payment scheme such as Zerocah or Zether. To our best knowledge, this is the first generic collaborative regulation scheme for decentralized anonymous payment schemes. As the building block of DAPCR, we design a regulatable commitment (RegCom), which is appended to each transaction for regulation. From regulatable commitments, Filter can screen out illegal transactions but no transaction privacy.
2. We provide formal security definitions of DAPCR and give the security proofs for it. We define two different adversaries: *Type-1 adversary* and *Type-2 adversatry* for DAPCR and prove that DAPCR satisfies the security against these adversaries.
3. We evaluate the performance of DAPCR on both local devices and the Ethereum testnet *Rinkeby*. The time cost of generating a transaction is about 11 s and the transaction fee is $12,183,490$ gas. For the regulator, the time cost of screening out illegal transactions from $10,000$ transactions is about 5.5×10^5 ms. These experimental results indicate the effectiveness of DAPCR.

Paper Outline. Section II reviews the recent literature related to the decentralized anonymous payment system. In Section III, we define the notations in this paper and review the background materials associated with our work. In Section IV, we present the system framework and security definitions of DAPCR. In Section V, we propose an efficient construction of the RegCom scheme, which is the building block of DAPCR. In Section VI and Section VII, we present an efficient and generic construction of DAPCR. Section VIII and Section IX present security analysis and performance analysis for DAPCR. Finally, Section X concludes our work.

II. REALTED WORK

In this section, a non-exhaustive review of related works is presented.

Decentralized Payment System. In a decentralized payment system such as Bitcoin [2] or Ethereum [3], users

can trade with each other using pseudonyms instead of their identities in the real world. The pseudonym achieves weak privacy protection for the user's identity, which is normally considered non-confidential against transaction analysis [13], network traffic analysis [14] and ownership mappings [15]. Moreover, the payment amount of each transaction is publicly available. Dash [16] only achieves privacy protection for the user's identity but the payment amount is still public.

Decentralized Anonymous Payment System. Considering the lack of privacy protection in decentralized payment systems, some solutions have been proposed. Zerocash [5] is a decentralized anonymous payment scheme based on Bitcoin, which achieves privacy protection of traders' addresses and payment amounts. To spend a zerocoin, the relationship between the owner's private key and the commitment to the zerocoin should be publicly proved with zero-knowledge proof. Miners only verify the validity of a transaction but cannot extract any other information from the transaction. Zether [6] is a fully decentralized confidential payment mechanism based on Ethereum, which also guarantees the confidentiality of payment amounts and traders' addresses with zero-knowledge proof. In Zether, the user's address is hidden in the anonymous set. BlockMaze [17] is a privacy-preserving account-based blockchain. BlockMaze constructs a dual-balance model and a two-step fund transfer procedure for the blockchain using zk-SNARK.

Regulation for DAP. The decentralized anonymous payment scheme achieves strong privacy protection but also gets into a delimma: Crimes cannot be detected by the regulator in DAP. To eliminate concerns about potential crimes, some solutions have been proposed recently.

Wang et al. [8] propose a decentralized anonymous payment scheme with supervision (DAPS) based on zk-SNARK and the elliptic curve cryptography. A transaction in DAPS contains a ciphertext encrypted with the public key of the regulator. The regulator can decrypt ciphertexts in transactions with the private key to obtain the privacy. Faced with a large number of transactions, it is inefficient for the regulator needs to decrypt the ciphertext in each transaction. Lin et al. [9] present a secure and efficient decentralized conditional anonymous payment system (DCAP) based on signatures of knowledge. A transaction in DCAP contains anonymous addresses of the sender and the recipient. The regulator can trace the long-term address for an anonymous address, but the payment amount of each transaction is public in DCAP. Wang et al. [8] and Lin et al. [9] neglect to restrict the regulatory power, which may be abused.

Some solutions recognize the importance of restricted regulation, but still have some shortcomings. Garman et al. [10] design a DAP scheme based on Zerocash that forces users to comply with specific policies and grants regulators the power of coin tracing and user tracing. [10] restricts the power of the regulator who is asked to provide an accountable record of the power being used. However, the regulator can still obtain privacy from a transaction. PRCash [18] is a new blockchain currency with privacy protection and regulation, in which the sender's identity is also encrypted with the regulator's public key and included in the transaction. UTT [19] is a decentralized ecash system with accountable privacy. In UTT each user needs to get *budget coins*, which are used to limit the total sum of payments, from the auditor per month.

zkLedger [12] and miniLedger [11] are two decentralized payment systems that achieve privacy protection and verifiability auditing. These solutions realize rich auditing functions, but require auditors to interact with users. However, users may not always be online and a malicious user may ignore the auditor's queries, which leads to delays in the audit.

Platypus [20] and PEReDi [21] are two central bank digital currencies with privacy protection and regulation. In Platypus each user needs to encrypt his privacy with the regulator's public key and the ciphertext is included in the transaction. The regulator can decrypt the ciphertext for transaction privacy. In PEReDi several authorities form a committee to revoke privacy or trace transactions from some user. The committee revokes the privacy of a transaction by decrypting the ciphertext saved in the ledger. It is inefficient to decrypt the ciphertext of each transaction for its privacy when auditing a large number of transactions.

TABLE II
NOTATION

Notation	Description
$\mathcal{F}$	Filter
$\mathcal{S}$	Supervisor
$\mathcal{U}$	User
$\mathcal{I}$	Initialization Module
$addr$	the address of $\mathcal{U}$
max	the upper limit of total payment volume
nym	the pseudonym of $\mathcal{U}$
$\mathcal{U}_{nym}$	the user with the pseudonym nym
$\mathbf{tx}$	the transaction in DAP
$\mathbf{rtx}$	the regulatable transaction in DAPCR
$\vec{\mathcal{V}_{tp}}$	the vector of transaction privacy
com_R	the regulatable commitment to $\vec{\mathcal{V}_{tp}}$
π_C	the consistency proof
lab	the regulation-label extracted from com_R
lab_S	the sender label in lab
lab_v	the amount label in lab
$\otimes$	an operator of vectors in Section III-D

III. PRELIMINARIES

A. Notation

The notations used in the paper and their descriptions are shown in Table II.

B. Decentralized Anonymous Payments

A DAP scheme such as Zerocash [5], Monero [4] or Zether [6] can be highly simplified into four algorithms as follows.

- $\mathsf{DAP.Setup}(1^\lambda) \rightarrow pp_{tx}$. This algorithm takes a security parameter λ as input and outputs a public parameter pp_{tx}.
- $\mathsf{DAP.GenAddr}(pp_{tx}) \rightarrow (addr, sk)$. This algorithm takes the public parameter pp_{tx} as input and outputs an address $addr$ and a private key sk.
- $\mathsf{DAP.GenTx}(addr_S, addr_R, v, sk, pub, pri, pp_{tx}) \rightarrow \mathbf{tx}$. This algorithm takes a sender's address $addr_S$, a recipient's address $addr_R$, the payment amount v, a private

key sk, pub, pri and pp_{tx} as input and outputs a transaction $\mathbf{tx}$. pub represents public inputs besides pp_{tx}. pri represents private inputs besides $addr_S$, $addr_R$, v and sk.
- $\mathsf{DAP.VerifyTx}(\mathbf{tx}, pp_{tx}) \rightarrow 0/1$. This algorithm takes a transaction $\mathbf{tx}$ and the public parameter pp_{tx} as input and outputs 1 if $\mathbf{tx}$ is valid or 0 otherwise.

Although security properties of DAP schemes are different from each other, most DAP schemes satisfy *Transaction Indistinguishability*.

Transaction Indistinguishability. This property means that no information of the transaction is leaked to other users. For any probabilistic polynomial-time algorithm $\mathcal{D}$ and a simulator $\mathcal{E}$,

$$|Pr[\mathcal{D}(\mathbf{tx}, pp_{tx}) = 1] - Pr[\mathcal{D}(\mathbf{tx}', pp_{tx}) = 1]| \leq \mathsf{negl}(\lambda)$$

where $\mathbf{tx} \leftarrow \mathsf{DAP.GenTx}(addr_S, addr_R, v, sk, pub, pri, pp_{tx})$ and $\mathbf{tx}' \leftarrow \mathcal{E}(pp_{tx})$.

C. zk-SNARK

We first present a binary relation $R = \{(\phi, \varpi) | f(\phi, \varpi) = true\}$, which constructed by an arithmetic circuit C. For any pair $(\phi, \varpi) \in R$, ϕ and ϖ are considered as a statement and a witness. Zero-knowledge succinct non-interactive argument of knowledge, also known as zk-SNARK, consists of three algorithms as follows [5], [22].

- $\mathsf{ZK.KeyGen}(1^\lambda, C) \rightarrow (pk_{zk}, vk_{zk})$. The algorithm takes a security parameter λ and an arithmetic circuit C as input, and outputs a proving key pk_{zk} and a verification key pk_{zk}.
- $\mathsf{ZK.Prove}(\phi, \varpi, pk_{zk}) \rightarrow \pi$. The prover algorithm takes a statement ϕ, a witness ϖ and the proving key pk_{zk} as input, and outputs a proof π.
- $\mathsf{ZK.Verify}(\phi, \pi, vk_{zk}) \rightarrow 0/1$. The algorithm takes a statement ϕ, a proof π and the verification key vk_{zk} as input, and outputs 1 if the proof π is valid or 0 otherwise.

A secure zk-SNARK scheme satisfies the following properties [22]–[24].

1. *Succinctness*. The runtime of $\mathsf{ZK.Verify}$ is polynomial in $|a|+\lambda$ and the size of π output by $\mathsf{ZK.Prove}$ is polynomial in λ.
2. *Completeness*. For any $(\phi, \varpi) \in R$,
$$Pr[\mathsf{ZK.Verify}(\phi, \pi, vk_{zk}) \rightarrow 1] = 1$$
in which $(pk_{zk}, vk_{zk}) \leftarrow \mathsf{ZK.KeyGen}(1^\lambda, C)$ and $\pi \leftarrow \mathsf{ZK.Prove}(\phi, \varpi, pk_{zk})$.
3. *Zero-knowledge*. This property means that no information other than the truth of the statement is leaked. For any $(\phi, \varpi) \in R$, any probabilistic polynomial-time adversary $\mathcal{A}$ and a polynomial-time simulator Sim,
$$Pr\left[\mathcal{A}(\pi, pk_{zk}, vk_{zk}) = 1 \,\middle|\, \begin{array}{c}\mathsf{ZK.KeyGen}(1^\lambda, C) \\ \rightarrow (pk_{zk}, vk_{zk}) \\ \mathsf{ZK.Prove}(\phi, \varpi, pk_{zk}) \rightarrow \pi\end{array}\right] -$$
$$Pr\left[\mathcal{A}(\pi', pk_{zk}, vk_{zk}) = 1 \,\middle|\, \begin{array}{c}\mathsf{Sim}(1^\lambda, C) \\ \rightarrow (pk_{zk}, vk_{zk}, \tau) \\ \mathsf{Sim}(\phi, pk_{zk}, \tau) \rightarrow \pi'\end{array}\right]$$
$\leqslant \mathsf{negl}(\lambda)$.
4. *Knowledge Soundness*. This property means that a secure zk-SNARK cannot prove a false statement. For any probabilistic polynomial-time adversary $\mathcal{A}$ and a probabilistic polynomial-time extractor $\mathcal{E}$,
$$Pr\left[\begin{array}{c}\mathsf{ZK.Verify}(\phi, \pi, vk_{zk}) \\ \rightarrow 1 \text{ for } (\phi, \varpi) \notin R\end{array} \,\middle|\, \begin{array}{c}\mathsf{ZK.KeyGen}(1^\lambda, C) \\ \rightarrow (pk_{zk}, vk_{zk}) \\ \mathcal{A}(pk_{zk}, vk_{zk}) \rightarrow (\phi, \pi) \\ \mathcal{E}(pk_{zk}, vk_{zk}) \rightarrow \varpi\end{array}\right]$$
$\leq \mathsf{negl}(\lambda)$.

D. Bilinear Pairings

The pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$ is a bilinear pairing if it has three properties as follows [25].

1. *Bilinear*. For $g_1, g_1' \in \mathbb{G}_1$ and $g_2, g_2' \in \mathbb{G}_2$, $e(g_1 \cdot g_1', g_2) = e(g_1, g_2) \cdot e(g_1', g_2)$ and $e(g_1, g_2 \cdot g_2') = e(g_1, g_2) \cdot e(g_1, g_2')$.
2. *Non-degenerate*. If g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$, $g_3 = e(g_1, g_2)$ is a generator of $\mathbb{G}_3$.
3. *Computable*. The map e is efficiently computable.

$\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_3$ are cyclic multiplicative groups of the prime order q.

Discrete Logarithm Problem. For a tuple (g, g^a) where $g \in \mathbb{G}_1$ and $a \in \mathbb{Z}_q^*$, the discrete logarithm (DL) problem is to compute a. For any probabilistic polynomial-time adversary $\mathcal{A}$, the advantage in solving the DL problem is negligible.

Decisional Diffie-Hellman Problem. For a problem instance (g, g^a, g^b, Z) where $g \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$, the decisional Diffie-Hellman (DDH) problem is to determine whether $Z = g^{ab}$. If e is a Type III bilinear pairing, the advantage of any probabilistic polynomial-time adversary $\mathcal{A}$ in solving the DDH problem is negligible [26].

Variant Decisional Diffie-Hellman Problem. For a tuple (g, g^a, g^b, g^{ac}, T) where $g \in \mathbb{G}_1$ and $a, b, c \in \mathbb{Z}_q^*$, the variant DDH problem is to determine whether $T = g^{bc}$. For any probabilistic polynomial-time adversary $\mathcal{A}$, the advantage in solving the variant DDH problem is negligible [27].

Operator of Vectors. For a vector $(k_1, k_2, ..., k_n)$ where $k_i \in \mathbb{G}_1$ and another vector $(t_1, t_2, ..., t_n)$ where $t_i \in \mathbb{Z}_q^*$, we define an operator $\otimes$ of vectors:

$$(k_1, k_2, ..., k_n) \otimes (t_1, t_2, ..., t_n) = \prod_{1 \leq i \leq n} k_i^{t_i}$$

E. Hash Function

The hash function contained in DAPCR takes a bit string with arbitrary length as input and outputs an element in $\mathbb{Z}_q^*$, which can be written as $\mathsf{Hash} : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$. A secure hash function satisfies *collision resistance*, *preimage resistance* and *second preimage resistance* [28].

IV. System framework and Security definitions

In this section, the system framework and security definitions of DAPCR are going to be introduced.

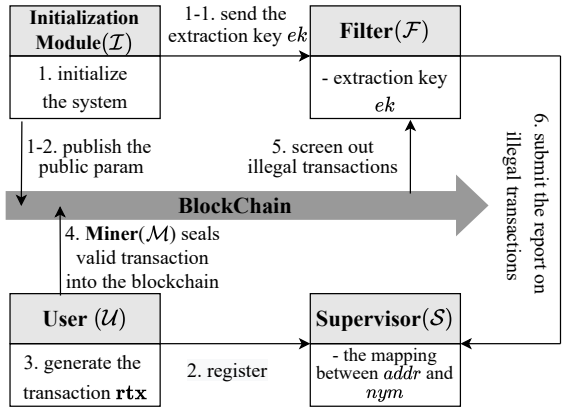

Fig. 1. **System Framework.** First of all, Initialization Module publishes the public parameter on the blockchain and sends the extraction key to Filter. Then, Supervisor registers each user. A user generates a regulatable transaction $\mathtt{rtx}$ and publishes it on the blockchain. Miners verify the validity of each regulatable transaction and seals valid transactions into blocks. Filter screens out illegal transactions sealed in blocks and cooperates with Supervisor to obtain the sender's address and payment amounts of these transactions.

A. System Framework of DAPCR

An entity that independently regulates the anonymous payment system may abuse the regulatory power. To avoid the disadvantage, we divide the regulatory power into two authorities called *Filter* and *Supervisor*. Figure 1 deplicts the system framework of DAPCR, which consists of six entities as follows:

- *Blockchain*. The DAPCR scheme is based on a permissioned blockchain, which is only accessed by users with permissions. Supervisor is responsible for registering users who are allowed to join the permissioned blockchain.
- *Initialization Module*. Initialization Module $\mathcal{I}$ is an honest entity that publishes the public parameter and sends the extraction key to Filter over a secure channel. For decentralization, multiple nodes can form Initialization Module and execute a distributed key generation protocol to generate the public parameter and the extraction key, although Initialization Module can be served by a trusted center.
- *User*. Only with the permission of Supervisor can users join the blockchain and get a pseudonym. The mapping between the user's address and his pseudonym is only known to Supervisor. We assume that users may take any malicious action.
- *Supervisor*. Supervisor $\mathcal{S}$ is a semi-trusted entity responsible for registering users and identifying the sender's address of illegal transactions with the assistance of Filter. $\mathcal{S}$ sets the secret key, the pseudonym and the upper limit of the total payment amount for each user, and publishes the user's certificate. Therefore, the mapping between addresses and pseudonyms is known to $\mathcal{S}$.
- *Filter*. Filter $\mathcal{F}$ is a semi-honest entity responsible for screening out illegal transactions on the permissioned blockchain, and assisting Supervisor in obtaining privacy from illegal transactions. Filter uses an extraction key to extract the regulation label from each transaction and determine the legality of transactions according to regulation labels.
- *Miner*. The role of miners is to verify the validity of regulatable transactions and seal valid transactions into new blocks.

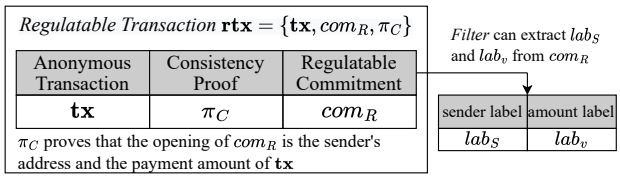

Fig. 2. **Transaction Structure.** A regulatable transaction consists of an anonymous transaction, a regulatable commitment and a consistency proof. Filter can extract the amount label and the sender label from the regulatable transaction.

Transaction Policy. A policy widely used in banks is that each user is given an upper limit and his total payment amount cannot exceed this upper limit in a period. This policy needs to be slightly modified to apply in DAPCR. The transaction policy in DAPCR is that each user is given an upper limit and his total payment amount should be equal to his upper limit in a period. If the total payment volume does not reach the upper limit, the user should publish a transaction of which the recipient is his own to fill the gap in amount. Once some user violates the transaction policy, his transactions in this period are considered as illegal. Please note that miners cannot determine whether transactions comply with the transaction policy. A transaction sealed in the block may be valid but against the policy, which is the object of regulation.

Transaction Structure. A transaction in DAPCR is called a *regulatable transaction* that consists of three parts: an anonymous transaction $\mathtt{tx}$, a regulatable commitment com_R and a consistency proof π_C.

- *Anonymous Transaction.* The anonymous transaction can be viewed as a transaction in DAP, which is responsible for transferring tokens from one user to another.
- *Regulatable Commitment.* The regulatable commitment is a commitment to transaction privacy from which Filter can extract the regulation label with an extraction key.
- *Consistency Proof.* The role of consistency proof is to prove that the opening of a regulatable commitment is consistent with the privacy of an anonymous transaction. Definition 3 of *consistency* is presented in Section VI-B.

Therefore a regulatable transaction can be written as $\mathtt{rtx} = \{\mathtt{tx}, com_R, \pi_C\}$. Figure 2 deplicts the structure of a regulatable transaction.

Regulation Label. The regulation label is an important mechanism to achieve restricted regulation in DAPCR. A regulation label lab consists of a sender label lab_S and an amount label lab_v, i.e. $lab = \{lab_S, lab_v\}$. Filter makes use of the extraction key to extract the regulation label lab from the regulatable commitment com_R in a transaction $\mathtt{rtx}$.

Filter can screen out transactions published by the same user according to the sender label. If the transaction $\mathtt{rtx}$ is published by a user $\mathcal{U}_{nym}$ with the pseudonym nym, lab_S

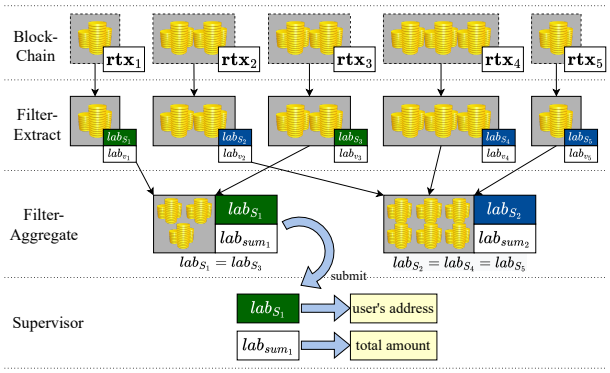

Fig. 3. **Regulation Label Mechanism.**

extracted from $\mathbf{rtx}$ satisfies that $lab_S = nym$. In other words, the same sender label can be extracted from transactions published by the same user.

The amount label lab_v is aggregable and associated with the payment amount of $\mathbf{rtx}$. For all transactions published by the same user $\mathcal{U}$, amount labels extracted from these transactions can be aggregated to a label lab_{sum}. lab_{sum} is associated with the total payment amount from $\mathcal{U}$ and can be used to determine whether the total payment amount exceeds the upper limit of $\mathcal{U}$. If it exceeds, transactions published by $\mathcal{U}$ in this period are illegal and Filter submits the sender label and amount labels of these transactions to Supervisor. It should be emphasized that Filter cannot obtain any privacy from the regulation label.

Considering that the mapping between addresses and pseudonyms is only known to Supervisor and the sender label is equal to the user's pseudonym, Supervisor can identify the address of an illegal user. In addition, Supervisor holds the secret key of each user which can be used to obtain the payment amount from the amount label.

Figure 3 shows the regulation label mechanism with an example.

Formally, a DAPCR scheme is composed by the following probabilistic polynomial-time algorithms.

- $\mathsf{Setup}(1^\lambda) \rightarrow (pp, ek)$. This algorithm takes as input a security parameter λ and outputs a public parameter pp and an extraction key ek. pp will be published on the blockchain and ek will be sent to $\mathcal{F}$ over a secure channel.
- $\mathsf{Register}(addr, pp) \rightarrow (\nu, nym, max, lab_{max}, ce)$. The algorithm takes the address $addr$ of $\mathcal{U}$ and the public parameter pp as input and outputs the secret key ν, the pseudonym nym, the upper limit of total payment volume max, the limit-label lab_{max} and the certificate ce for $\mathcal{U}$.
- $\mathsf{Generate}(\vec{\mathcal{V}_{tp}}, \nu, sk, \mathbf{ce}, pub, pri, pp) \rightarrow \mathbf{rtx}$. This algorithm takes a tuple $(\vec{\mathcal{V}_{tp}}, \nu, sk, \mathbf{ce}, pub, pri, pp)$ as input, and outputs a regulatable transaction $\mathbf{rtx}$. $\vec{\mathcal{V}_{tp}}$ is a vector of transaction privacy. $\mathbf{ce}$ is a set of certificates in which the certificate ce of $\mathcal{U}$ is hidden. pub and pri are part of inputs to $\mathsf{DAP.GenTx}$ in the DAP scheme.
- $\mathsf{Verify}(\mathbf{rtx}, pub, pp) \rightarrow 0/1$. This algorithm, which verify the validity of a regulatable transaction $\mathbf{rtx}$, takes $\mathbf{rtx}$, pub and pp as input, and outputs 1 if $\mathbf{rtx}$ is valid or 0 otherwise.
- $\mathsf{Extract}(\mathbf{rtx}, ek, pp) \rightarrow lab$. This algorithm takes $\mathbf{rtx}$, ek and pp as input and extracts the regulation-label $lab = \{lab_S, lab_v\}$ from $\mathbf{rtx}$. lab_S is the sender label and lab_v is the amount label. If the transaction $\mathbf{rtx}$ is published by the user with the pseudonym nym, lab_S is equal to nym. Therefore, the sender label can be used to screen out transactions from the same sender.
- $\mathsf{Aggregate}(lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}, lab_{max}, nym) \rightarrow \theta$. Let $\mathbf{rtx}_0, \mathbf{rtx}_1, ..., \mathbf{rtx}_{l-1}$ be all l transactions published by a user $\mathcal{U}_{nym}$ with the pseudonym nym. For $i \in \{0, 1, ..., l-1\}$ the amount label lab_{v_i} is extracted from $\mathbf{rtx}_i$. The algorithm takes $lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}$, lab_{max} and nym as input and outputs a report θ. If the total amount paid by $\mathcal{U}_{nym}$ exceeds his upper limit max, $\mathcal{F}$ submits the report θ on illegal transactions to $\mathcal{S}$.
- $\mathsf{Supervise}(\theta, \nu) \rightarrow (addr_S, \{v_0, v_1, ..., v_{l-1}\})$. This algorithm takes a report θ and the secret key ν as input, and outputs the sender's address $addr_S$ and $\{v_0, v_1, ..., v_{l-1}\}$ in which v_i is the payment amount of the transaction $\mathbf{rtx}_i$ published by $\mathcal{U}$.

B. Regulatable Commitment

In this section, we introduce a formal definition of the RegCom scheme, which is the building block of DAPCR.

The RegCom scheme is used to generate the regulatable commitment com_R for the anonymous transaction $\mathbf{tx}$. Filter can extract the regulation label lab from com_R. lab consists of a sender label lab_S and an amount label lab_v. lab_S is equal to the sender's pseudonym nym, so the sender label can be used to screen out transactions published by the same user. lab_v is aggregable and associated with the payment amount of $\mathbf{tx}$. For all transactions published by a user $\mathcal{U}$, Filter can aggregate amount labels extracted from these transactions for lab_{sum} which is associated with the total payment amount from $\mathcal{U}$.

For an anonymous transaction, we assume that the sender's address is $addr_S$, the recipient's address is $addr_R$ and the payment amount is v. A vector $\vec{\mathcal{V}_{tp}}$ can be used to represent transaction privacy, i.e. $\vec{\mathcal{V}_{tp}} = (addr_S, addr_R, v)$. A RegCom scheme consists of the following algorithms:

- $\mathsf{RegCom.Setup}(1^\lambda) \rightarrow (pp_{rc}, ek)$. The algorithm takes a security parameter λ as input, and generates a public parameter pp_{rc} and an extraction key ek. pp_{rc} is published on the blockchain and ek is sent to $\mathcal{F}$ over a secure channel.
- $\mathsf{RegCom.GenKey}(addr, pp_{rc}) \rightarrow \nu$. The algorithm takes the address $addr$ of $\mathcal{U}$ and the public parameter pp_{rc} as input and outputs the secret key ν for $\mathcal{U}$.
- $\mathsf{RegCom.GenCom}(\vec{\mathcal{V}_{tp}}, \nu, pp_{rc}) \rightarrow com_R$. The algorithm takes a vector of transaction privacy $\vec{\mathcal{V}_{tp}}$, a secret key ν and pp_{rc} as input, and generates the regulatable commitment com_R of $\vec{\mathcal{V}_{tp}}$.
- $\mathsf{RegCom.Extract}(com_R, ek, pp_{rc}) \rightarrow lab$. The algorithm takes com_R, ek and pp_{rc} as input, and outputs the regulation label $lab = \{lab_S, lab_v\}$ for com_R. lab_S is the sender label, and lab_v is the amount label.

- $\mathsf{RegCom.Aggregate}(lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}) \rightarrow lab_{sum}$. Let $\mathbf{rtx}_0, \mathbf{rtx}_1, ..., \mathbf{rtx}_{l-1}$ be all l transactions published by a user. For $i \in \{0, 1, ..., l-1\}$ the amount label lab_{v_i} is extracted from the regulatable commitment in $\mathbf{rtx}_i$. The algorithm takes $lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}$ as input, and outputs the label lab_{sum} of total payment volume.

C. Security Definition

To evaluate the security of the DAPCR scheme and the RegCom scheme, we must take into account two different types of adversaries.

- *Type-1 adversary* ($\mathcal{A}_1$) can be considered as Supervisor that colludes with a malicious user. The secret key ν of each user is known to the type-1 adversary. Supervisor can obtain any privacy of the malicious user, such as the private key sk used to generate anonymous transactions, and random numbers used to generate regulatable commitments. The type-1 adversary attempts to obtain the transaction privacy of other users.
- *Type-2 adversary* ($\mathcal{A}_2$) can be considered as Filter that colludes with a malicious user. The extraction key ek is available to the type-2 adversary and can be used to extract the regulation label from each transaction. Any privacy of the malicious user is also known to Filter, such as the private key sk and random numbers used to generate regulatable commitments. The type-2 adversary attempts to obtain the transaction privacy of other users.

Definition 1. *We say that a RegCom scheme is secure if it satisfies three properties: Hiding, Binding and Extractability.*

1. *Hiding.* The regulatable commitment reveals no information about transaction privacy. We say that a RegCom scheme satisfies *Hiding* if, for $i \in \{1, 2\}$, any probabilistic polynomial-time adversaries $\mathcal{A}_i$ and a security parameter λ, the advantage of $\mathcal{A}_i$ in winning the hiding experiment is negligible, i.e. $|Pr[\mathsf{exp}^{\mathsf{Hiding}}_{\mathcal{A}_i}(\lambda) = 1] - 1/2| \leq \mathsf{negl}(\lambda)$.
2. *Binding.* The opening of a regulatable commitment is unique. We say that a RegCom scheme satisfies *Binding* if, for $i \in \{1, 2\}$, any probabilistic polynomial-time adversaries $\mathcal{A}_i$ and a security parameter λ, the advantage of $\mathcal{A}_i$ in winning the binding experiment is negligible, i.e. $Pr[\mathsf{exp}^{\mathsf{Binding}}_{\mathcal{A}_i}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$.
3. *Extractability.* Filter can extract the regulation label from a regulatable commitment using the extraction key. We say that a RegCom scheme satisfies *Extractability* if

$$Pr\left[\begin{array}{l|c} \mathsf{Extract}(com_R, & \mathsf{Setup}(1^\lambda) \rightarrow (pp_{rc}, ek) \\ ek, pp_{rc}) \rightarrow lab & \mathsf{GenKey}(addr, pp_{rc}) \rightarrow \nu \\ & \mathsf{GenCom}(\vec{\mathcal{V}_{tp}}, \nu, pp_{rc}) \rightarrow com_R \end{array}\right] = 1$$

Definition 2. *We say that a DAPCR scheme is secure if it satisfies Indistinguishability, Balance and Extractability.*

1. *Indistinguishability.* The regulatable transaction reveals no information about transaction privacy. We say that a DAPCR scheme satisfies *Indistinguishability* if, for $i \in \{1, 2\}$, any probability polynomial-time adversary $\mathcal{A}_i$ and a security parameter λ, the advantage of $\mathcal{A}_i$ in winning the indistinguishability experiment is negligible, i.e. $Pr|\mathsf{exp}^{\mathsf{ind}}_{\mathcal{A}_i}(\lambda) - \frac{1}{2}| \leq \mathsf{negl}(\lambda)$.
2. *Balance.* This property means that the amount paid by any probabilistic polynomial-time adversaries $\mathcal{A}_i$ cannot exceed his balance for $i \in \{1, 2\}$. Since a regulatable transaction $\mathbf{rtx}$ consists of an anonymous transaction $\mathbf{tx}$, i.e. a transaction in DAP, a regulatable commitment com_R and a consistency proof π_C, the DAPCR scheme satisfies *Balance* if the DAP scheme composing it satisfies *Balance*.
3. *Extractability*. Filter can extract the regulation label from a regulatable transaction. It is easy to see that if a RegCom scheme satisfies *Extractability*, the DAPCR scheme based on the RegCom scheme satisfies *Extractability* too.

The security experiments $\mathsf{exp}^{\mathsf{Hiding}}_{\mathcal{A}_i}(\lambda)$, $\mathsf{exp}^{\mathsf{Binding}}_{\mathcal{A}_i}(\lambda)$ and $\mathsf{exp}^{\mathsf{ind}}_{\mathcal{A}_i}(\lambda)$ are as follows.

- **Hiding experiment** $\mathsf{exp}^{\mathsf{Hiding}}_{\mathcal{A}_i}(\lambda)$:
  1. $(pp_{rc}, ek) \leftarrow \mathsf{RegCom.Setup}(1^\lambda)$
  2. $(addr_{S_0}, v_0, addr_{S_1}, v_1) \leftarrow \mathcal{A}_i(pp_{rc})$
  3. $b \xleftarrow{\$} \{0, 1\}$
  4. $com_{R_b} \leftarrow \mathsf{RegCom.GenCom}(addr_{S_b}, v_b, \nu_b, pp_{rc})$
  5. $b' \leftarrow \mathcal{A}_i(com_{R_b}, addr_{S_0}, v_0, addr_{S_1}, v_1, pp_{rc})$
  6. return 1 if $b = b'$ or 0 otherwise
- **Binding experiment** $\mathsf{exp}^{\mathsf{Binding}}_{\mathcal{A}_i}(\lambda)$:
  1. $(pp_{rc}, ek) \leftarrow \mathsf{RegCom.Setup}(1^\lambda)$
  2. $(addr_{S_0}, v_0, x_0, y_0, addr_{S_1}, v_1, x_1, y_1) \leftarrow \mathcal{A}_i(pp_{rc})$
  3. $com_{R_0} \leftarrow \mathsf{RegCom.GenCom}(addr_{S_0}, v_0, \nu_0, pp_{rc})$ in which x_0 and y_0 are implicit inputs
  4. $com_{R_1} \leftarrow \mathsf{RegCom.GenCom}(addr_{S_1}, v_1, \nu_1, pp_{rc})$ in which x_1 and y_1 are implicit inputs
  5. return 1 if $(addr_{S_0} \neq addr_{S_1} \vee v_0 \neq v_1) \wedge com_{R_0} = com_{R_1}$ or 0 otherwise
- **Indistinguishability experiment** $\mathsf{exp}^{\mathsf{ind}}_{\mathcal{A}_i}(\lambda)$:
  1. $(pp, ek) \leftarrow \mathsf{DAPCR.Setup}(1^\lambda)$
  2. $(addr_{S_0}, addr_{R_0}, v_0, addr_{S_1}, addr_{R_1}, v_1) \leftarrow \mathcal{A}_i(pp)$
  3. $b \xleftarrow{\$} \{0, 1\}$
  4. $\mathbf{rtx}_b \leftarrow \mathsf{DAPCR.Generate}(\vec{\mathcal{V}_{tp}}, \nu, sk, \mathbf{ce}, pub, pri, pp)$ in which $\vec{\mathcal{V}_{tp}} = \{addr_{S_b}, addr_{R_b}, v_b\}$.
  5. $b' \leftarrow \mathcal{A}_i(\mathbf{rtx}_b, addr_{S_0}, addr_{R_0}, v_0, addr_{S_1}, addr_{R_1}, v_1)$
  6. return 1 if $b = b'$ or 0 otherwise
- *In above three experiments, the type-1 adversary $\mathcal{A}_1$ takes secret keys ν_0 and ν_1 of $addr_{S_0}$ and $addr_{S_1}$ as implicit inputs. The type-2 adversary $\mathcal{A}_2$ takes the extraction key ek as implicit inputs.*

V. Regulatable Commitment

In this section, we are going to introduce the construction of RegCom, which is the building block of DAPCR.

An efficient construction of RegCom consists of the following algorithms:

- $\mathsf{RegCom.Setup}$. Initialization Module $\mathcal{I}$ takes a security parameter λ as input and outputs a public parameter pp_{rc} and an extraction key ek. Let g_1 and g_2 be the generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. $\mathcal{I}$ randomly samples $\delta_1, \delta_2, \delta_3, s_1, s_2 \in \mathbb{Z}^*_q$ and computes $\vec{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$,

$\vec{\beta} = (\beta_1, \beta_2, \beta_3)$, $\vec{\gamma} = (\gamma_1, \gamma_2, \gamma_3)$ and $ek = \{ek_1, ek_2\}$, where

$$\begin{aligned} ek_i = g_2^{s_i},\ \alpha_i = g_1^{\delta_i}, \\ \beta_i = \alpha_i^{s_1},\ \gamma_i = \alpha_i^{s_2}. \end{aligned} \tag{1}$$

Finally, $\mathcal{I}$ publishes the public parameter $pp_{rc} = \{\vec{\alpha}, \vec{\beta}, \vec{\gamma}, g_1, g_2\}$ on the blockchain and sends the extraction key ek to $\mathcal{F}$ over a secure channel.

- RegCom.GenKey. If $\mathcal{S}$ approves $\mathcal{U}$ with the address $addr$ to join the system, $\mathcal{S}$ randomly samples a secret key $\nu \in \mathbb{Z}_q^*$ for $\mathcal{U}$. ν is sent to $\mathcal{U}$ over a secure channel.
- RegCom.GenCom. $\mathcal{U}$ takes the vector of transaction privacy $\vec{\mathcal{V}_{tp}} = (addr_S, addr_R, v)$, the secret key ν and pp_{rc} as input and outputs the regulatable commitment com_R. Obviously, $addr_S = addr$. $\mathcal{U}$ randomly samples $x, y \in \mathbb{Z}_q^*$, and computes com, φ, ψ and Y.

$$\begin{aligned} com &= \vec{\alpha} \otimes (addr_S \cdot \nu, v \cdot \nu, x) \\ \varphi &= \vec{\beta} \otimes (y, v \cdot \nu, x) \\ \psi &= \vec{\gamma} \otimes (addr_S \cdot \nu, y, x) \\ Y &= g_2^y \end{aligned} \tag{2}$$

Finally, $\mathcal{U}$ sets $com_R = \{com, \varphi, \psi, Y\}$.

- RegCom.Extract. Filter $\mathcal{F}$ takes a regulatable commitment com_R, an extraction key ek and pp_{rc} as input and outputs the regulation-label lab. $\mathcal{F}$ computes the sender label lab_S and the amount label lab_v, and sets $lab = \{lab_S, lab_v\}$. lab_S is used to screen out transactions published by the same user. lab_v is aggregable. For all transactions published by the same user, amount labels extracted from these transactions can be aggregated to the label of the total payment amount.

$$\begin{aligned} lab_S &= \frac{e(com, ek_1) \cdot e(\beta_1, Y)}{e(\varphi, g_2)} = e(\beta_1, g_2)^{addr_S \cdot \nu} \\ lab_v &= \frac{e(com, ek_2) \cdot e(\gamma_2, Y)}{e(\psi, g_2)} = e(\gamma_2, g_2)^{v \cdot \nu} \end{aligned} \tag{3}$$

- RegCom.Aggregate. Let $\mathbf{rtx}_0, \mathbf{rtx}_1, ..., \mathbf{rtx}_{l-1}$ be all l transactions published by a user $\mathcal{U}$. For $i \in \{0, 1, ..., l-1\}$ the amount label lab_{v_i} is extracted from the regulatable commitment in $\mathbf{rtx}_i$. This algorithm aggregates $lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}$ to obtain lab_{sum} which is the label of the total payment amount. $\mathcal{F}$ computes

$$lab_{sum} = \prod_{j=0}^{l-1} lab_{v_j}. \tag{4}$$

Since $\prod_{j=0}^{l-1} lab_{v_j} = e(\gamma_2, g_2)^{\nu \cdot \sum_{j=0}^{l-1} v_j}$, lab_{sum} is considered the label of the total payment amount from $\mathcal{U}$.

Theorem 1. *The above RegCom scheme satisfies Extractability.*

PROOF. If $com_R = \{com, \varphi, \psi, Y\}$, ek and pp_{rc} is generated according to the RegCom scheme, Filter can extract lab from com_R using ek.

$$\begin{aligned} lab_S &= \frac{e(com, ek_1) \cdot e(\beta_1, Y)}{e(\varphi, g_2)} \\ &= \frac{e(\alpha_1^{addr_S \cdot \nu} \cdot \alpha_2^{v \cdot \nu} \cdot \alpha_3^x, g_2^{s_1}) \cdot e(\beta_1, Y)}{e(\beta_1^y \cdot \beta_2^{v \cdot \nu} \cdot \beta_3^x, g_2)} \\ &= \frac{e(\alpha_1^{addr_S \cdot \nu}, g_2^{s_1}) \cdot e(\alpha_2^{v \cdot \nu} \cdot \alpha_3^x, g_2^{s_1}) \cdot e(\beta_1, Y)}{e(\beta_2^{v \cdot \nu} \cdot \beta_3^x, g_2) \cdot e(\beta_1^y, g_2)} \\ &= e(\beta_1, g_2)^{addr_S \cdot \nu} \\ lab_v &= \frac{e(com, ek_2) \cdot e(\gamma_2, Y)}{e(\psi, g_2)} \\ &= \frac{e(\alpha_1^{addr_S \cdot \nu} \cdot \alpha_2^{v \cdot \nu} \cdot \alpha_3^x, g_2^{s_2}) \cdot e(\gamma_2, Y)}{e(\gamma_1^{addr_S \cdot \nu} \cdot \gamma_2^y \cdot \gamma_3^x, g_2)} \\ &= \frac{e(\alpha_2^{v \cdot \nu}, g_2^{s_2}) \cdot e(\alpha_1^{addr_S \cdot \nu} \cdot \alpha_3^x, g_2^{s_2}) \cdot e(\gamma_2, Y)}{e(\gamma_1^{addr_S \cdot \nu} \cdot \gamma_3^x, g_2) \cdot e(\gamma_2^y, g_2)} \\ &= e(\gamma_2, g_2)^{v \cdot \nu} \end{aligned}$$

Therefore, the RegCom scheme satisfies *Extractability*. □

VI. DAP WITH COLLABORATIVE REGULATION

In this section, we first introduce a DAPCR scheme for honest users and then present how to modify it to a DAPCR scheme against malicious users.

A. DAPCR for Honest Users

The DAPCR scheme for honest users is based on a DAP scheme and a RegCom scheme and consists of four phases: Preparation Phase, Transaction Phase, Verification Phase, and Supervision Phase.

In Preparation Phase, Initialization Module $\mathcal{I}$ initializes the DAPCR system, and Supervisor $\mathcal{S}$ registers users in the system.

Setup. $\mathcal{I}$ takes the security parameter λ as input, and executes algorithms DAP.Setup and RegCom.Setup to generate the public parameter pp and the extraction key ek.

$$\begin{aligned} \mathsf{DAP.Setup}(1^\lambda) &\rightarrow pp_{tx} \\ \mathsf{RegCom.Setup}(1^\lambda) &\rightarrow (pp_{rc}, ek) \end{aligned}$$

$\mathcal{I}$ sets $pp = \{pp_{tx}, pp_{rc}\}$ and publishes it on the blockchain. ek is sent to $\mathcal{F}$ over a secure channel.

Register. $\mathcal{S}$ executes the algorithm RegCom.GenKey to register $\mathcal{U}$ with the address $addr$.

$$\mathsf{RegCom.GenKey}(addr, pp_{rc}) \rightarrow \nu$$

In addition, $\mathcal{S}$ sets the upper limit max of total payment volume, and computes the pseudonym nym and the limit-label lab_{max} for $\mathcal{U}$.

$$\begin{aligned} nym &= e(\beta_1, g_2)^{addr \cdot \nu} \\ lab_{max} &= e(\gamma_2, g_2)^{max \cdot \nu} \end{aligned}$$

Finally, $\mathcal{S}$ adds $(addr, \nu, max, nym, lab_{max})$ to the list $\mathcal{L}_0$, which is initially empty. (ν, max) is sent to $\mathcal{U}$ over a secure channel. (nym, lab_{max}) is sent to $\mathcal{F}$ over a secure channel. After receiving (nym, lab_{max}) from $\mathcal{S}$, $\mathcal{F}$ adds it to the list $\mathcal{L}_1$ which is also initially empty.

In Transaction Phase, $\mathcal{U}$ generates the regulatable transaction $\mathbf{rtx}'$ and publishes it on the blockchain.

Generate. $\mathcal{U}$ first executes the algorithm DAP.GenTx to generate an anonymous transaction $\mathbf{tx}$. $addr_S$ is the address of $\mathcal{U}$, $addr_R$ is the recipient's address and v is the payment amount.

$$\mathsf{DAP.GenTx}(addr_S, addr_R, v, sk, pub, pri, pp_{tx}) \rightarrow \mathbf{tx}$$

Then $\mathcal{U}$ sets $\vec{\mathcal{V}_{tp}} = (addr_S, addr_R, v)$ and executes the algorithm RegCom.GenCom to generate the regulatable commitment to $\vec{\mathcal{V}_{tp}}$.

$$\mathsf{RegCom.GenCom}(\vec{\mathcal{V}_{tp}}, \nu, pp_{rc}) \rightarrow com_R$$

Finally, $\mathcal{U}$ sets $\mathbf{rtx}' = \{\mathbf{tx}, com_R\}$ and publishes it on the blockchain.

In Verification Phase, miners verify the validity of regulatable transactions and seal valid transactions into blocks.

Verify. For a regulatable transaction $\mathbf{rtx}' = \{\mathbf{tx}, com_R\}$, a miner executes the algorithm DAP.VerifyTx to verify the validity of $\mathbf{tx}$.

$$\mathsf{DAP.VerifyTx}(\mathbf{tx}, pp_{tx}) \rightarrow b_0$$

If and only if $b_0 = 1$, $\mathbf{tx}$ is valid and the miner seals $\mathbf{rtx}'$ into a new block. The miner does not have to verify the validity of com_R because users are assumed to be honest in executing the scheme.

In Supervision Phase, Filter $\mathcal{F}$ screens out illegal transactions and submits the report on illegal transactions to Supervisor $\mathcal{S}$. $\mathcal{S}$ receives the report from $\mathcal{F}$ and obtains the sender's address and payment amounts of illegal transactions.

Extract. $\mathcal{F}$ executes the algorithm RegCom.Extract with the extraction key ek to extract the regulation-label $lab = \{lab_S, lab_v\}$ from com_R in $\mathbf{rtx}'$.

$$\mathsf{RegCom.Extract}(com_R, ek, pp_{rc}) \rightarrow lab$$

If $lab_S = nym$, $\mathbf{rtx}'$ is published by $\mathcal{U}_{nym}$ which represents the user with the pseudonym nym. Therefore, $\mathcal{F}$ can make use of RegCom.Extract to screen out all transactions from $\mathcal{U}_{nym}$. Let $\mathbf{rtx}_0, \mathbf{rtx}_1, ..., \mathbf{rtx}_{l-1}$ be all l transactions published by $\mathcal{U}_{nym}$. For $i \in \{0, 1, ..., l-1\}$ the amount label lab_{v_i} is extracted from the regulatable commitment in $\mathbf{rtx}_i$. These amount labels can be aggregated to lab_{sum}, which is the label of the total payment amount.

Aggregate. $\mathcal{F}$ executes the algorithm RegCom.Aggregate to aggregate amount labels extracted from all transactions published by $\mathcal{U}_{nym}$.

$$\mathsf{RegCom.Aggregate}(lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}) \rightarrow lab_{sum}$$

Then $\mathcal{F}$ searches the list $\mathcal{L}_1$ for the limit-label lab_{max} corresponding to nym. Considering that $lab_{max} = e(\gamma_2, g_2)^{max \cdot \nu}$ and $lab_{sum} = e(\gamma_2, g_2)^{\nu \cdot \sum_{j=0}^{l-1} v_j}$, the total payment volume is equal to the upper limit if $lab_{max} = lab_{sum}$. Otherwise, it exceeds the upper limit, and $\mathcal{F}$ submits the report on illegal transactions $\theta = \{lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}, nym\}$ to $\mathcal{S}$.

Supervise. $\mathcal{S}$ attempts to obtain the sender's address and payment amounts of illegal transactions, after receiving the report $\theta = \{lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}, nym\}$ from $\mathcal{F}$. Considering that $\mathcal{S}$ maintains the list $\mathcal{L}_0$ recording the tuple $(addr, \nu, max, nym, lab_{max})$, $\mathcal{S}$ can search $\mathcal{L}_0$ for the address $addr$ and the secret key ν corresponding to nym. In addition, $\mathcal{S}$ can obtain the payment amount v_i of $\mathbf{rtx}_i$ from its amount label lab_{v_i}. It should be emphasized that v_i is not a large number due to the restriction from the user's balance. Since the secret key ν is available to $\mathcal{S}$ and $lab_{v_i} = e(\beta_1, g_2)^{v_i \cdot \nu}$, $\mathcal{S}$ can obtain v_i from lab_{v_i} with *baby-step giant-step* which is an algorithm for computing the order of an element in the finite group [29]. Therefore, $\mathcal{S}$ can obtain the address and payment amounts of the illegal user with the assistance of $\mathcal{F}$.

However, the above scheme is based on the assumption that each user will honestly generate the regulatable transaction $\mathbf{rtx}' = \{\mathbf{tx}, com_R\}$. Unfortunately, we cannot trust any user on the blockchain, who may generate a commitment to incorrect information and attach it to the transaction. The regulator will extract invalid regulatory information from the commitment. To address the problem, a consistency proof should be added to the regulatable transaction. The proof is used to publicly prove that the opening of the commitment is consistent with the transaction privacy.

B. Consistency Proof

In this section, we present how to generate the consistency proof with zk-SNARK.

Definition 3. *We consider that com_R is consistent with $\mathbf{tx}$, if com_R and $\mathbf{tx}$ satisfy the following four conditions.*

1. com_R is output by the algorithm RegCom.GenCom with $(\vec{\mathcal{V}_{tp}}, \nu, pp_{rc})$ as input.
2. ν is the private key.
3. pp_{rc} is the public parameter.
4. $\vec{\mathcal{V}_{tp}}$ is a vector of $\mathbf{tx}$'s privacy.

First of all, the algorithm Register in Section VI-A should be modified to the following version.

Register. $\mathcal{S}$ executes the algorithm RegCom.GenKey to register $\mathcal{U}$ with the address $addr$.

$$\mathsf{RegCom.GenKey}(addr, pp_{rc}) \rightarrow \nu$$

In addition, $\mathcal{S}$ sets the upper limit max of total payment volume, and computes the pseudonym nym, the limit-label lab_{max} and the certificate ce for $\mathcal{U}$.

$$nym = e(\beta_1, g_2)^{addr \cdot \nu}$$
$$lab_{max} = e(\gamma_2, g_2)^{max \cdot \nu}$$
$$ce = \mathsf{Hash}(addr || \nu)$$

Finally, $\mathcal{S}$ adds $(addr, \nu, max, nym, lab_{max}, ce)$ to the list $\mathcal{L}_0$, which is initially empty. ce is published on the blockchain with $\mathcal{S}$'s signature. (ν, max) is sent to $\mathcal{U}$ over a secure channel. (nym, lab_{max}) is sent to $\mathcal{F}$ over a secure channel. After receiving (nym, lab_{max}) from $\mathcal{S}$, $\mathcal{F}$ adds it to the list $\mathcal{L}_1$ which is also initially empty.

To prove that com_R and $\mathbf{tx}$ satisfy the first three conditions, an arithmetic circuit C is built. R is the binary relation constructed by C. The public inputs to C are as follows.

- regulatable commitment $com_R = \{com, \varphi, \psi, Y\}$

- hash h of the vector $\vec{\mathcal{V}_{tp}}$ and the random number y
- a set of certificates $\mathbf{ce} = \{ce_0, ce_1, ..., ce_{n-1}\}$ in which the certificate ce of $\mathcal{U}$ is hidden.
- public parameter pp_{rc}

The private inputs to C are as follows.

- vector $\vec{\mathcal{V}_{tp}} = \{addr_S, addr_R, v\}$ of transaction privacy
- random numbers x and y used in RegCom.GenCom to compute com_R
- secret key ν

C imposes the following constraints on public inputs and private inputs.

1. com_R is generated by the algorithm RegCom.GenCom with inputs $\vec{\mathcal{V}_{tp}}$, ν and pp_{rc}.

$$\begin{aligned} com &= \vec{\alpha} \otimes (addr_S \cdot \nu, v \cdot \nu, x), \\ \varphi &= \vec{\beta} \otimes (y, v \cdot \nu, x), \\ \psi &= \vec{\gamma} \otimes (addr_S \cdot \nu, y, x), \\ Y &= g_2^y. \end{aligned}$$

2. $\vec{\mathcal{V}_{tp}}||y$ is the preimage of h, i.e. $h = \mathsf{Hash}(\vec{\mathcal{V}_{tp}}||y)$.
3. $addr_S||\nu$ is the preimage of ce, i.e. $ce = \mathsf{Hash}(addr_S||\nu)$, and $ce \in \mathbf{ce}$.

To prove that com_R and $\mathbf{tx}$ satisfy the last condition, another arithmetic circuit C' is built. R' is the binary relation constructed by C'. The public inputs to C' are as follows.

- anonymous transaction $\mathbf{tx}$
- hash h of the vector $\vec{\mathcal{V}_{tp}}$ and the random number y
- public parameter pp_{tx}
- pub used to generate $\mathbf{tx}$ in the algorithm DAP.GenTx

The private inputs to C' are as follows.

- vector $\vec{\mathcal{V}_{tp}} = \{addr_S, addr_R, v\}$ of transaction privacy
- random number y used in RegCom.GenCom to compute com_R
- private key sk
- pri used to generate $\mathbf{tx}$ in the algorithm DAP.GenTx

C' imposes the following constraints on public inputs and private inputs.

1. $\mathbf{tx}$ is output by the algorithm DAP.GenTx with inputs $(addr_S, addr_R, v, sk, pub, pri, pp_{tx})$.
2. $\vec{\mathcal{V}_{tp}}||y$ is the preimage of h, i.e. $h = \mathsf{Hash}(\vec{\mathcal{V}_{tp}}||y)$.

A consistency proof can be generated and verified as follows.

Consistency proof

- the consistency proof $\pi_C = \{\pi, \pi', h, \mathbf{ce}\}$ is used to prove that com_R is consistent with $\mathbf{tx}$.

Initialization Module $\mathcal{I}$:

- **KeyGen**:
  1: compute $\mathsf{ZK.KeyGen}(1^\lambda, C) \rightarrow (pk_{zk}, vk_{zk})$
  2: comupte $\mathsf{ZK.KeyGen}(1^\lambda, C') \rightarrow (pk'_{zk}, vk'_{zk})$
- proving and verification keys $(pk_{zk}, vk_{zk}, pk'_{zk}, vk'_{zk})$ are published on the blockchain.

User $\mathcal{U}$:

- **Prove**:
  1: compute $\mathsf{ZK.Prove}(\phi, \varpi, pk_{zk}) \rightarrow \pi$
  2: compute $\mathsf{ZK.Prove}(\phi', \varpi', pk'_{zk}) \rightarrow \pi'$

Miner $\mathcal{M}$:

- **Verify**:
  1: compute $\mathsf{ZK.Verify}(\phi, \pi, vk_{zk}) \rightarrow b$
  2: compute $\mathsf{ZK.Verify}(\phi', \pi', vk'_{zk}) \rightarrow b'$
  3: If and only if $b \cdot b' = 1$, com_R is consistent with $\mathbf{tx}$.
- In above algorithms,
  1: statement $\phi = \{com_R, h, \mathbf{ce}, pp_{rc}\}$
  2: statement $\phi' = \{\mathbf{tx}, h, pp_{tx}, pub\}$
  3: witness $\varpi = \{\vec{\mathcal{V}_{tp}}, x, y, \nu\}$
  4: witness $\varpi' = \{\vec{\mathcal{V}_{tp}}, y, sk, pri\}$

Therefore, the consistency proof $\pi_C = \{\pi, \pi', h, \mathbf{ce}\}$ and the regulatable transaction $\mathbf{rtx} = \{\mathbf{tx}, com_R, \pi_C\}$. Miners and other entities can determine whether com_R is consistent with $\mathbf{tx}$ according to π_C.

VII. Construction

Based on a decentralized anonymous payment scheme, a zk-SNARK scheme and the RegCom scheme in Section V, we propose a generic construction of the DAPCR scheme against malicious users, which is composed by a tuple of algorithms (Setup, Register, Generate, Verify, Extract, Aggregate, Supervise).

I-Preparation Phase

- In Preparation Phase, $\mathcal{I}$ initializes the DAPCR system and $\mathcal{S}$ registers users in the system.

Initialization Module $\mathcal{I}$:

- **Setup**
  - Inputs: security parameter λ
  - Outputs:
    1. public parameter pp
    2. extraction key ek
  - $\mathcal{I}$ executes the following steps to generate the public parameter pp and the extraction key ek:
    1. generate arithmetic circuits C and C' in Section VI-B
    2. compute $\mathsf{DAP.Setup}(1^\lambda) \rightarrow pp_{tx}$
    3. compute $\mathsf{RegCom.Setup}(1^\lambda) \rightarrow (pp_{rc}, ek)$
    4. compute $\mathsf{ZK.KeyGen}(1^\lambda, C) \rightarrow (pk_{zk}, vk_{zk})$
    5. compute $\mathsf{ZK.KeyGen}(1^\lambda, C') \rightarrow (pk'_{zk}, vk'_{zk})$
    6. set $pp = \{pp_{tx}, pp_{rc}, pk_{zk}, vk_{zk}, pk'_{zk}, vk'_{zk}\}$
  - $\mathcal{I}$ publishes pp on the blockchain and sends ek to $\mathcal{F}$ over a secure channel.

Supervisor $\mathcal{S}$:

- **Register:**
  - Inputs:
    1. address $addr$
    2. public parameter pp
  - Outputs:
    1. secret key ν
    2. pseudonym nym
    3. upper limit max
    4. limit-label lab_{max}
    5. certificate ce
  - $\mathcal{S}$ executes RegCom.GenKey to register $\mathcal{U}$:
    1. compute $\mathsf{RegCom.GenKey}(addr, pp_{rc}) \rightarrow \nu$
    2. set the upper limit of total payment volume max for $\mathcal{U}$
    3. compute $nym = e(\beta_1, g_2)^{addr \cdot \nu}$

4. compute $lab_{max} = e(\gamma_2, g_2)^{max \cdot \nu}$
5. compute $ce = \mathsf{Hash}(addr||\nu)$

- $\mathcal{S}$ adds $(addr, \nu, max, nym, lab_{max}, ce)$ to the list $\mathcal{L}_0$, which is initially empty. ce is published on the blockchain with $\mathcal{S}$'s signature. (ν, max) is sent to $\mathcal{U}$ over a secure channel. (nym, lab_{max}) is sent to $\mathcal{F}$ over a secure channel. (After receiving (nym, lab_{max}) from $\mathcal{S}$, $\mathcal{F}$ adds it to the list $\mathcal{L}_1$ which is also initially empty.)

II-Transaction Phase

- In Transaction Phase, $\mathcal{U}$ generates the regulatable transaction $\mathbf{rtx}$ and publishes it on the blockchain.

User $\mathcal{U}$:

- **Generate:**
  - Inputs:
    1. vector $\vec{\mathcal{V}_{tp}} = (addr_S, addr_R, v)$ of transaction privacy
    2. secret key ν
    3. private key sk
    4. a set of certificates $\mathbf{ce}$ in which ce of $\mathcal{U}$ is hidden
    5. pub and pri used in $\mathsf{DAP.GenTx}$
    6. public parameter pp
    7. random numbers x and y used in $\mathsf{RegCom.GenCom}$
  - Outputs: regulatable transaction $\mathbf{rtx}$
  - $\mathcal{U}$ executes the following steps to generate the regulatable transaction $\mathbf{rtx}$:
    1. comupte $\mathsf{DAP.GenTx}(addr_S, addr_R, v, sk, pub, pri, pp_{tx}) \rightarrow \mathbf{tx}$
    2. compute $\mathsf{RegCom.GenCom}(\vec{\mathcal{V}_{tp}}, \nu, pp_{rc}) \rightarrow com_R$
    3. compute $h = \mathsf{Hash}(\vec{\mathcal{V}_{tp}}||y)$
    4. set $\phi = \{com_R, h, \mathbf{ce}, pp_{rc}\}$
    5. set $\varpi = \{\vec{\mathcal{V}_{tp}}, x, y, \nu\}$
    6. set $\phi' = \{\mathbf{tx}, h, pp_{tx}, pub\}$
    7. set $\varpi' = \{\vec{\mathcal{V}_{tp}}, y, sk, pri\}$
    8. compute $\mathsf{ZK.Prove}(\phi, \varpi, pk_{zk}) \rightarrow \pi$
    9. compute $\mathsf{ZK.Prove}(\phi', \varpi', pk'_{zk}) \rightarrow \pi'$
    10. set $\pi_C = \{\pi, \pi', h, \mathbf{ce}\}$ and $\mathbf{rtx} = \{\mathbf{tx}, com_R, \pi_C\}$
  - $\mathcal{U}$ publishes $\mathbf{rtx}$ on the blockchain.

III-Verification Phase

- In Verification Phase, miners verify the validity of regulatable transactions and seal valid transactions into blocks.

Miner $\mathcal{M}$:

- **Verify:**
  - Inputs
    1. regulatable transaction $\mathbf{rtx} = \{\mathbf{tx}, com_R, \pi_C\}$ where $\pi_C = \{\pi, \pi', h, \mathbf{ce}\}$
    2. pub used in $\mathsf{DAP.GenTx}$
    3. public parameter pp
  - $\mathcal{M}$ executes the following steps to verify the validity of the regulatable transaction $\mathbf{rtx}$:
    1. set $\phi = \{com_R, h, \mathbf{ce}, pp_{rc}\}$
    2. set $\phi' = \{\mathbf{tx}, h, pp_{tx}, pub\}$
    3. compute $\mathsf{DAP.VerifyTx}(\mathbf{tx}, pp_{tx}) \rightarrow b_1$
    4. compute $\mathsf{ZK.Verify}(\phi, \pi, vk_{zk}) \rightarrow b_2$
    5. compute $\mathsf{ZK.Verify}(\phi', \pi', vk'_{zk}) \rightarrow b_3$
    6. If $b_1 = 1$, $\mathbf{tx}$ is valid. If $b_2 \cdot b_3 = 1$, com_R is consistent with $\mathbf{tx}$. Therefore, if $b_1 \cdot b_2 \cdot b_3 = 1$, $\mathbf{rtx}$ is valid. Otherwise, $\mathbf{rtx}$ is invalid.
  - $\mathcal{M}$ seals valid regulatable transactions into blocks.

IV-Supervision Phase

- In Supervision Phase, Filter $\mathcal{F}$ screens out illegal transactions and submits the report on illegal transactions to Supervisor $\mathcal{S}$. $\mathcal{S}$ receives the report from $\mathcal{F}$ and obtains the sender's address and payment amounts of illegal transactions.

Filter $\mathcal{F}$:

- **Extract:**
  - Inputs:
    1. regulatable transaction $\mathbf{rtx} = \{\mathbf{tx}, com_R, \pi_C\}$
    2. extraction key ek
    3. public parameter pp
  - Outputs: regulation-label lab
  - $\mathcal{F}$ executes the algorithm $\mathsf{RegCom.Extract}$ to obtain the regulation-label $lab = \{lab_S, lab_v\}$ from $\mathbf{rtx}$:

$$\mathsf{RegCom.Extract}(com_R, ek, pp_{rc}) \rightarrow lab$$

- **Aggregate:**
  - Inputs:
    1. $lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}$
    2. limit-label lab_{max} of $\mathcal{U}_{nym}$
    3. pseudonym nym
  - Outputs: report θ on illegal transactions
  - $\mathcal{F}$ aggregates amount labels to get the label of total payment volume. Let $\mathbf{rtx}_0, \mathbf{rtx}_1, ..., \mathbf{rtx}_{l-1}$ be all l transactions published by $\mathcal{U}_{nym}$. lab_{v_i} is extracted from $\mathbf{rtx}_i$ for $i \in \{0, 1, ..., l-1\}$.
    1. compute $\mathsf{RegCom.Aggregate}(lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}) \rightarrow lab_{sum}$
  - If and only if $lab_{sum} \neq lab_{max}$, transactions from $\mathcal{U}_{nym}$ are illegal. $\mathcal{F}$ sets $\theta = \{lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}, nym\}$ and submits it to $\mathcal{S}$.

Supervisor $\mathcal{S}$:

- **Supervise:**
  - Inputs:
    1. $\theta = \{lab_{v_0}, lab_{v_1}, ..., lab_{v_{l-1}}, nym\}$
    2. secret key ν of $\mathcal{U}_{nym}$
  - Outputs:
    1. address $addr$ of $\mathcal{U}_{nym}$
    2. payment amount v_i of $\mathbf{rtx}_i$ for $i \in \{0, 1, ..., l-1\}$
  - $\mathcal{S}$ searches the list $\mathcal{L}_0$ for the address $addr$ and the private key ν corresponding to nym, and then extracts the payment amount v_i of $\mathbf{rtx}_i$ from lab_{v_i}.

VIII. Security Analysis

In this section, we analyze the security of our scheme.

A. Security Analysis of RegCom

Theorem 2. *If the variant DDH assumption holds, no probabilistic polynomial-time adversary can break the Hiding of the RegCom scheme in Section V with non-negligible advantage.*

Theorem 2 follows from Lemma 1 and Lemma 2.

Lemma 1. *If a type-1 adversary $\mathcal{A}_1$ breaks the Hiding of the RegCom scheme with advantage $\mathcal{E}_1$, a simulator $\mathcal{B}$ can use $\mathcal{A}_1$ to solve the variant DDH problem with advantage $\mathcal{E}_{\mathsf{DDH}} = \frac{\mathcal{E}_1}{2}$.*

PROOF. For a problem instance (g, g^a, g^b, g^{ac}, T) where $g \in \mathbb{G}_1$ and $a, b, c \in \mathbb{Z}_q^*$, if a type-1 adversary $\mathcal{A}_1$ can breaks the *Hiding* of the RegCom scheme proposed with advantage

$\mathcal{E}_1$, we can construct a simulator $\mathcal{B}$ to solve the variant DDH problem. $\mathcal{B}$ interacts with $\mathcal{A}_1$ as follows.

Setup. Let g_2 be the generator of $\mathbb{G}_2$. $\mathcal{B}$ randomly samples $\delta_1, \delta_2, \delta_3, k \in \mathbb{Z}_q^*$, and computs $\beta_i = (g^a)^{\delta_i}$, $\gamma_i = (g^b)^{\delta_i}$ and $\alpha_i = (g^b)^{k\delta_i}$ where $g_1 = g^{bk}$, $s_1 = \frac{a}{bk}$ and $s_2 = \frac{1}{k}$ for $i \in \{1, 2, 3\}$. Then, $\mathcal{B}$ publishes g_1, g_2, $\vec{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$, $\vec{\beta} = (\beta_1, \beta_2, \beta_3)$ and $\vec{\gamma} = (\gamma_1, \gamma_2, \gamma_3)$.

Challenge. $\mathcal{A}_1$ chooses $(addr_{S_0}, v_0, addr_{S_1}, v_1)$ in which $addr_{S_0} \neq addr_{S_1}$. Receiving $(addr_{S_0}, v_0, addr_{S_1}, v_1)$ from $\mathcal{A}_1$, $\mathcal{B}$ randomly chooses $b \in \{0, 1\}$. Then $\mathcal{B}$ randomly samples $y \in \mathbb{Z}_q^*$, and computes $com_{R_b} = \{com_b, \varphi_b, \psi_b, Y_b\}$.

$$\begin{aligned} com_b &= (\alpha_1, \alpha_2, T) \otimes (\nu_b \cdot addr_{S_b}, \nu_b v_b, k\delta_3) \\ \varphi_b &= (\beta_1, \beta_2, g^{ac}) \otimes (y, \nu_b v_b, \delta_3) \\ \psi_b &= (\gamma_1, \gamma_2, T) \otimes (\nu_b \cdot addr_{S_b}, y, \delta_3) \\ Y_b &= g_2^y \end{aligned}$$

Finally, $\mathcal{B}$ sends com_{R_b} to $\mathcal{A}_1$.

Guess. Receiving com_{R_b} from $\mathcal{B}$, $\mathcal{A}_1$ outputs a guess b'. If $b' = b$ the simulator $\mathcal{B}$ outputs *True*. Otherwise, it outputs *False*.

Then we analyze the correctness and the indistinguishability of the simulation as follows.

Correctness. We first analyze φ_b in com_{R_b}.

$$\begin{aligned} \varphi_b &= (\beta_1, \beta_2, g^{ac}) \otimes (y, \nu_b v_b, \delta_3) \\ &= (\beta_1, \beta_2, \beta_3) \otimes (y, \nu_b v_b, c) \end{aligned}$$

If $T = g^{bc}$,

$$\begin{aligned} com_b &= (\alpha_1, \alpha_2, T) \otimes (\nu_b \cdot addr_{S_b}, \nu_b v_b, k\delta_3) \\ &= (\alpha_1, \alpha_2, \alpha_3) \otimes (\nu_b \cdot addr_{S_b}, \nu_b v_b, c) \\ \psi_b &= (\gamma_1, \gamma_2, T) \otimes (\nu_b \cdot addr_{S_b}, y, \delta_3) \\ &= (\gamma_1, \gamma_2, \gamma_3) \otimes (\nu_b \cdot addr_{S_b}, y, c) \end{aligned}$$

Therefore, $\mathcal{B}$ can generate a correct challenge commitment for $\mathcal{A}_1$.

Indistinguishability. In the simulation, $\mathcal{B}$ uses $\frac{a}{bk}$ and $\frac{1}{k}$ instead of s_1 and s_2 to generate the public parameter, and uses c instead of the random number x to generate the challenge commitment. s_1, s_2 and x are randomly sampled from $\mathbb{Z}_q^*$. In the problem instance (g, g^a, g^b, g^{ac}, T), a, b and c can be considered random. In the setup phase, k is randomly sampled from $\mathbb{Z}_q^*$. Thus, $\frac{a}{bk}$ and $\frac{1}{k}$ are random. Therefore, the simulation satisfies randomness and is indistinguishable from the real hiding experiment. In addition, since the simulation of the RegCom scheme will not be terminated, the probability of generating a valid challenge commitment is 1.

Advantage of $\mathcal{A}_1$. B can use $\mathcal{A}_1$ to solve the variant DDH problem with advantage $\frac{\mathcal{E}_1}{2}$.

If $T = g^{bc}$, the simulation is indistinguishable from the real hiding experiment. So $\mathcal{A}_1$ guesses the correct address $addr_{S_b}$ and amount v_b with the probability $P_1 = \frac{\mathcal{E}_1}{2} + \frac{1}{2}$.

If $T \neq g^{bc}$, it can be written as $T = g^t$ where $t \xleftarrow{\$} \mathbb{Z}_q^*$. $\mathcal{A}_1$ gets $\delta_1, \delta_2, \delta_3, s_1 = \frac{a}{bk}, s_2 = \frac{1}{k}$ from the public parameter pp_{rc}, and gets

$$\begin{aligned} r_1 &= (\delta_1 \nu_b addr_{S_b} + \delta_2 \nu_b v_b)kb + \delta_3 kt \\ r_2 &= (\delta_1 y + \delta_2 \nu_b v_b + \delta_3 c)s_1 kb \\ r_3 &= (\delta_1 \nu_b addr_{S_b} + \delta_2 y)b + \delta_3 t \end{aligned}$$

and y from com_R. Since b, t and c are random and unknown to $\mathcal{A}_1$, r_2 is obviously random and independent of r_1, r_3 and y. We can get the determinant $\mathbf{D}_1$ of coefficients of r_1 and r_3.

$$\mathbf{D}_1 = \begin{vmatrix} (\delta_1 \nu_b addr_{S_b} + \delta_2 \nu_b v_b)k & \delta_3 k \\ (\delta_1 \nu_b addr_{S_b} + \delta_2 y) & \delta_3 \end{vmatrix}$$

where the absolute value of $\mathbf{D}_1$ is nonzero. Thus, r_1, r_2 and r_3 are random and independent of each other. In addition, r_1, r_2, r_3 and y are independent of the public parameter. $\mathcal{A}_1$ guesses the correct address $addr_{S_b}$ and amount v_b with the probability $P_2 = \frac{1}{2}$.

The advantage of $\mathcal{B}$ in solving the variant DDH problem is $\mathcal{E}_{\mathsf{DDH}} = P_1 - P_2 = \frac{\mathcal{E}_1}{2}$. □

Lemma 2. *If a type-2 adversary $\mathcal{A}_2$ breaks the Hiding of the RegCom scheme with advantage $\mathcal{E}_2$, a simulator $\mathcal{B}$ can use $\mathcal{A}_2$ to solve the variant DDH problem with advantage $\mathcal{E}_{\mathsf{DDH}} = \frac{\mathcal{E}_2}{2}$.*

PROOF. For a problem instance (g, g^a, g^b, g^{ac}, T) where $g \in \mathbb{G}_1$ and $a, b, c \in \mathbb{Z}_q^*$, if a type-2 adversary $\mathcal{A}_2$ can breaks the *Hiding* of the RegCom scheme proposed with advantage $\mathcal{E}_2$, we can construct a simulator $\mathcal{B}$ to solve the variant DDH problem. $\mathcal{B}$ interacts with $\mathcal{A}_2$ as follows.

Setup. Let g_2 be the generator of $\mathbb{G}_2$. $\mathcal{B}$ sets $\alpha_1 = g^a$ and $\alpha_2 = g^b$ where $g_1 = g$, $\delta_1 = a$ and $\delta_2 = b$, and randomly samples $\delta_3, s_1, s_2 \in \mathbb{Z}_q^*$. Then $\mathcal{B}$ computes $ek_1 = g_2^{s_1}$, $ek_2 = g_2^{s_2}$, $\alpha_3 = g^{\delta_3}$, $\beta_i = \alpha_i^{s_1}$ and $\gamma_i = \alpha_i^{s_2}$ for $i \in \{1, 2, 3\}$. Finally, $\mathcal{B}$ publishes g_1, g_2, $\vec{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$, $\vec{\beta} = (\beta_1, \beta_2, \beta_3)$ and $\vec{\gamma} = (\gamma_1, \gamma_2, \gamma_3)$, and sends $ek = \{ek_1, ek_2\}$ to $\mathcal{A}_2$.

Challenge. $\mathcal{A}_2$ chooses $(addr_{S_0}, v_0, addr_{S_1}, v_1)$ in which $addr_{S_0} \neq addr_{S_1}$. Receiving $(addr_{S_0}, v_0, addr_{S_1}, v_1)$ from $\mathcal{A}_2$, $\mathcal{B}$ randomly chooses $b \in \{0, 1\}$. Then $\mathcal{B}$ randomly samples $x, y \in \mathbb{Z}_q^*$, and computes $com_{R_b} = \{com_b, \varphi_b, \psi_b, Y_b\}$.

$$\begin{aligned} com_b &= (g^{ac}, T, \alpha_3) \otimes (addr_{S_b}, v_b, x) \\ \varphi_b &= (\beta_1, T, \beta_3) \otimes (y, s_1 v_b, x) \\ \psi_b &= (g^{ac}, \gamma_2, \gamma_3) \otimes (s_2 \cdot addr_{S_b}, y, x) \\ Y_b &= g_2^y \end{aligned}$$

Finally, $\mathcal{B}$ sends com_{R_b} to $\mathcal{A}_2$.

Guess. Receiving com_{R_b} from $\mathcal{B}$, $\mathcal{A}_2$ outputs a guess b'. If $b' = b$ the simulator $\mathcal{B}$ outputs *True*. Otherwise, it outputs *False*.

Then we analyze the correctness and the indistinguishability of the simulation as follows.

Correctness. We first analyze ψ_b in com_{R_b}.

$$\begin{aligned} \psi_b &= (g^{ac}, \gamma_2, \gamma_3) \otimes (s_2 \cdot addr_{S_b}, y, x) \\ &= (\gamma_1, \gamma_2, \gamma_3) \otimes (c \cdot addr_{S_b}, y, x) \end{aligned}$$

where $c = \nu_b$. If $T = g^{bc}$,

$$\begin{aligned} com_b &= (g^{ac}, T, \alpha_3) \otimes (addr_{S_b}, v_b, x) \\ &= (\alpha_1, \alpha_2, \alpha_3) \otimes (c \cdot addr_{S_b}, v_b c, x) \\ \varphi_b &= (\beta_1, T, \beta_3) \otimes (y, s_1 v_b, x) \\ &= (\beta_1, \beta_2, \beta_3) \otimes (y, v_b c, x) \end{aligned}$$

where $c = \nu_b$. Therefore, $\mathcal{B}$ can generate a correct challenge commitment for $\mathcal{A}_2$.

Indistinguishability. In the simulation, $\mathcal{B}$ uses a and b instead of δ_1 and δ_2 to generate the public parameter and the extraction key, and uses c instead of the secret key ν_b to generate the challenge commitment com_{R_b}. δ_1, δ_2 and ν_b are randomly sampled from $\mathbb{Z}_q^*$. In the problem instance (g, g^a, g^b, T), a, b and c can be considered random. Thus, the simulation satisfies randomness, and is indistinguishable from the real hiding experiment. In addition, since the simulation of the RegCom scheme will not be terminated, the probability of generating a valid challenge commitment is 1.

Advantage of $\mathcal{A}_2$. B can use $\mathcal{A}_2$ to solve the variant DDH problem with advantage $\frac{\mathcal{E}_2}{2}$.

If $T = g^{bc}$, the simulation is indistinguishable from the real hiding experiment. So $\mathcal{A}_2$ guesses the correct address $addr_{S_b}$ and amount v_b with the probability $P_3 = \frac{\mathcal{E}_2}{2} + \frac{1}{2}$.

If $T \neq g^{bc}$, it can be written as $T = g^t$ where $t \xleftarrow{\$} \mathbb{Z}_q^*$. $\mathcal{A}_2$ gets $\delta_1, \delta_2, \delta_3, s_1, s_2$ from the public parameter and the extraction key, and gets

$$\begin{aligned} r_1 &= \delta_1 c \cdot addr_{S_b} + v_b t + \delta_3 x \\ r_2 &= s_1 v_b t + s_1 \delta_3 x + s_1 \delta_1 y \\ r_3 &= s_2 \delta_1 c \cdot addr_{S_b} + s_2 \delta_3 x + s_2 \delta_2 y \end{aligned}$$

and y from com_R. Since c, t and x are random and unknown to $\mathcal{A}_2$, we can get the determinant $\mathbf{D}_2$ of coefficients of r_1, r_2 and r_3.

$$\mathbf{D}_2 = \begin{vmatrix} \delta_1 \cdot addr_{S_b} & v_b & \delta_3 \\ 0 & s_1 v_b & s_1 \delta_3 \\ s_2 \delta_1 \cdot addr_{S_b} & 0 & s_2 \delta_3 \end{vmatrix}$$

where the absolute value of $\mathcal{D}_2$ is $s_1 s_2 \delta_1 \delta_3 v_b \cdot addr_{S_b} \neq 0$. Thus, r_1, r_2, r_3 and y are random and independent of each other. In addition, r_1, r_2, r_3 and y are independent of the public parameter and the extraction key. $\mathcal{A}_2$ has no advantage in outputing $b' = b$. $\mathcal{A}_2$ guesses the correct address $addr_{S_b}$ and amount v_b with the probability $P_4 = \frac{1}{2}$.

The advantage of $\mathcal{B}$ in solving the variant DDH problem is $\mathcal{E}_{\mathsf{DDH}} = P_3 - P_4 = \frac{\mathcal{E}_2}{2}$. □

Theorem 3. *If the DL assumption holds, no probabilistic polynomial-time adversary can break the Binding of the RegCom scheme in Section V with non-negligible advantage.*

PROOF. Assume that there exists an adversary $\mathcal{A}$ who holds the secret key of each address, the extraction key ek, and $\delta_1, \delta_2, s_1, s_2$ used to generate the public parameter. It is easy to see that $\mathcal{A}$ gets more information than the type-1 adversary $\mathcal{A}_1$ or the type-2 adversary $\mathcal{A}_2$. Therefore, if $\mathcal{A}$ cannot break the *Binding* of the RegCom scheme, $\mathcal{A}_1$ and $\mathcal{A}_2$ cannot too.

First, $\mathcal{A}$ generates two tuples $(addr_{S_0}, v_0, x_0, y_0)$ and $(addr_{S_1}, v_1, x_1, y_1)$ where $addr_{S_0} \neq addr_{S_1} \vee v_0 \neq v_1$. $\mathcal{A}$ computes

$$com_{R_0} \leftarrow \mathsf{RegCom.GenCom}(addr_{S_0}, v_0, \nu_0, pp_{rc})$$

which takes x_0 and y_0 as implicit inputs, and

$$com_{R_1} \leftarrow \mathsf{RegCom.GenCom}(addr_{S_1}, v_1, \nu_1, pp_{rc})$$

which takes x_1 and y_1 as implicit inputs. If $\mathcal{A}$ breaks the *Binding* of the RegCom scheme successfully, $com_{R_0} = com_{R_1}$, i.e. $com_0 = com_1$, $\varphi_0 = \varphi_1$, $\psi_0 = \psi_1$ and $Y_0 = Y_1$, thus $y_0 = y_1$. Considering that $\delta_1, \delta_2, s_1, s_2$ are available to $\mathcal{A}$, the adversary can get

$$\begin{aligned} \varphi_0 &= g_1^{s_1(\delta_1 y_0 + \delta_2 \nu_0 v_0)} \cdot \alpha_3^{s_1 x_0} \\ \varphi_1 &= g_1^{s_1(\delta_1 y_1 + \delta_2 \nu_1 v_1)} \cdot \alpha_3^{s_1 x_1} \\ \psi_0 &= g_1^{s_2(\delta_1 \nu_0 addr_{S_0} + \delta_2 y_0)} \cdot \alpha_3^{s_2 x_0} \\ \psi_1 &= g_1^{s_2(\delta_1 \nu_1 addr_{S_1} + \delta_2 y_1)} \cdot \alpha_3^{s_2 x_1} \end{aligned}$$

If $addr_{S_0} \neq addr_{S_1}$, we get that $s_2(\delta_1 \nu_0 addr_{S_0} + \delta_2 y_0) \neq s_2(\delta_1 \nu_0 addr_{S_0} + \delta_2 y_0)$ and $x_0 \neq x_1$, since the secret key ν_i of $addr_{S_i}$ is randomly sampled from $\mathbb{Z}_q^*$ by Initialization Module. Therefore, $\mathcal{A}$ can get

$$\mathsf{log}_{g_1}(\alpha_3) = (\delta_1 \nu_1 addr_{S_1} - \delta_1 \nu_0 addr_{S_0})/(x_0 - x_1)$$

If $v_0 \neq v_1 \wedge addr_{S_0} = addr_{S_1}$, $\nu_0 = \nu_1$ and $s_1(\delta_1 y_0 + \delta_2 \nu_0 v_0) \neq s_1(\delta_1 y_1 + \delta_2 \nu_1 v_1)$. Since $\varphi_0 = \varphi_1$, $x_0 \neq x_1$. Thus, $\mathcal{A}$ can get

$$\mathsf{log}_{g_1}(\alpha_3) = (\delta_2 \nu_0 v_0 - \delta_2 \nu_1 v_1)/(x_1 - x_0)$$

$\mathcal{A}$ solves the DL problem while breaking the *Binding* of the RegCom scheme. The advantage of $\mathcal{A}$ in solving the DL problem is $\mathcal{E}_{\mathsf{DL}} \geqslant \mathcal{E}_3$ where $\mathcal{E}_3$ is the advantage of $\mathcal{A}$ in breaking the *Binding* property. □

B. Security Analysis of DAPCR

We design an *Indistinguishability* experiment $\mathsf{exp}^{\mathsf{ind}}_{\mathcal{A}_i}(\lambda)$ between an adversary $\mathcal{A}_i$ and the challenger $\mathcal{C}$. The oracle $\mathcal{O}_{\mathsf{DAPCR}}$ is executed by $\mathcal{C}$ to respond to queries on different algorithms in the DAPCR scheme.

- **Query on** DAPCR.Register**.** $\mathcal{C}$ maintains the list $\mathcal{L}_0$. Receiving a query on DAPCR.Register with $addr$, $\mathcal{C}$ searches $\mathcal{L}_0$ for the secret key ν associated with $addr$ and outputs ν.
- **Query on** DAP.GenAddr**.** $\mathcal{C}$ maintains the list $\mathcal{L}_{addr}$ recording the address $addr$ and the private key sk associated with it. Receiving a query on DAP.GenAddr with $addr$, $\mathcal{C}$ searches $\mathcal{L}_{addr}$ for the private key sk associated with $addr$ and outputs sk.
- **Query on** DAPCR.Generate**.** Receiving a query on DAPCR.Generate with $(addr_S, addr_R, v)$, $\mathcal{C}$ outputs a regulatable transaction $\mathbf{rtx} = \{\mathbf{tx}, com_R, \pi, \pi', h, \mathbf{ce}\}$.

Please note that the type-1 adversary $\mathcal{A}_1$ does not send any query on DAPCR.Register to $\mathcal{C}$, since the secret key ν of each address $addr$ is available to $\mathcal{A}_1$.

The *Indistinguishability* experiment $\mathsf{exp}^{\mathsf{ind}}_{\mathcal{A}_i}(\lambda)$ between $\mathcal{A}_i$ and $\mathcal{C}$ is as follows.

1. $\mathcal{C}$ executes the algorithm DAPCR.Setup to generate the public parameter pp and the extraction key ek. pp is sent to $\mathcal{A}_i$. If $i = 2$, ek is sent to $\mathcal{A}_2$.
2. $\mathcal{C}$ uses $\mathcal{O}_{\mathsf{DAPCR}}$ to respond to queries on different algorithms from $\mathcal{A}_i$, until $\mathcal{A}_i$ outputs two tuples $(addr_{S_0}, addr_{R_0}, v_0)$ and $(addr_{S_1}, addr_{R_1}, v_1)$.
3. $\mathcal{C}$ randomly chooses $b \in \{0, 1\}$ and computes the regulatable transaction $\mathbf{rtx}_b = \{\mathbf{tx}_b, com_{R_b}, \pi_b, \pi'_b, h_b, \mathbf{ce}_b\}$ for $\mathcal{A}_i$.

4. $\mathcal{C}$ uses $\mathcal{O}_{\mathsf{DAPCR}}$ to respond to queries on different algorithms from $\mathcal{A}_i$, until $\mathcal{A}_i$ outputs b'.
5. $\mathsf{exp}^{\mathsf{ind}}_{\mathcal{A}_i}(\lambda)$ outputs *True* if $b = b'$ or *False* otherwise.

Theorem 4. *No probabilistic polynomial-time adversary can break the Indistinguishability of the DAPCR scheme with non-negligible advantage, if the DAP scheme, the zk-SNARK scheme, the RegCom scheme and the hash function making up it are secure.*

PROOF. We use $\mathcal{G}_{real}$ to denote the experiment $\mathsf{exp}^{\mathsf{ind}}_{\mathcal{A}_i}(\lambda)$ executed in the real world. To analyze the security of DAPCR, we design a simulation $\mathcal{G}_{sim}$. In $\mathcal{G}_{sim}$, the challenger $\mathcal{C}$ interacts with $\mathcal{A}_i$ as in $\mathcal{G}_{real}$. The only modification is that $\mathcal{C}$ outputs a challenge transaction $\mathbf{rtx}_{\mathcal{G}_{sim}} = \{\mathbf{tx}_b, com_R, \pi, \pi', h, \mathbf{ce}\}$ in which com_R, π, π', h and $\mathbf{ce}$ are independent of b. Therefore, the advantage of $\mathcal{A}_i$ in winning $\mathcal{G}_{sim}$ is $\mathcal{E}_{sim} = \mathcal{E}_{\mathsf{DAP}}$, where $\mathcal{E}_{\mathsf{DAP}}$ is the advantage of any probabilistic polynomial-time adversary in distinguishing $\mathbf{tx}_0$ from $\mathbf{tx}_1$. $\mathcal{E}_{\mathcal{G}_j}$ represents the absolute value of the difference between the advantage of $\mathcal{A}_i$ in winning $\mathcal{G}_j$ and that in winning $\mathcal{G}_{real}$.

Game $\mathcal{G}_1$. This game $\mathcal{G}_1$ is the same as $\mathcal{G}_{real}$, but the only modification is that $\mathcal{C}$ uses trapdoors to simulate proofs in the challenge transaction. To generate trapdoors, $\mathcal{C}$ executes $(pk_z, vk_z, \mathsf{trap}) \leftarrow \mathsf{Sim}(1^\lambda, C)$ and $(pk'_z, vk'_z, \mathsf{trap}') \leftarrow \mathsf{Sim}(1^\lambda, C')$ in the setup phase. Then $\mathcal{C}$ computs $\pi \leftarrow \mathsf{Sim}(\phi, pk_{zk}, \mathsf{trap})$ and $\pi' \leftarrow \mathsf{Sim}(\phi', pk'_{zk}, \mathsf{trap}')$. The challenge transaction in $\mathcal{G}_1$ is $\mathbf{rtx}_{\mathcal{G}_1} = \{\mathbf{tx}_b, com_{R_b}, \pi, \pi', h_b, \mathbf{ce}_b\}$. Considering that a secure zk-SNARK scheme satisfies zero-knowledge, the distribution of π and π' is identical to that of π_b and π'_b in $\mathcal{G}_{real}$. Therefore, $\mathcal{E}_{\mathcal{G}_1} \leqslant \mathsf{negl}(\lambda)$.

Game $\mathcal{G}_2$. This game is the same as $\mathcal{G}_1$, but the only modification is that $\mathcal{C}$ uses a random string h to replace the hash value h_b in $\mathbf{rtx}_{\mathcal{G}_1}$. The challenge transaction in $\mathcal{G}_2$ is $\mathbf{rtx}_{\mathcal{G}_2} = \{\mathbf{tx}_b, com_{R_b}, \pi, \pi', h, \mathbf{ce}_b\}$. Since the hash function is secure, the distribution of h is identical to that of h_b in $\mathcal{G}_1$. Therefore, $|\mathcal{E}_{\mathcal{G}_2} - \mathcal{E}_{\mathcal{G}_1}| \leqslant \mathsf{negl}(\lambda)$.

Game $\mathcal{G}_3$. This game is the same as $\mathcal{G}_2$, but the only modification is that $\mathcal{C}$ uses another user's certificate to replace the certificate of $addr_{S_b}$ in $\mathbf{ce}$. The challenge transaction in $\mathcal{G}_3$ is $\mathbf{rtx}_{\mathcal{G}_3} = \{\mathbf{tx}_b, com_{R_b}, \pi, \pi', h, \mathbf{ce}\}$. Since the hash function is secure, the distribution of the new certificate is identical to that of the certificate of $addr_{S_b}$. Therefore, $|\mathcal{E}_{\mathcal{G}_3} - \mathcal{E}_{\mathcal{G}_2}| \leqslant \mathsf{negl}(\lambda)$.

Game $\mathcal{G}_{sim}$. This game is the same as $\mathcal{G}_3$, but the only modification is that $\mathcal{C}$ uses com_R to replace com_{R_b}. com_R satisfies that $com_R \leftarrow \mathsf{RegCom.GenCom}(addr, v, \nu, pp_{rc})$ where $addr \neq addr_{S_b} \wedge v \neq v_b \wedge \nu \neq \nu_b$. The challenge transaction in $\mathcal{G}_{sim}$ is $\mathbf{rtx}_{\mathcal{G}_{sim}} = \{\mathbf{tx}_b, com_R, \pi, \pi', h, \mathbf{ce}\}$. Since the RegCom scheme satisfies *Hiding*, the advantage of $\mathcal{A}_i$ in distinguishing com_R from com_{R_b} is negligible. Thus, $|\mathcal{E}_{\mathcal{G}_{sim}} - \mathcal{E}_{\mathcal{G}_3}| \leqslant \mathsf{negl}(\lambda)$.

Since the advantage of $\mathcal{A}_i$ in winning $\mathcal{G}_{sim}$ is $\mathcal{E}_{sim} = \mathcal{E}_{\mathsf{DAP}}$, the advantage of $\mathcal{A}_i$ in winning $\mathcal{G}_{real}$ is $\mathcal{E}_{real} = \mathcal{E}_{\mathsf{DAP}}$. If the DAP scheme satisfies *Transaction Indistinguishability*, $\mathcal{E}_{real} \leqslant \mathsf{negl}(\lambda)$. Therefore, the DAPCR scheme satisfies *Indistinguishability* if the DAP scheme, the zk-SNARK scheme, the RegCom scheme and the hash function making up it are secure. □

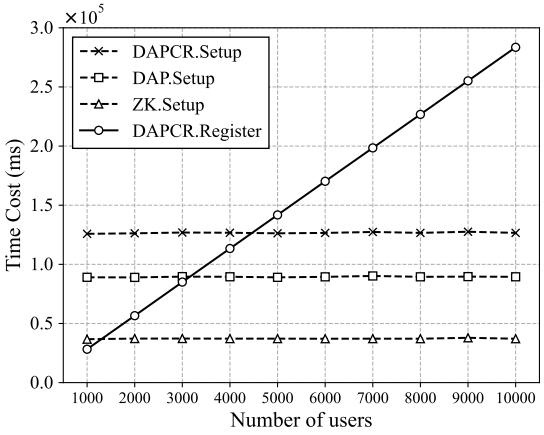

Fig. 4. Time cost in Preparation Phase

IX. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the DAPCR scheme proposed in Section VII.

Considering that DAP is a key component of DAPCR and Zether [6] is an efficient DAP scheme compatible with Ethereum, we construct the DAPCR scheme based on Zether and implement it on the Ethereum testnet *Rinkeby*. For compatibility with Ethereum, the validity of the regulatable transaction is verified by a smart contract instead of miners. It should be emphasized that DAPCR is theoretically based on a permissioned blockchain but *Rinkeby* is a public blockchain. We deploy the DAPCR scheme on *Rinkeby* to evaluate its performance in Verification Phase and temporarily ignore its security.

We carry out five experiments to evaluate the performance of DAPCR. *Experiment* I, *Experiment* II, *Experiment* III and *Experiment* V is carried out on a local device with a 8-core Intel(R) Core(TM) i7-10700 @ 2.90GHz CPU, 8-GB RAM and Ubuntu 20.04 LTS OS. *Experiment* IV is carried out on *Rinkeby*. Programming languages *Golang* and *Rust* and zero-knowledge proof tools *Bellman* and *Snarkjs* are used for these experiments. Moreover, the optimal ate pairing on a BN256 curve [30] is used in these experiments, which cannot offer 128-bits of security [31]. Considering that currently Ethereum is only compatible with the optimal ate pairing on the BN256 curve, we make a compromise on the security of DAPCR in experiments.

In *Experiment* I, we evaluate the time cost of initializing the DAPCR system and registering users in Preparation Phase, which is shown in Fig. 4. As described in Section VII, the algorithm DAPCR.Setup consists of algorithms DAP.Setup, RegCom.Setup and ZK.Setup, of which the time cost is not related to the number of users. The time cost of RegCom.Setup, about 35 ms, is much less than that of other algorithms. The time cost of registering users is linear to the number of users. Assuming there are $10,000$ users in the DAPCR system, the

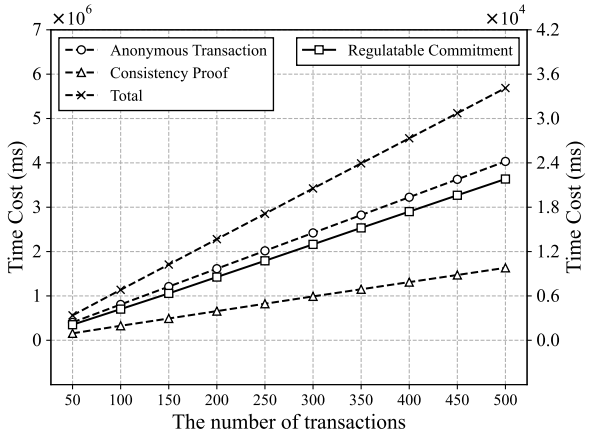

Fig. 5. Time cost in Transaction Phase

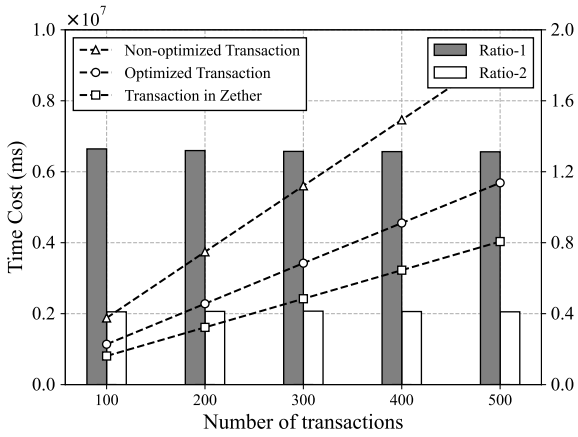

Fig. 7. Comparison of transactions in DAPCR and Zether

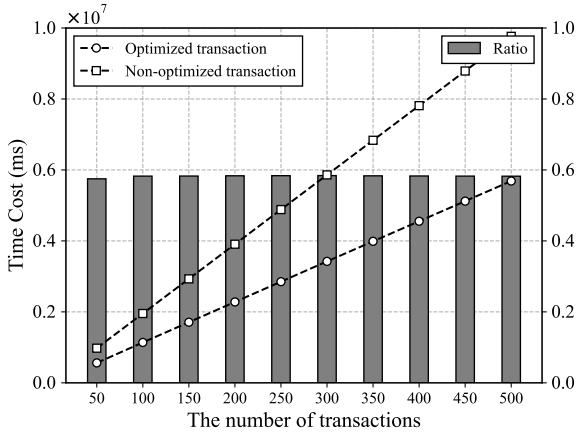

Fig. 6. Comparison of optimized and non-optimized transactions

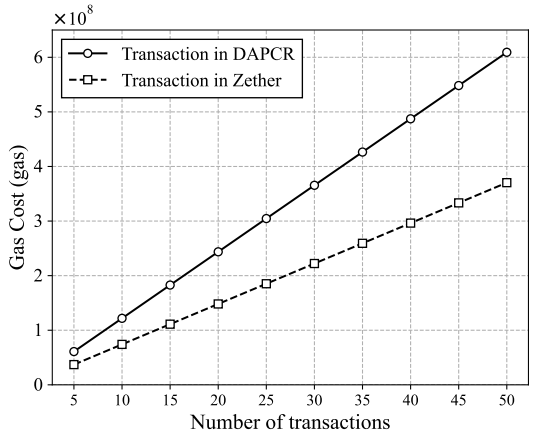

Fig. 8. The gas cost of transfer in DAPCR and Zether

time cost in Preparation Phase is about 4.1×10^5 ms, which indicates the effectiveness of DAPCR in Preparation Phase.

In *Experiment* II, we evaluate the time cost of generating the regulatable transaction, which is shown in Fig. 5. As described in Section IV, the regulatable transaction consists of an anonymous transaction, a consistency proof and a regulatable commitment. The time cost of generating regulatable transactions is linear to the number of transactions. It takes about 11 s to generate one regulatable transaction and less than 6×10^6 s to generate 500 transactions. The result of *Experiment* II indicates that DAPCR is effective in Transaction Phase.

In *Experiment* III, we compare the non-optimized and optimized schemes, which is shown in Fig. 6 and Fig. 7.

In Appendix A, an optimized DAPCR scheme is proposed to reduce the time cost of generating transactions. We compare the time cost of generating transactions in the optimized scheme and that in the non-optimized scheme, which is shown in Fig. 6. **Ratio** represents the ratio of the calculation time for optimized transactions to that for non-optimized transactions. With the optimized scheme, users can save more than 40% time for generating transactions.

In Fig. 7, we compare the transaction in DAPCR with that in Zether. A regulatable transaction consists of an anonymous transaction and additional fields, i.e. a regulatable commitment and a consistency proof. The anonymous transaction can be considered a transaction in Zether. **Ratio-1** represents the ratio of the calculation time for additional fields in the non-optimized scheme to that for anonymous transactions. **Ratio-2** represents the ratio of the calculation time for additional fields in the optimized scheme to that for anonymous transactions. **Ratio-1** is about 130% and **Ratio-2** is about 40%, which shows remarkable optimization effect. Result of *Experiment* III indicates that the optimization for DAPCR is significant and DAPCR is efficient in Transaction Phase.

In *Experiment* IV, we evaluate the gas cost of the regulatable transaction. Fig. 8 shows the gas cost of transactions in DAPCR and that in Zether. The gas cost in DAPCR is about 64% higher than that in Zether, which is acceptable. Since the block generation time is 15 s in *Rinkeby*, the time interval between a regulatable transaction being published and the transaction being sealed into a block is always about 15 s. The result of *Experiment* IV indicates that DAPCR is effective in Verification Period.

In *Experiment* V, we evaluate the time cost of Filter in

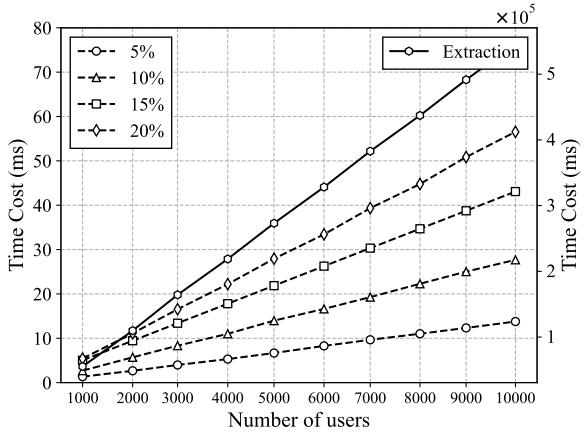

Fig. 9. Screening out illegal transactions

screening out illegal transactions, which is shown in Fig. 9. We use **Extraction** to denote the time cost of extracting regulation-labels from transactions, which is linear to the number of transactions. If there are $10,000$ transactions in the DAPCR system, **Extraction** is less than 5.5×10^5 ms. Assuming that transactions from Bob account for 5% (10%, 15%, 20%) of all transactions, the time cost of aggregating amount labels is shown in Fig. 9. Compared to **Extraction**, the time cost of aggregating amount labels is negligible. The result of *Experiment* V indicates that DAPCR is effective in Supervision Phase.

X. Conclusion

In this paper, we propose a decentralized anonymous payment scheme with collaborative regulation, which achieves generality, restricted regulation and aggregability of transactions. We also present an efficient and generic construction of DAPCR which can be implemented based on a DAP scheme. Security analysis and performance analysis show that DAPCR is secure and effective. The most important contribution in this paper is the restricted regulation for anonymous payments. Filter and Supervisor must cooperate to obtain traders' addresses and payment amounts from illegal transactions. Neither Filter nor Supervisor can get privacy from transactions without the assistance of the other one. Possible further work is reducing the calculation overhead for generating the regulatable transaction and the gas cost for verifying the validity of transactions.

Appendix A
Optimization for DAPCR

In Section VII, we make use of zk-SNARK to generate the consistency proof for a regulatable commitment and an anonymous transaction. In this appendix, we present how to reduce the time cost of generating the consistency proof.

Reviewing the arithmetic circuit C in Section VI-B, the binary relation R constructed by C includes some *exponentiation relations*. For example, Y and g_2 are two public inputs to C and y is one of the private inputs to C. C forces the tuple (g_2, y, Y) satisfies that $Y = g_2^y$, which is considered an *exponentiation relation*. In an arithmetic circuit C, the subcircuit that constructs the *exponentiation relation* is quite complex. If we can reduce the number of *exponentiation relations* in the binary relation R, the calculation overhead will be reduced synchronously.

User $\mathcal{U}$ first computes the one-time key vk and the auxiliary field aux for the regulatable commitment com_R.

- $\mathsf{GenAux}(\vec{\mathcal{V}_{tp}}, \nu, x, y, pp) \rightarrow (vk, aux)$. $\mathcal{U}$ takes the vector $\vec{\mathcal{V}_{tp}}$ of transaction privacy, the secret key ν, two random numbers x and y, and the public parameter pp as input, and generates the one-time key vk and the auxiliary field aux. x and y are the random numbers used to generate com_R in the algorithm $\mathsf{RegCom.GenCom}$. $\mathcal{U}$ randomly samples $r \in \mathbb{Z}_q^*$, computes vk and $aux = \{aux_1, aux_2, aux_3, aux_4\}$.

$$\begin{aligned} vk &= g_2^r \\ aux_1 &= vk^{addr_S \cdot \nu} \\ aux_2 &= vk^{v \cdot \nu} \\ aux_3 &= vk^{x} \\ aux_4 &= vk^{y} \end{aligned}$$

If vk, aux and Y are output by algorithms GenAux and $\mathsf{RegCom.Generate}$ with correct inputs, they can be used to verify the consistency of com_R and $\mathbf{tx}$.

- $\mathsf{VerifyCom}(com_R, vk, aux, pp) \rightarrow 0/1$. If the following three equations are true, this algorithm outputs 1 which means com_R is consistent with $\mathbf{tx}$. Otherwise, the algorithm outputs 0.

$$\begin{aligned} e(com, vk) &= e(\alpha_1, aux_1) \cdot e(\alpha_2, aux_2) \cdot e(\alpha_3, aux_3) \\ e(\varphi, vk) &= e(\beta_1, aux_4) \cdot e(\beta_2, aux_2) \cdot e(\beta_3, aux_3) \\ e(\psi, vk) &= e(\gamma_1, aux_1) \cdot e(\gamma_2, aux_4) \cdot e(\gamma_3, aux_3) \end{aligned}$$

Specifically, vk, aux and Y are valid, if they satisfy the following six conditions.

1. vk and aux are output by the algorithm GenAux with $(\vec{\mathcal{V}_{tp}}, \nu, x, y, pp)$ as input.
2. ν is the private key of $\mathcal{U}$.
3. x and y are random numbers used to generate com_R in the algorithm $\mathsf{RegCom.GenCom}$.
4. pp is the public parameter.
5. $Y = g_2^y$.
6. $\vec{\mathcal{V}_{tp}}$ is the vector of transaction privacy.

Then we make use of zk-SNARK to prove that vk, aux and Y satisfy the first five conditions. An arithmetic circuit C^* is built and R^* is the binary relation constructed by C^*. The public inputs to C^* are as follows.

- one-time key vk
- Y in the regulatable commitment com_R
- hash h of $\vec{\mathcal{V}_{tp}}$ and y
- a set of certificates $\mathbf{ce}$ in which the certificate ce of $\mathcal{U}$ is hidden
- hash h_i of aux_i for $i \in \{1, 2, 3, 4\}$
- $k = aux_1^{h_1} \cdot aux_2^{h_2} \cdot aux_3^{h_3} \cdot aux_4^{h_4}$

The private inputs to C^* are as follows.

- vector $\vec{\mathcal{V}_{tp}}$ of transaction privacy

- random numbers x and y used to generate com_R
- secret key ν of $\mathcal{U}$

C^* imposes the following constraints on public inputs and private inputs.

1. $t = addr_S \cdot \nu \cdot h_1 + v \cdot \nu \cdot h_2 + x \cdot h_3 + y \cdot h_4$
2. $k = vk^t$
3. $Y = g_2^y$
4. $\vec{\mathcal{V}_{tp}}||y$ is the preimage of hash h, i.e. $h = \mathsf{Hash}(\vec{\mathcal{V}_{tp}}||y)$.
5. $addr_S||\nu$ is the preimage of ce, i.e. $ce = \mathsf{Hash}(addr_S||\nu)$, and $ce \in \mathbf{ce}$.

In addition, we can construct the arithmetic circuit C' in Section VI-B to prove that vk, aux and Y satisfy the last condition.

Finally, Preparation Phase, Transaction Phase and Verification Phase in the DAPCR scheme should be modified to the following version.

I-Preparation Phase

- In Preparation Phase, $\mathcal{I}$ initializes this system and $\mathcal{S}$ registers users in the system.

Initialization Module $\mathcal{I}$:

- **Setup**
  - Inputs: security parameter λ
  - Outputs:
    1. public parameter pp
    2. extraction key ek
  - $\mathcal{I}$ executes the following steps to generate the public parameter pp and the extraction key ek:
    1. compute $\mathsf{DAP.Setup}(1^\lambda) \rightarrow pp_{tx}$
    2. compute $\mathsf{RegCom.Setup}(1^\lambda) \rightarrow (pp_{rc}, ek)$
    3. build the circuit C^* introduced above and the circuit C' in Section VI-B
    3. compute $\mathsf{ZK.KeyGen}(1^\lambda, C^*) \rightarrow (pk^*_{zk}, vk^*_{zk})$
    4. compute $\mathsf{ZK.KeyGen}(1^\lambda, C') \rightarrow (pk'_{zk}, vk'_{zk})$
    5. set $pp = \{pp_{tx}, pp_{rc}, pk^*_{zk}, vk^*_{zk}, pk'_{zk}, vk'_{zk}\}$
  - $\mathcal{I}$ publishes pp on the blockchain and sends ek to $\mathcal{F}$ over a secure channel.

Supervisor $\mathcal{S}$:

- **Register:**
  - Inputs:
    1. address $addr$
    2. public parameter pp
  - Outputs:
    1. secret key ν
    2. pseudonym nym
    3. upper limit max
    4. limit-label lab_{max}
    5. certificate ce
  - $\mathcal{S}$ executes $\mathsf{RegCom.GenKey}$ to register $\mathcal{U}$:
    1. compute $\mathsf{RegCom.GenKey}(addr, pp_{rc}) \rightarrow \nu$
    2. set the upper limit of total payment volume max for $\mathcal{U}$
    3. compute $nym = e(\beta_1, g_2)^{addr \cdot \nu}$
    4. compute $lab_{max} = e(\gamma_2, g_2)^{max \cdot \nu}$
    5. compute $ce = \mathsf{Hash}(addr||\nu)$
  - $\mathcal{S}$ adds $(addr, \nu, max, nym, lab_{max}, ce)$ to the list $\mathcal{L}_0$, which is initially empty. ce is published on the blockchain with $\mathcal{S}$'s signature. (ν, max) is sent to $\mathcal{U}$ over a secure channel. (nym, lab_{max}) is sent to $\mathcal{F}$ over a secure channel. (After receiving (nym, lab_{max}) from $\mathcal{S}$, $\mathcal{F}$ adds it to the list $\mathcal{L}_1$ which is also initially empty.)

II-Transaction Phase

- In Transaction Phase, $\mathcal{U}$ generates the regulatable transaction $\mathbf{rtx}$ and publishes it on the blockchain.

User $\mathcal{U}$:

- **Generate:**
  - Inputs:
    1. vector $\vec{\mathcal{V}_{tp}} = (addr_S, addr_R, v)$ of transaction privacy
    2. secret key ν of $\mathcal{U}$
    3. private key sk of $\mathcal{U}$
    4. a set of certificates $\mathbf{ce}$ in which ce of $\mathcal{U}$ is hidden
    5. pub and pri used to generate $\mathbf{tx}$ in $\mathsf{DAP.GenTx}$
    6. public parameter pp
    7. random numbers x and y used in $\mathsf{RegCom.GenCom}$
  - Outputs: a regulatable transaction $\mathbf{rtx}$
  - $\mathcal{U}$ executes the following steps to generate the regulatable transaction $\mathbf{rtx}$:
    1. comupte $\mathsf{DAP.GenTx}(addr_S, addr_R, v, sk, pub, pri, pp_{tx}) \rightarrow \mathbf{tx}$
    2. compute $\mathsf{RegCom.GenCom}(\vec{\mathcal{V}_{tp}}, \nu, pp_{rc}) \rightarrow com_R$
    3. compute $h = \mathsf{Hash}(\vec{\mathcal{V}_{tp}}||y)$
    4. compute $\mathsf{GenAux}(\vec{\mathcal{V}_{tp}}, \nu, x, y, pp) \rightarrow (vk, aux)$
    5. compute $h_i = \mathsf{Hash}(aux_i)$ for $i \in \{1, 2, 3, 4\}$
    6. compute $k = aux_1^{h_1} \cdot aux_2^{h_2} \cdot aux_3^{h_3} \cdot aux_4^{h_4}$
    7. set $\phi^* = \{vk, Y, h, \mathbf{ce}, h_1, h_2, h_3, h_4, k\}$
    8. set $\varpi^* = \{\vec{\mathcal{V}_{tp}}, x, y, \nu\}$
    9. set $\phi' = \{\mathbf{tx}, h, pp_{tx}, pub\}$
    10. set $\varpi' = \{\vec{\mathcal{V}_{tp}}, y, sk, pri\}$
    11. compute $\mathsf{ZK.Prove}(\phi^*, \varpi^*, pk^*_{zk}) \rightarrow \pi^*$
    12. compute $\mathsf{ZK.Prove}(\phi', \varpi', pk'_{zk}) \rightarrow \pi'$
    13. set $\pi_C = \{\pi^*, \pi', h, \mathbf{ce}, vk, aux\}$ and $\mathbf{rtx} = \{\mathbf{tx}, com_R, \pi_C\}$
  - $\mathcal{U}$ publishes $\mathbf{rtx}$ on the blockchain.

III-Verification Phase

- In Verification Phase, miners verify the validity of regulatable transactions and seal valid transactions into blocks.

Miner $\mathcal{M}$:

- **Verify:**
  - Inputs
    1. regulatable transaction $\mathbf{rtx} = \{\mathbf{tx}, com_R, \pi_C\}$ where $\pi_C = \{\pi^*, \pi', h, \mathbf{ce}, vk, aux\}$
    2. pub used to generate $\mathbf{tx}$ in $\mathsf{DAP.GenTx}$
    3. public parameter pp
  - $\mathcal{M}$ executes the following steps to verify the validity of a regulatable transaction $\mathbf{rtx}$:
    1. compute $h_i = \mathsf{Hash}(aux_i)$ for $i \in \{1, 2, 3, 4\}$
    2. compute $k = aux_1^{h_1} \cdot aux_2^{h_2} \cdot aux_3^{h_3} \cdot aux_4^{h_4}$
    3. set $\phi^* = \{vk, Y, h, \mathbf{ce}, h_1, h_2, h_3, h_4, k\}$
    4. set $\phi' = \{\mathbf{tx}, h, pp_{tx}, pub\}$
    5. compute $\mathsf{DAP.VerifyTx}(\mathbf{tx}, pp_{tx}) \rightarrow b_1$
    6. compute $\mathsf{ZK.Verify}(\phi^*, \pi^*, vk^*_{zk}) \rightarrow b_2$
    7. compute $\mathsf{ZK.Verify}(\phi', \pi', vk'_{zk}) \rightarrow b_3$
    8. $\mathsf{VerifyCom}(com_R, vk, aux, pp) \rightarrow b_4$
    9. if $b_1 \cdot b_2 \cdot b_3 \cdot b_4 = 1$, $\mathbf{tx}$ is valid and com_R is consistent with $\mathbf{tx}$, which means $\mathbf{rtx}$ is valid. Otherwise, $\mathbf{rtx}$ is invalid.
  - $\mathcal{M}$ seals valid transactions into blocks.

IV-Supervision Phase

- Supervision Phase is same as that in Section VII.

REFERENCES

[1] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International journal of web and grid services*, vol. 14, no. 4, pp. 352–375, 2018.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[3] G. Wood *et al.*, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

[4] S. Noether, A. Mackenzie *et al.*, "Ring confidential transactions," *Ledger*, vol. 1, pp. 1–18, 2016.

[5] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *2014 IEEE symposium on security and privacy*. IEEE, 2014, pp. 459–474.

[6] B. Bünz, S. Agrawal, M. Zamani, and D. Boneh, "Zether: Towards privacy in a smart contract world," in *International Conference on Financial Cryptography and Data Security*. Springer, 2020, pp. 423–443.

[7] S. Allen, S. Čapkun, I. Eyal, G. Fanti, B. A. Ford, J. Grimmelmann, A. Juels, K. Kostiainen, S. Meiklejohn, A. Miller *et al.*, "Design choices for central bank digital currency: Policy and technical considerations," National Bureau of Economic Research, Tech. Rep., 2020.

[8] Z. Wang, Q. Pei, X. Liui, L. Ma, H. Li, and S. Yu, "Daps: A decentralized anonymous payment scheme with supervision," in *International Conference on Algorithms and Architectures for Parallel Processing*. Springer, 2019, pp. 537–550.

[9] C. Lin, D. He, X. Huang, M. K. Khan, and K.-K. R. Choo, "Dcap: A secure and efficient decentralized conditional anonymous payment system based on blockchain," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2440–2452, 2020.

[10] C. Garman, M. Green, and I. Miers, "Accountable privacy for decentralized anonymous payments," in *International conference on financial cryptography and data security*. Springer, 2016, pp. 81–98.

[11] P. Chatzigiannis and F. Baldimtsi, "Miniledger: compact-sized anonymous and auditable distributed payments," in *European Symposium on Research in Computer Security*. Springer, 2021, pp. 407–429.

[12] N. Narula, W. Vasquez, and M. Virza, "{zkLedger}:{Privacy-Preserving} auditing for distributed ledgers," in *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, 2018, pp. 65–80.

[13] D. Ron and A. Shamir, "Quantitative analysis of the full bitcoin transaction graph," in *International Conference on Financial Cryptography and Data Security*. Springer, 2013, pp. 6–24.

[14] P. Koshy, D. Koshy, and P. McDaniel, "An analysis of anonymity in bitcoin using p2p network traffic," in *International Conference on Financial Cryptography and Data Security*. Springer, 2014, pp. 469–485.

[15] D. Vandervort, "Challenges and opportunities associated with a bitcoin-based transaction rating system," in *International Conference on Financial Cryptography and Data Security*. Springer, 2014, pp. 33–42.

[16] E. Duffield and D. Diaz, "Dash: A privacycentric cryptocurrency," 2015.

[17] Z. Guan, Z. Wan, Y. Yang, Y. Zhou, and B. Huang, "Blockmaze: An efficient privacy-preserving account-model blockchain based on zk-snarks," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[18] K. Wüst, K. Kostiainen, V. Čapkun, and S. Čapkun, "Prcash: fast, private and regulated transactions for digital currencies," in *International Conference on Financial Cryptography and Data Security*. Springer, 2019, pp. 158–178.

[19] A. Tomescu, A. Bhat, B. Applebaum, I. Abraham, G. Gueta, B. Pinkas, and A. Yanai, "Utt: Decentralized ecash with accountable privacy," *Cryptology ePrint Archive*, 2022.

[20] K. Wüst, K. Kostiainen, N. Delius, and S. Capkun, "Platypus: A central bank digital currency with unlinkable transactions and privacy preserving regulation," *Cryptology ePrint Archive*, 2021.

[21] A. Kiayias, M. Kohlweiss, and A. Sarencheh, "Peredi: Privacy-enhanced, regulated and distributed central bank digital currencies," in *The 29th ACM Conference on Computer and Communications Security*, 2022.

[22] J. Groth, "On the size of pairing-based non-interactive arguments," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2016, pp. 305–326.

[23] R. Gennaro, C. Gentry, B. Parno, and M. Raykova, "Quadratic span programs and succinct nizks without pcps," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2013, pp. 626–645.

[24] J. Groth and M. Maller, "Snarky signatures: Minimal signatures of knowledge from simulation-extractable snarks," in *Annual International Cryptology Conference*. Springer, 2017, pp. 581–612.

[25] M. Abe, J. Groth, K. Haralambiev, and M. Ohkubo, "Optimal structure-preserving signatures in asymmetric bilinear groups," in *Annual Cryptology Conference*. Springer, 2011, pp. 649–666.

[26] S. D. Galbraith, K. G. Paterson, and N. P. Smart, "Pairings for cryptographers," *Discrete Applied Mathematics*, vol. 156, no. 16, pp. 3113–3121, 2008.

[27] R. Cramer and V. Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack," in *Annual international cryptology conference*. Springer, 1998, pp. 13–25.

[28] Q. Dang *et al.*, *Recommendation for applications using approved hash algorithms*. US Department of Commerce, National Institute of Standards and Technology ..., 2008.

[29] S. D. Galbraith, P. Wang, and F. Zhang, "Computing elliptic curve discrete logarithms with improved baby-step giant-step algorithm," *Advances in Mathematics of Communications*, vol. 11, no. 3, p. 453, 2017.

[30] M. Naehrig, R. Niederhagen, and P. Schwabe, "New software speed records for cryptographic pairings," in *International Conference on Cryptology and Information Security in Latin America*. Springer, 2010, pp. 109–123.

[31] T. Kim and R. Barbulescu, "Extended tower number field sieve: A new complexity for the medium prime case," in *Annual international cryptology conference*. Springer, 2016, pp. 543–571.