Evaluating CryptoNote-Style blockchains

Runchao Han[1], Jiangshan Yu[2], Joseph Liu[2], and Peng Zhang[3]

[1] The University of Manchester, UK,
runchao.han@student.manchester.ac.uk
[2] Monash University, Australia
{jiangshan.yu, joseph.liu}@monash.edu.au
[3] Shenzhen University, China
zhangp@szu.edu.cn

Abstract. To hide user identity, blockchain-based cryptocurrencies utilize public key based coin addresses to represent users. However, the user identity can still be identified by linking the coin addresses to the IP address of a user, through network traffic analysis.
Ring Signature based protocols, such as CryptoNote and RingCT, have been designed to anonymize the payers of a transaction, and deployed in leading cryptocurrencies like Bytecoin and Monero. This paper provides a comprehensive evaluation on the performance of Bytecoin and Monero, at both the protocol level and the system level. In particular, our evaluation includes theoretical complexity analysis of the protocols and practical performance analysis of the Bytecoin and Monero implementation. In addition, we also provide an analysis on the existing Bytecoin and Monero transactions, based on the public blockchain data. Our results identify the execution bottleneck and space overhead of generating and verifying transactions, which may encourage the design of more efficient protocols. We also provide insights based on our analysis on the performance of specific cryptographic algorithms, static analysis of the ring size distribution, of the input size distribution and output size distribution, and of the transaction size distribution.

Keywords: Cryptocurrency, Blockchain, Ring Signature

1 Introduction

Cryptocurrencies have been very prevalent since the seminal Bitcoin system [9], which targets at democratizing the currency by a decentralized P2P network without governance. However, Bitcoin transactions are accessible for anyone with plaintext senders, receivers and amounts. Although the senders and receivers are cryptographically generated coin addresses, the coin addresses can still be linked to the identity of the real owner via traffic analysis.

In particular, with Bitcoin, a transaction is signed by the transaction sender, broadcasted to peers and verified by peers [9]. The transaction senders and receivers are represented by the explicit addresses generated from the public keys which is irreversible and deterministic. Each transaction is signed by the sender's

private key and verified by the public key, e.g. ECDSA in Bitcoin [9]. However, because the Bitcoin address is uniquely determined by the corresponding public key and both addresses and public keys are public, the individuals behind the Bitcoin network are traceable. For example, quantitative analyses towards the whole Bitcoin blockchain [12, 7] potentially reveal most Bitcoin participants.

CryptoNote [13] has been proposed to improve the anonymity of Bitcoin. In particular, it uses a modified version of traceable ring signatures [3], called One-time Ring Signature, to hide both the payer and payee of a transaction. However, CryptoNote cannot hide the amount of a transaction. Monero[4] proposed Ring Confidential Transactions [10] (RingCT), to further hide the amount by using Pedersen Commitment [11].

This paper aims at providing an understanding on the performance of the above two systems. We evaluate the performance both theoretically and experimentally. We first informally evaluate the algorithms in terms of their security, complexity, and parallelism. Then, we evaluate the systems through experiments.

In particular, we use Bytecoin v2.1.2[5] as the reference CryptoNote implementation, which is a CryptoNote-based and actively maintained cryptocurrency. It has a market cap of more than 432 Million USD to date, and is ranked 25th in the cryptocurrency market cap[6]. We use Monero v0.12.3.0[7] as the reference RingCT implementation, which has a market cap of about 1.9 Billion USD, and is ranked as the 13th in the cryptocurrency market cap[8].

Our analysis includes the performance of the specific cryptographic algorithms (such as time of creating/verifying a transaction with different inputs and outputs), static analysis of the ring size distribution, of the input size distribution and output size distribution, and of the transaction size distribution.

To evaluate the most recent status of the Bytecoin blockchain and Monero blockchain, we crawled more than 200,000 Bytecoin transactions, all Monero V6 transactions (from height 1400000 to 1539500) with the mandatory ring size 5, and all Monero V7 transactions (with the mandatory ring size 7) up to July 28th, 2018 (from height 1539500 to 1626649). Our results give several insights on the two blockchains. Our result shows that while providing a better privacy guarantee, Monero transaction is more time-consuming to create and to verify a transaction. We also observe that with Bytecoin, the average ring size is approximately 3, and the mandatory minimum ring size is 1 (no mixins) in Bytecoin. So it might be vulnerable to "zero-mixin" attacks [4, 8]. With Monero, the mandatory minimum ring size has been changed a few times in its earlier versions. Our analysis shows that for Monero V6 where the mandatory minimum ring size is 5, the average used ring size is also 5. Then, when the mandatory ring

[4] `https://getmonero.org/`.

[5] `https://github.com/amjuarez/bytecoin/tree/frozen-master`.

[6] `https://coinmarketcap.com/currencies/bytecoin-bcn/`. Data fetched on 7th August 2018.

[7] `https://github.com/monero-project/monero/`.

[8] `https://coinmarketcap.com/currencies/monero/`. Data fetched on 7th August 2018.

size is changed to 7 in Monero V7, the mean ring size in Monero is approximately 8. Thus, compared to transactions in Bytecoin, transactions in Monero have a much larger ring in average. This might indicate that Monero users concern more on privacy than Bytecoin users, so they intend to use system with better privacy guarantee and bigger rings.

For the number of inputs and outputs of a single transaction, compared to Monero, Bytecoin users intend to include more inputs and outputs in a single transaction. As for averages, each Bytecoin transactions include approximately 11 inputs and 12 outputs on average, while the average inputs and outputs of a transaction are only 2 and 3 for Monero, respectively. For the number of inputs and outputs, Monero transactions have an upper bound (by practise rather than by pre-defined rules) of 100 inputs and 40 outputs in a single transaction, whereas the upper bounds in Bytecoin are about 10 times as much.

2 Primitives

Ring signature was proposed to hide the real signer in a way that given a signed message, a third party only knows that someone in a particular group of people created the signature, but does not know who is the signer. It provides two anonymity properties [3]:

- Signature Unlinkability: Given two arbitrary signatures σ_a and σ_b, it is computationally infeasible to check if σ_a and σ_b are signed by the same signer
- Signature Untraceability: Given an arbitrary signature σ_a, it is computationally infeasible to determine which public key in the ring is the true signer

Ring signatures cannot be used directly to achieve anonymity of the blockchain transactions, due to a possible double spending attack. In particular, since a third party cannot identify who is the real signer, an attacker can spend the same coin as many times as the size of the group.

To prevent double spending attacks, the ring signature schemes applied to cryptocurrencies must be linkable to eliminate multiple uses of the money. Both linkable ring signature and traceable ring signature can be used to achieve these requirements.

It should be noted that the linkability of ring signature does not imply the transaction linkability. Instead, the transaction utilizes multiple cryptographic techniques including the Linkable Ring Signature to achieve the transaction unlinkability and untraceability, which will be discussed later.

One-time Ring Signature in CryptoNote CryptoNote utilizes a modified version of *Traceable Ring Signature* [3], called One-time Ring Signature. In One-time Ring Signature, a public key P_π and a Key Image I are derived from a private key x. The private key x and its key image P_π are used to prove that the signer knows at least one pair of public and private keys, while I aims at preventing against the creation of multiple signatures using the same key. Thus, it prevents the double spending attack. The detailed process of One-time Ring Signature is shown in Fig. 1.

1. **GEN**
   (a) Alice randomly picks a private key $x \in \mathbb{Z}_q$
   (b) Alice computes the public key $P_\pi = xG$ and the Key Image $I = xH_p(P_\pi)$
2. **SIG**
   (a) Alice randomly selects a list L of $n-1$ public keys, and let $0 \leq \pi \leq n$ be Alice's secret index
   (b) Alice randomly selects $q_1, ..., q_n$ and $w_1, ..., w_n \in \mathbb{Z}_q$
   (c) Alice computes c by

$$L_i = \begin{cases} q_i G, & \text{if } i = \pi \\ q_i G + w_i P_i, & \text{if } i \neq \pi \end{cases}$$

$$R_i = \begin{cases} q_i H_p(P_i), & \text{if } i = \pi \\ q_i H_p(P_i), & \text{if } i \neq \pi \end{cases}$$

   Then $c = H_s(m, L_1, ..., L_n, R_1, ..., R_n)$
   (d) Alice computes the signature $\sigma(m)$ by

$$c_i = \begin{cases} w_i, & \text{if } i \neq \pi \\ c - \sum c_i \mod q, & \text{if } i = \pi \end{cases}$$

$$r_i = \begin{cases} q_i, & \text{if } i \neq \pi \\ q_\pi - c_\pi x \mod q, & \text{if } i = \pi \end{cases}$$

   Then $\sigma(m) = (I, c_1, ...c_n, ..., r_1, ..., r_n)$
3. **VER**:
   (a) Bob recovers $L'_1, ..., L'_n$ and $R'_1, ..., R'_n$ by $\begin{cases} L'_i = r_i G + c_i P_i \\ R'_i = r_i H_p(P_i) + c_i I \end{cases}$
   (b) Bob checks if

$$\sum c_i \stackrel{?}{=} H_s(m, L'_0, ..., R'_0, ..., R'_m) \mod q$$

4. **LNK**: Bob checks if I has been used in previous signatures (where previous Key Images are stored separately)

Fig. 1. Signing and verification process of the One-time Ring Signature

Multilayered Linkable Spontaneous Anonymous Group (MLSAG) Signature in RingCT RingCT is based on linkable ring signature, of which the security model is shown in Fig. 2. RingCT defines the Multilayered Linkable Spontaneous Anonymous Group (MLSAG) Signature which extends the Linkable Spontaneous Anonymous Group Signature (LSAG) [5]. Each individual holds a vector of key pairs rather than only one key pair in order to hide the transaction amount by Pedersen Commitment, which will be discussed later. The detailed process of MLSAG is shown in Fig. 3.

Excluding the key vector, the One-time Ring Signature security model is essentially the same as the MLSAG Signature referred to Fig. 2.

3 Protocol-level Comparisons

This section compares the performance-related metrics between CryptoNote and RingCT at the protocol-level, including the core ring signature algorithm and the approaches of achieving anonymity.

1. **GEN**: Generating the private key k_π and the corresponding public key K_π
2. **SIG**: Signing the message m with L, a set of n public keys $K_1, ..., K_\pi, ..., K_n$ and k_π. The signature output is $\sigma(m)$
3. **VER**: Verifying $\sigma(m)$ with an arbitrary public key K_i in L, with an output *valid* or *invalid*
4. **LNK**: Checking if there is a signature using the same set of public keys L, with an output *linked* or *unlinked*

Fig. 2. Linkable Ring Signature.

3.1 Algorithm Analysis

We first evaluate the the security, complexity and parallelism of the signature schemes in the context of protocol specifications [9].

Security Both the One-time Ring Signature and the MLSAG Signature are unforgeable, untraceable and linkable. The unforgeability is a basic requirement for signature algorithms, and the untraceability is key to the blockchain transaction untraceability, while the linkability is exploited to combat the double-spending. Due to the usage of hash functions in both signature schemes, both security proofs are based on the Random Oracle (RO) Model [13] [10]. In the context of CryptoNote and RingCT specification, both signature schemes are based on the elliptic curve Ed25519 [1]. Therefore, the security assumption of both schemes is the Elliptic Curve Discrete Logarithm Problem (ECDLP). A tabulated remark on the security is shown in Table 1.

Table 1. The security analysis on One-time Ring Signature and MLSAG Signature

Signature	Proof	Group	Hardness	Forgeable	Linkable	Traceable
One-time Ring Signature	RO	Ed25519	ECDLP	✗	✓	✗
MLSAG	RO	Ed25519	ECDLP	✗	✓	✗

Complexity and Parallelism Compared to the One-time Ring Signature, MLSAG uses key vectors in its input. We denote the ring size as n and the key vector size as m.

The One-time Ring Signature signing includes computing vectors of L_i, R_i, c_i and r_i where $1 \leq i \leq n$, so the time and space complexity are both $O(n)$. Meanwhile, the One-time Ring Signature verification includes the inverse computation of L_i and R_i, so the time and space complexity are identical to the signing. On the other hand, MLSAG involves similar operations on $m \times n$ keys, so the time and space complexity for signing and verifying MLSAG signatures are all $O(mn)$.

Computations of L_i, R_i, c_i and r_i in the One-time Ring Signature are parallelizable, as no data dependency exists. However, the MLSAG Signature scheme

[9] CryptoNote Signature specification: `https://cryptonote.org/cns/cns002.txt`.

1. **GEN**
   (a) Alice randomly picks a private key vector $x_{\pi,1}, ..., x_{\pi,m}$
   (b) Alice computes each corresponding public key $P_{\pi,j} = x_{\pi,j}G$ and Key Image $I_j = x_{\pi,j}H_p(P_{\pi,j})$ where $j \in (1, 2, ..., m)$
2. **SIG**
   (a) Alice randomly selects a list L of $(n-1)$ public key vectors $P_{i,j}$ where $i \in (1, 2, ..., n) \wedge i \neq \pi$ and $j \in (1, 2, ..., m)$. Let $0 \leq \pi \leq n$ be Alice's secret index
   (b) $\alpha_j = rand() \in \mathbb{Z}_q$ for all $j \in (1, 2, ..., m)$
   (c) $r_{i,j} = rand() \in \mathbb{Z}_q$ for all $i \in (1, 2, ..., n) \wedge i \neq \pi$ and $j \in (1, 2, ..., m)$
   (d) Alice computes $c_{\pi+1}$ by

$$c_{\pi+1} = H_n(m, \alpha_1 G, \alpha_1 H_p(P_{\pi,1}), ..., \alpha_n G, \alpha_n H_p(P_{\pi,m}))$$

   (e) For $i = \pi + 1, ..., n$ and $1, ..., \pi - 1$, replacing $n + 1$ by 1, Alice computes:

$$\begin{aligned} c_{i+1} =& H_n(m, r_{i,1}G + c_i P_{i,1}, r_{i,1}H_p(P_{i,1}) + c_i I_1 \\ & , ..., r_{i,m}G + c_i P_{i,m}, r_{i,m}H_p(P_{i,m}) + c_i I_m) \end{aligned}$$

   (f) Alice computes

$$r_{\pi,j} = \alpha_j - x_{\pi,j}c_\pi \mod q$$

   and the signature $\sigma(m)$ is

$$\sigma(m) = (c_1, r_{1,1}, ..., r_{n,1}, ..., r_{n,m}, I_1, ..., I_m)$$

3. **VER**
   (a) For $1, ..., n$, replacing $n + 1$ by 1, Bob computes:

$$\begin{aligned} c'_{i+1} =& H_n(m, r_{i,1}G + c_i P_{i,1}, r_{i,1}H_p(P_{i,1}) + c_i I_1 \\ & , ..., r_{i,m}G + c_i P_{i,m}, r_{i,m}H_p(P_{i,m}) + c_i I_m) \end{aligned}$$

   (b) If $c'_1 = c_1$, then the signature is valid.
4. **LNK**: Bob checks if the Key Image vector I_j has been used in previous signatures (where previous Key Images are stored separately)

Fig. 3. Signing and verification process of the MLSAG Signature.

can only be parallelized by vectorizing key vector operations, while operations across n key vectors are not parallelizable due to the iterative data dependency.

As the One-time Ring Signature contains $(c_1, ..., c_n)$ and $(r_1, .., r_n)$, the size of One-time Ring Signature is $O(n)$. Meanwhile, the MLSAG signature involves all $r_{i,j}$ values where $1 \leq i \leq n$ and $1 \leq j \leq m$, the size of MLSAG signature is $O(mn)$. A tabulated remark on the complexity and parallelism is shown in Table 2.

Table 2. The complexity analysis on One-time Ring Signature and MLSAG Signature

Signature	Sign		Verify		Signature size	Parallelism
	Time	Space	Time	Space		
One-time Ring Signature	O(n)	O(n)	O(n)	O(n)	O(n)	Across n
MLSAG	O(mn)	O(mn)	O(mn)	O(mn)	O(mn)	Across m

3.2 CryptoNote and RingCT Transactions

After comparing the ring signature schemes, we turn to compare the transaction generation and verification between CryptoNote and RingCT.

The cryptocurrency system is basically a currency system, which can be regarded as a ledger recording transactions time-wise. A conventional transaction consists of the sender, the receiver, the amount of money and the signature signed by the sender. Both CryptoNote and RingCT hide the sender address and the receiver address, and RingCT further hides the amount. Similar to the ring signature, the transaction anonymity includes the unlinkability and the untraceability, but with different definitions. Informally,

- Transaction Unlinkability: Given two arbitrary transactions TX_a and TX_b, it is impossible to prove that they were sent to the same person.
- Transaction Untraceability: Given a transaction input, the real output being redeemed in it should be anonymous among a set of other outputs.

The transaction unlinkability is achieved by One-time Public Key, while the transaction untraceability is achieved by the ring signature schemes above. We start from comparing the transaction formats, then we analyse the untraceability and unlinkability provided by different systems.

Transaction Formats We start from analyzing the transaction formats of CryptoNote and RingCT.

As a generalization of conventional transactions, a CryptoNote or RingCT transaction consists of multiple inputs and multiple outputs. Basically, an input is a spendable deposit in the payer account, while an output is an amount of money that is transferred to the payee. The sum of inputs should equal to the sum of outputs in a single transaction. Due to the space limitation, we refer readers to the original paper for a detailed presentation of creating a CryptoNote transaction and a RingCT transaction.

While the inputs and outputs are simialr in CryptoNote and RingCT, the amount of the transferred money is masked in RingCT by using Pedersen Commitment, and each masked amount is with a commitment and the range proof by using Borromean Signature [6]. Moreover, the ring used in RingCT combines amounts besides public keys. Our following analysis will show that the range proof and the ring signature mechanism in RingCT contributes the most overhead, which is a sacrifice for a better privacy, i.e., also hiding the amount.

Unlinkability by One-time Public Key The One-time Public Key mechanism is the same in CryptoNote and RingCT, which is shown in Fig. 4. The design rationale is simple: A temporary public key is generated with random components and the receiver public key which can only be recognized by the receiver and the corresponding temporary private key can only be recovered by the receiver so that the money is spendable for by receiver.

1. Alice generates the transaction
   (a) Alice chooses a random $r \in \mathbb{Z}_q$
   (b) Alice computes a one-time public key $P = H_s(rA)G + B$
   (c) Alice packs a transaction TX including P and $R = rG$
2. Bob finds TX by scanning the blockchain or by the Payment Proof secretly sent from Alice by other communication approaches
3. Bob judges if the recipient of TX is himself
   (a) Bob computes $P' = H_s(aR)G + B$
   (b) If $P' = P$, the receiver of TX is Bob
4. Bob recovers the one-time private key $p = H_s(aR) + b$, where $P = pG$. Therefore, Bob can spend the money in TX

Fig. 4. One-time public key scheme

Alice chooses a random $r \in \mathbb{Z}_q$ and mixes r with Bob's public key A and B to produce the One-time Public Key P, then the corresponding transaction TX is committed to the blockchain if verified by the term leader.

In the meantime, Bob finds TX by scanning the blockchain or by the Payment Proof (which will be described later) secretly sent from Alice by other communication approaches. Bob tries to find out if the receiver of TX is himself or not.

To prove this, Bob recovers P again, but by his private key a rather than the public key A which exploited the Elliptic Curve scalar multiplication homomorphism. Bob compares the recovered P' to P, and claims the money ownership if $P' = P$.

Furthermore, Bob needs to prove this ownership to peers without revealing his public key A and B. The One-time Public Key P has an associated private key p which is only recoverable for Bob. As Bob exclusively knows a and b, p can be computed without conducting the computationally infeasible Elliptic Curve scalar divisions. With the exclusively owned One-time Private Key p, Bob can prove the money ownership by digital signatures which is verifiable by anyone.

The One-time Public Key approach indicates that Bob should verify the One-time Public Keys of all new transactions appended to the blockchain, which is similar to claiming the ownership anonymously. However, as a single verification is a fairly time-consuming cryptographic process, the claiming process introduces huge overhead. Monero leverages the overhead by the Payment Proof (also called Transaction Key) which is generated by Alice and unicasted to Bob with other approaches secretly. The Payment Proof is generated from the transaction information cryptogaphically, by which Bob can easily identify his transaction on the blockchain[10].

Untraceability with Double Spending Resistance by Key Image and One-time Ring Signature in CryptoNote While the unlinkability is achieved by the One-time Public Key, the untraceability is achieved by the One-time Ring Signature mentioned in Section 2. The process is shown in Fig. 5, which essentially wraps the One-time Ring Signature in Fig. 1 and further fits the signature scheme into the transaction creation.

[10] `https://getmonero.org/resources/user-guides/prove-payment.html`.

1. **GEN**: Alice generates the private key x, public key P and Key Image I
   (a) Alice chooses a random private key $x \in \mathbb{Z}_q$
   (b) Alice computes a one-time public key $P = xG$
   (c) Alice computes the Key Image $I = xH_p(P)$
2. **SIG**: Alice signs the transaction TX with One-time Ring Signature
   (a) Alice selects a random set of $n-1$ public keys P_i
   (b) Alice generates the One-time Ring Signature $\sigma(TX)$ with P_i, x, and I for the transactions TX
3. **VER**: Bob verifies the One-time Ring Signature signature $\sigma(TX)$
4. **LNK**: Bob checks if I was used in previous signatures

Fig. 5. CryptoNote scheme.

Firstly, Alice generates a random private key $x \in \mathbb{Z}_q$ and the corresponding public key P and Key Image I. Secondly, Alice grabs a random set of public keys to form a ring and produce the One-time Ring Signature for the transaction TX, in which the inputs store I as the masked sender address. After that TX along with the signature $\sigma(TX)$ is sent to Bob the verifier. Bob verifies the signature with the routine in Fig. 1, unpacks TX to recover I, then checks if I was used in previous signatures. If TX is valid and I is new, TX is treated as valid and broadcasted to more peers by Bob.

The One-time Public Key and One-time Ring Signature are both modular, so easy to fit into a single system in a mutually exclusive manner. However, CryptoNote only masks the sender and receiver, while the amount is visible.

Hiding the Sender and the Amount by Combining MLSAG, Pedersen Commitment and Range Proof in RingCT The solution, RingCT, mixes the Pedersen Commitment into the ring signature in order to mask the amounts. However, this modification on CryptoNote introduces the Range Proof problem which contributes to significant overhead.

The proposed RingCT scheme is shown in Fig. 6, which integrates the MLSAG scheme in Fig. 3. Instead of "one user one public key", each user has a vector of m key pairs to be compatible with the number of outputs m. Each output amount is replaced by the commitment value which is generated randomly with constraints, and the commitment values are involved in the ring used by the MLSAG Signature. In addition, the message to be signed is a series of commit values rather than the transaction itself.

The Range Proof is utilized in order to determine the range of unmasked amounts. As the amounts are masked and the Elliptic Curve Group is cyclic, a recovered value may have multiple possible values. Therefore, a Range Proof with commitments is conducted again for each output. To make the proof verifiable, a simpler Ring Signature scheme called Borromean Ring Signature is utilized to sign the commitments. However, the Range Proof takes much space in practise. A commitment value takes at least 8 Bytes according to the Ed25519 curve specification, which is 64 bits. For each bit a commitment value is generated in order to form the Ring Signature. In other words, 64 commitment values and a Ring Signature with 128 keys are responsible for only one output amount.

Table 3. The computational complexity of transaction-related operations for CryptoNote and RingCT

		Generate		Verify (+ Link)	
		Time	Space	Time	Space
CryptoNote	One-time address	O(out)	O(out)	O(out)	O(out)
	One-time Ring Signature	O(n)	O(n)	O(n)	O(n)
RingCT	One-time address	O(out)	O(out)	O(out)	O(out)
	Pedersen Commitment	O(in+out)	O(in+out)		
	Fake Transaction Generaion	O(n*(in+out))	O(n*(in+out))		
	MLSAG	O(n*out)	O(n*out)	O(n*out)	O(n*out)
	Range Proof	O(out*amount)	O(out*amount)	O(out*amount)	O(out*amount)

Evaluating Complexity and Transaction Size We evaluated the computational complexity and the theoretical transaction size against the number of inputs, the number of outputs and the ring size for transaction-related operations. The results are shown in Table. 3.

Computational Complexity Obviously, an One-time Public Key is generated for each output in a transaction, so the generation of verification of One-time Public Key is $O(out)$ for both CryptoNote and RingCT.

The Pedersen Commitment generation consists of finding random masked values and masks for each inputs and outputs, so the time and space complexity is $O(in + out)$.

Similarly, generating $(n-1)$ fake key vectors and commitment values involves $n-1$ One-time Public Key generations, $n-1$ fake amount and Pedersen Commitment generations. Therefore, the time and space complexity is $n-1$ times of $O(in + out)$, which is $O(n * (in + out))$.

The verifications of Pedersen Commitments, fake transaction generations, and the MLSAG Signature are all accomplished by the MLSAG Signature verification, as those three processes are deeply coupled. Therefore, we only consider the computational complexity of the MLSAG Signature. As the key matrix is $n \times (out + 1)$, the MLSAG Signature generation and verification are all with the time and space complexity $O(n * out)$ according to Section 3.1.

As for the Range Proof, for each output in a transaction, a Range Proof is conducted, including the commitment and the Borromean Signature. The commitment value generation for an output is with the time and space complexity $O(out * amount)$ apparently. Based on the Borromean Signature process the time and space complexity is the same. Therefore, for a single transaction with *out* outputs, the time and space complexity for the Range Proof generation and verification is $O(out * amount)$.

Transaction Size The previous space complexity analysis implies that a RingCT transaction takes significantly more memory space than a CryptoNote transaction with the same number of inputs, outputs and the ring size. Therefore, we focus on the space overhead in RingCT.

1. **GEN**
   (a) Generate m random public keys $P_\pi^1, ..., P_\pi^m$, where m is the number of outputs
   (b) Generate masked amounts and the corresponding commitments by the Pedersen Commitment:
      i. Alice computes $H = to\_point(SHA3(G))$
      ii. Alice determines $x_1, ..., x_n$, $a_1, ..., a_n$, $y_1, ..., y_{y_m}$ and $b_1, ..., b_m$ which satisfy
      $$C_{in,i} = x_i G + a_i H, i \in [1, n]$$
      $$C_{out,j} = y_j G + b_j H, j \in [1, m]$$
      $$\sum_{i=1}^{n} C_{in,i} - \sum_{j=1}^{m} C_{out,j} = zG$$
      where n is the number of inputs
      iii. Alice denotes
      $$C_\pi^j = C_{out,j}, \text{where } j \in [1, m]$$
      iv. Alice randomly chooses $q - 1$ vectors of public keys $P_1^1, ..., P_1^m$ to $P_q^1, ..., P_q^m$ (excluding $p_\pi^1, ..., p_\pi^m$) and the corresponding commitments $C_1^1, ..., C_q^m$, where q is the ring size
      v. Alice computes a set of output values and the corresponding commitments $(Q_i, C_{i,out})$ s.t. $\sum_{j=1}^{m} C_\pi^j - \sum_{i=1}^{q} C_{i,out} = 0$.
2. **SIG**
   (a) Alice signs the message msg
   $$msg = H_p(Q_1, C_{1,out}, ..., Q_q, C_{q,out})$$
   on the ring
   $$\begin{bmatrix} (P_1^1, C_1^1) \dots (P_1^m, C_1^m) \ (\sum_{j=1}^{m} C_1^j - \sum_{i=1}^{q} C_{i,out}) \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ (P_q^1, C_q^1) \dots (P_q^m, C_q^m) \ (\sum_{j=1}^{m} C_q^j - \sum_{i=1}^{q} C_{i,out}) \end{bmatrix}$$
   using the MLSAG Signature in Fig. 3, and the output signature is $\sigma(msg)$
   (b) Alice produces the Range Proof for each unmasked output amount b
      i. Alice converts each b to the 64-bit binary format
      $$b = \sum_{i=0}^{63} b_i 2^i$$
      with $b_i \in \{0, 1\}$
      ii. Alice computes commitments $C_i = a_i G + b_i 2^i H$ for $i \in [0, 63]$
      iii. Alice generates the Borromean Ring Signature on the ring
      $$\{C_i, C_i - 2^i H\}$$
      where $i \in [0, 63]$
3. **VER**: Bob checks if $\sigma(msg)$ is valid with the **VER** routine in Fig. 3
4. **LNK**: Bob checks if $P_\pi^1, ..., P_\pi^m$ are used in previous signatures with the **LNK** routine in Fig. 3

Fig. 6. Hiding the receiver and the amount by MLSAG, Pedersen Commitment and Range Proof

Ignoring the unimportant information in a transaction like the version number, the major parts include inputs, outputs and the extra data related to the verification. Apparently, the input size increases linearly with the number of inputs increases, which is the same as the output size. Meanwhile, the Ring Signature and the Range Proof contribute to the most overhead in RingCT. The following analysis assumes that the Elliptic Curve points are stored in the compressed format of 4 Bytes [11].

Firstly, the MLSAG scheme involves a $n \times (out + 1)$ key matrix, and the Ring Signature size is directly related to the number of rings according to Fig. 3.

[11] `https://crypto.stackexchange.com/questions/8914/ecdsa-compressed-public-key-point-back-to-uncompressed-public-key-point`.

Meanwhile, the One-time Ring Signature only takes n signatures. We quantify the relationship between the signature size and the number of inputs, outputs and public keys based on Fig. 1 and Fig. 3:

$$\begin{aligned} sizeof(RingSignature_{CN}) &= 4 * size(I, c_1, ..., c_n, r_1, ..., r_n) \\ &= 4 * (2n + 1)(Bytes) \\ sizeof(RingSignature_{RCT}) &= 4 * [size(ring) + in + 1] \\ &= 4 * [n * (in + 1) + in + 1]) \\ &= 4 * (n + 1)(in + 1)(Bytes) \end{aligned}$$

Secondly, the Range Proof takes much space. Although the Range Proof size increases linearly with the number of outputs increase, the coefficient is quite big in practical. We assume $amount = 64$ (which is identical with the CryptoNote and RingCT specifications), and quantify the relationship between the Range Proof size and the number of outputs:

$$\begin{aligned} sizeof(RangeProof_{RCT}) &= 4 * out * [size(signature) + size(maskedValues)] \\ &= 4 * out * [(size(ring) + 1) + amount] \\ &= 4 * out * [2 * amount + 1 + amount] \\ &= 4 * out * [64 * 2 + 1 + 64]) \\ &= 772 * out(Bytes) \end{aligned}$$

Table 4 concludes the results above. In conclusion, the approach of RingCT to hide the amount is expensive on the memory space. In the cryptocurrency context, big transactions lead to higher transaction fees [12] and lower transaction throughputs due to the block size limitation [2]. Therefore, leveraging the transaction size while keeping anonymous for cryptocurrencies is a crucial topic.

Table 4. The size of Ring Signatures and Range Proofs with different inputs, outputs and ring sizes

	Ring Signature	Range Proof
CryptoNote	(2n+1)*4	0
RingCT	(out+1)(n+1)*4	772*out

4 Performance and Security Comparisons

According to Section 3.2, the performance of privacy-related techniques introduces overhead. However, Section 3.2 only focuses on the theoretical analysis, which may be different from the real implementation.

[12] The Bitcoin transaction fee specification: `https://en.bitcoin.it/wiki/Transaction_fees`.

In this section, a detailed comparison between the CryptoNote protocol and the RingCT protocol is conducted, including the performance evaluation and the network status of the existing blockchain platforms based on CryptoNote and RingCT (We chose Bytecoin [13] as the CryptoNote reference implementation and Monero [14] as the RingCT reference implementation).

4.1 Experimental Methodologies

Evaluated Metrics While the privacy is enhanced, the computational and storage overhead is introduced based on our analysis in Section 3.2, which may lead to lower transaction throughput and higher transaction fees. To evaluate the performance of privacy-related techniques from the practical perspective, the evaluation task is divided into two subtasks:

- Evaluating the performance of specific cryptographic processes
- Evaluating the blockchain network usage

The first subtask benchmarks the performance of privacy-related cryptographic processes, including:

- Time of constructing a transaction with different inputs and outputs
- Time of verifying a transaction (signature) with different inputs and outputs
- Transaction size with different inputs and outputs

Meanwhile, the blockchain network usage evaluation focus on the actual status of the running blockchains, which represents the true attitudes of network participants rather than the whitepapers. The evaluated metrics include:

- Ring size of signatures
- Transaction size
- The number of transaction inputs
- The number of transaction outs

Experimental Data The data sources include:

- Results of running official test cases[15][16] with customized configurations.

[13] CryptoNote implementation in Bytecoin: `https://github.com/bcndev/bytecoin/blob/d3dd3acf0a3113c9801589c6a512ef68a6eabed2/src/crypto/crypto-ops.h`.

[14] RingCT implementation in Monero: `https://github.com/monero-project/monero/blob/3fde902394946281665531abd742c64bdb23be25/src/ringct/rctOps.cpp`.

[15] Bytecoin test cases: `https://github.com/amjuarez/bytecoin/tree/frozen-master/tests`.

[16] Monero test cases: `https://github.com/monero-project/monero/tree/master/tests`.

– The transaction data which can be queried on the blockchain explorers[17][18].

As for performance-related metrics, we chose $1, 2, 4, 6, 8, 16, 32, 64, 128$ as the ring size, the input number and the output number to obtain experimental results from existing test cases.

In the meantime, more than 200,000 most recent transactions are crawled from the Bytecoin and Monero blockchain explorers in order to conduct the network usage analysis.

Experimental Environment

Hardware The experiments for performance-related metrics were conducted on a laptop with a 64-bit Intel Core i7-6700HQ processor with 8 cores running at 2.60 GHz, 24 GB RAM, one Intel SATA SSD with 210 GB, a Nvidia GeForce GTX 960m GPU with 4GB DRAM.

Software We chosed Bytecoin v2.1.2[19] as the reference CryptoNote implementation, which is a CryptoNote-based and actively maintained cryptocurrency without modifying the CryptoNote core protocol. Meanwhile, Monero v0.12.3.0[20] was regarded as the reference RingCT implementation, which is the first and the most prevalent RingCT-based cryptocurrency.

The selected blockchain platforms were compiled from the source code with the compiler GCC 5.4.0. The operating system is Ubuntu 18.04.

4.2 Performance of Critical Cryptographic Processes

Constructing Transactions

Results The time of constructing a transaction with different inputs and outputs is shown in Fig. 7. Constructing a Monero transaction is more time-consuming than constructing a Bytecoin transaction with the same inputs and outputs. Moreover, the number of outputs is the dominant factor of constructing a Monero transaction, while the number of inputs and the number of outputs have similar impacts of the Bytecoin transaction construction.

Analysis The results are expected and consistent to our protocol-level analysis.

As for Bytecoin, the time increases linearly with the increase of inputs and outputs. According to Section 3.2, the overhead introduced by the CryptoNote protocol are mainly the One-time Ring Signature which is linearly influenced by the ring size. The rest overhead increases linearly with the increase of the transaction size, which is linearly correlated with the number of inputs and

[17] `https://xmrchain.net/block/1618540`.
[18] `https://explorer.bytecoin.org/`.
[19] `https://github.com/amjuarez/bytecoin/tree/frozen-master`.
[20] `https://github.com/monero-project/monero/`.

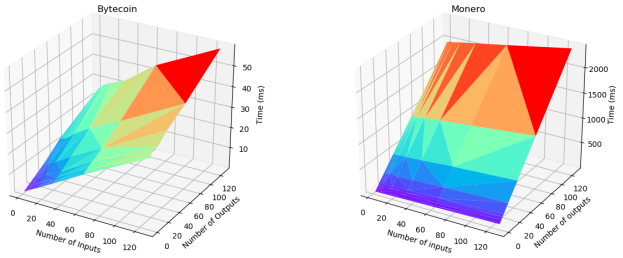

Fig. 7. Time of constructing a transaction with different inputs and outputs

outputs as well. Therefore, the constructing time increases with the number of inputs and outputs increases.

When it comes to Monero, the time is dominated by the number of outputs. According to Section 3.2, each output is attached with a Range Proof, and each Range Proof is attached with a Borromean Ring Signature. A Range Proof is considerably expensive, leading to a big overhead. Moreover, the size of a MLSAG Signature is linearly correlated to the number of outputs. Therefore, the number of outputs contributes to the most overhead so dominates the RingCT transaction construction time.

Verifying Transactions (Signatures)

Results The time of verifying a signature with different ring sizes are shown in Fig. 8. Note that the signature is on an empty transaction with only one input and one output. It is observed that verifying a signature in Bytecoin is faster than in Monero. Also, the consumed time increases linearly with the ring size increases for both Bytecoin and Monero. For example, with the average ring sizes (3 for Bytecoin and 8 for Monero, which will be discussed later), the verification time of the Bytecoin signature is approximately 1ms, while for Monero the verification time is 20ms.

Analysis Because the number of inputs and outputs is fixed, the only variable is the ring size. The One-time Ring Signature of CryptoNote has the time complexity $O(n)$, while the MLSAG Signature is $O(mn)$. In Monero context, $m = out$, and $out = 1$ in the test case, so $m = 1$ and the MLSAG Signature time complexity here is $O(n)$ as well. Therefore, the linear increase of the verification time is as expected. On the other hand, the reason why the MLSAG Signature verification time is longer than the One-time Ring Signature is because of the space overhead introduced by the iterative hashing process without parallelism for computing $c_1, ..., c_{n+1}$.

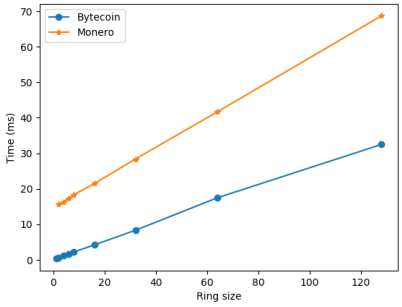

Fig. 8. Time of verifying a transaction (signature) with different ring sizes

4.3 Network Usage Analysis and Potential Threats

Ring Size

Results The results are represented as histograms with marked average ring sizes, shown in Fig. 11. The average ring size is approximately 3, and the mandatory minimum, ring size is 1 (no mixins) in Bytecoin.

With Bytecoin, the mandatory minimum ring size has not been changed. However, Monero has updated the mandatory minimum ring sizes several times in the history. Monero V6 (Helium Hydra) [21], which hard-forked Monero V5 at the block height 1400000, firstly forced RingCT transactions with the mandatory ring size of 5. Then Monero V7 (Lithium Luna) [22], which hard-forked Monero V6 at the block height 1539500, changed the mandatory ring size to 7. It is noted that RingCT was firstly introduced in Monero V5 (Wolfram Warptangent) [23], but was not mandatory for transactions. The mandatory ring size of Monero V5 is 5 as well. Before Monero V5 the RingCT was not deployed, and the mandatory ring size was even smaller, which is out of our topic.

We conducted ring size analysis on Monero V6 and V7, as we focus on RingCT transactions. With the mandatory ring size of 5, the average ring size is 5.65. After changing the mandatory ring size to 7, the average ring size turns to 7.59. In the meantime, Monero users intend to choose bigger ring sizes than Bytecoin users according to our statistics.

Analysis The ring size is directly correlated with the anonymity of senders and receivers. With more public keys mixed in a transaction, the identities of senders and receivers will be more ambiguous. Monero chose a bigger mandatory ring size to strongly guarantee the identity ambiguity, making the Monero transactions harder to trace. As a result, Monero users may concern more on privacy than Bytecoin users, so intend to use bigger rings.

[21] `https://github.com/monero-project/monero/releases/tag/v0.11.0.0`.
[22] `https://github.com/monero-project/monero/releases/tag/v0.12.0.0`.
[23] `https://github.com/monero-project/monero/releases/tag/v0.10.0`.

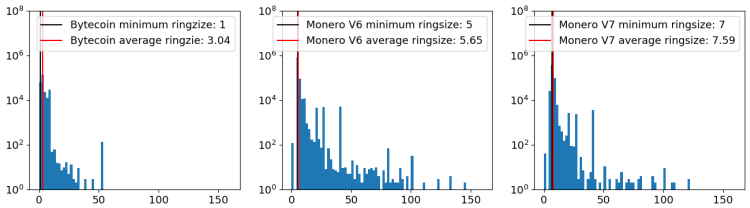

Fig. 9. The distribution and statistics of ring sizes

Inputs and Outputs

Results Similar to the ring size statistics, the inputs and outputs distributions are shown in Table 10, with marked averages. Compared to Monero, Bytecoin users intend to include more inputs and outputs in a single transaction. In average, each Bytecoin transactions include approximately 11 inputs and 12 outputs on average, while the average inputs and outputs are only 2 and 3 for Monero, respectively. Furthermore, Monero users never take more than 100 inputs or 40 outputs in a single transaction, but for Bytecoin the corresponding numbers are approximately 1000 and 300.

Analysis The reason that Monero users intend to take fewer inputs and outputs than Bytecoin users is mainly because of the high transaction fees introduced by bigger transaction sizes. According to the analysis in Section 3.2, hiding the output amount sacrifices the transaction size. Moreover, this sacrifice will be greater with more inputs or outputs. The transaction fee in Bytecoin and Monero is directly related to the transaction size [24]. Hiding the amount may not be critical for some Monero users compared to the high transaction fee. Therefore, Monero users intend to include fewer inputs and outputs.

In fact, high transaction fee in Monero is a concerning problem since the RingCT fork [25] [26]. Different solutions have been emerging, which will be discussed later.

Transaction Size

Results The size of a transaction has a direct impact on the transaction fee. We conducted the transaction size statistics like before, shown in Fig. 11. The transaction size is 3KB on average for Bytecoin, while 18KB for Monero. In

[24] Monero transaction fee calculator: `https://www.monero.how/monero-transaction-fee-calculator`.

[25] `https://www.reddit.com/r/Monero/comments/7h0i5e/why_is_the_fee_so_high_380/`.

[26] `https://www.reddit.com/r/Monero/comments/74flal/why_are_fees_so_high/`.

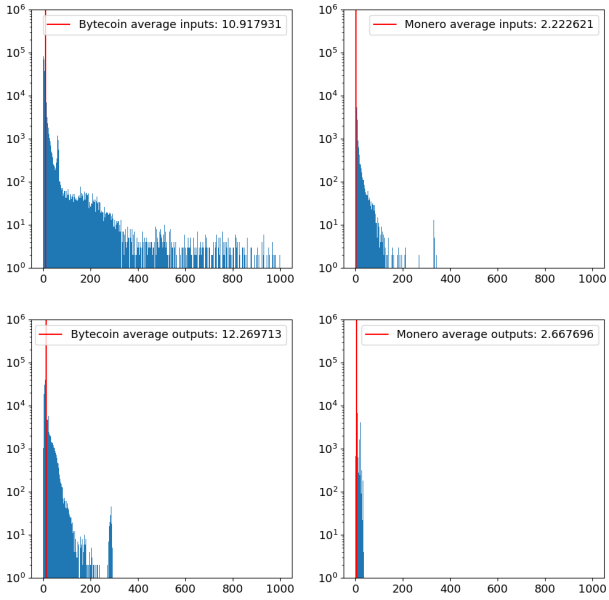

Fig. 10. The distribution and statistics of inputs and outputs

addition, Monero transactions are much bigger than Bytecoin transactions based on the distribution.

In addition, we correlated the transaction size with the number of inputs and outputs directly in Fig. 12. It should be noted that Fig. 12 omitted some unusual data on the blockchains. For example, a Monero transaction contains 2495 inputs, which is a really big number regarding to the RingCT transactions. Apparently, the transaction size of Monero is much bigger than of Bytecoin with the same number of inputs and outputs. Moreover, the number of outputs in Bytecoin is the dominant factor of the transaction size. As for Bytecoin, the relation between the transaction size and the number of inputs and outputs is fairly ambiguous.

Analysis Based on our protocol-level analysis in Section 3.2 and analysis on the ring size, inputs and outputs, the transaction size distribution is as expected: Hiding the transaction amount introduces space overhead which greatly affects the transaction size.

As for the relation between the transaction size and the number of inputs and outputs for Bytecoin, the transaction size is most influenced by the ring size rather than the number of inputs and outputs. In the meantime, due to the

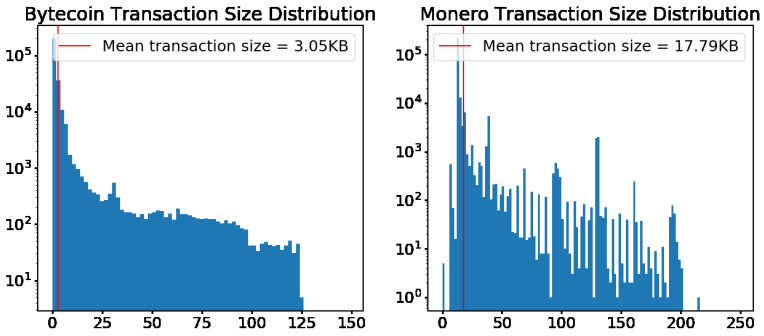

Fig. 11. The distribution and statistics of transaction sizes same scale for all gifures

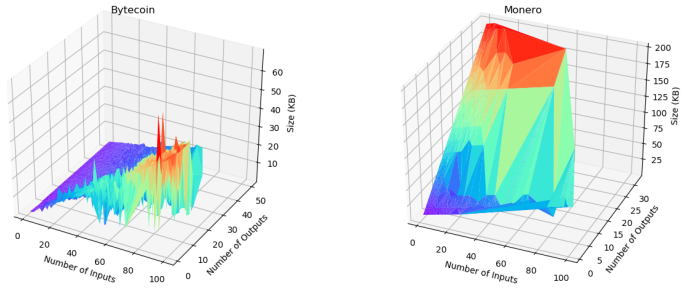

Fig. 12. Size of a transaction with different inputs and outputs

MLSAG Signature combining with the Pedersen Commitment, the outputs take the major part of a RingCT transaction, which proves the protocol-level analysis towards transactions in Section 3.2.

5 Conclusion

In this paper, we analysed the performance of two CryptoNote style blockchains, i.e., ByteCoin and Monero, from the protocol layer and the application perspective. Our protocol-level comparisons started from formalizing the core Ring Signature schemes and the processes of hiding senders, receivers and amounts in transactions. Then we compared the performance of the formalized anonymization processes, including the theoretical time and space complexity and the transaction sizes in depth. The protocol-level comparisons indicate that RingCT hides the amounts with significant space overhead, mainly from the MLSAG Signature and the Range Proof.

We experimented on benchmarking the aforementioned cryptographic processes and analyzing the real transaction data on these two blockchains. The benchmarking results proved our protocol-level analysis that the number of outputs dominate the performance of transaction generations and verifications for RingCT. Meanwhile, our network usage analysis based on the real blockchain data showed that Monero users intend to include fewer inputs and outputs but mix more public keys in a single transaction than Bytecoin users. The fewer inputs and outputs of Monero is because of the high transaction fees introduced by big transactions, implying that hiding the amounts is less concerning than the transaction fees for Monero users. More mixed public keys indicate that Monero users are actually concern more of the privacy than Bytecoin users, and the overhead of hiding the senders and receivers is acceptable.

Acknowledgement

This work was partially supported by the National Natural Science Foundation of China (61702342), the Science and Technology Innovation Projects of Shenzhen (JCYJ20170302151321095).

References

1. Bernstein, D.J., Duif, N., Lange, T., Schwabe, P., Yang, B.Y.: High-speed high-security signatures. Journal of Cryptographic Engineering **2**(2), 77–89 (2012)
2. Croman, K., Decker, C., Eyal, I., Gencer, A.E., Juels, A., Kosba, A., Miller, A., Saxena, P., Shi, E., Sirer, E.G., et al.: On scaling decentralized blockchains. In: FC. vol. 2016, pp. 106–125. Springer (2016)
3. Fujisaki, E., Suzuki, K.: Traceable ring signature. In: International Workshop on Public Key Cryptography. pp. 181–200. Springer (2007)
4. Kumar, A., Fischer, C., Tople, S., Saxena, P.: A traceability analysis of monero's blockchain. In: ESORICS. pp. 153–173 (2017)
5. Liu, J.K., Wei, V.K., Wong, D.S.: Linkable spontaneous anonymous group signature for ad hoc groups. In: Australasian Conference on Information Security and Privacy. pp. 325–335. Springer (2004)
6. Maxwell, G., Poelstra, A.: Borromean ring signatures (2015)
7. Moser, M.: Anonymity of bitcoin transactions (2013)
8. Möser, M., Soska, K., Heilman, E., Lee, K., Heffan, H., Srivastava, S., Hogan, K., Hennessey, J., Miller, A., Narayanan, A., Christin, N.: An empirical analysis of traceability in the monero blockchain. PoPETs **2018**(3), 143–163 (2018)
9. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008)
10. Noether, S., Mackenzie, A., et al.: Ring confidential transactions. Ledger **1**, 1–18 (2016)
11. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Annual International Cryptology Conference. pp. 129–140. Springer (1991)
12. Ron, D., Shamir, A.: Quantitative analysis of the full bitcoin transaction graph. In: International Conference on Financial Cryptography and Data Security. pp. 6–24. Springer (2013)
13. Van Saberhagen, N.: Cryptonote v 2.0 (2013)