Computer Networks xxx (xxxx) xxx

Contents lists available at ScienceDirect

Computer Networks

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

Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions

Umair Sarfraz [a], Masoom Alam [a,*], Sherali Zeadally [b], Abid Khan [a]

[a] Cybersec Lab Department of Computer Science, COMSATS University, Islamabad, Pakistan
[b] School of Information Science College of Communication and Information, University of Kentucky, Lexington

ARTICLE INFO

Article history:
Received 6 August 2018
Revised 2 November 2018
Accepted 14 November 2018
Available online xxx

Keywords:
IOTA
Anonymity
Internet Of Things (IoT)
Distributed ledger
Security

ABSTRACT

IOTA is a distributed ledger technology for the Internet-of-Things (IoT) industry. The protocol distinguishes itself from existing distributed ledgers by being formed on a directed acyclic graph. To enable micro-transactions for smart devices, it uses a scalable approach for network growth and transaction confirmations. Being a public distributed ledger, the transactions on the ledger are completely transparent hence opening up the possibilities for linking and identification attacks. Different promising privacy enhancing techniques have been proposed for improving anonymity in distributed ledgers. However, many of the proposed approaches provide security guarantees only against *Elliptic Curve Digital Signature (ECDSA)* schemes and thus become incompatible with the IOTA ledger because IOTA uses quantum resilient hash-based signatures. While centralized solutions can still work with IOTA ledger for enhancing privacy, they are still proprietary and prone to single point of failures. We propose a novel decentralized mixing protocol for the IOTA ledger that incorporates a combination of decryption mixnets and multi-signatures. Our technique does not require any (trusted or accountable) third party and it is completely compatible with the IOTA protocol. Analysis of our results for this technique shows that the security and privacy are guaranteed even in the presence of malicious entities in the system. Our technique provides strong privacy to the IOTA ledger and the degree of anonymity it adds, protects entities against identification and linking attacks.

1. Introduction

Distributed ledger is one of the most promising technologies of the new millennium even though much of its potential still remains undeveloped. Due to its inherent property of high redundancy, the technology offers a highly reliable layer of integrity and immutability. The technology emerged in 2008, when Satoshi Nakamoto proposed Bitcoin [1] as a fully decentralized electronic currency system. Digital currencies are still the most dominant applications built on distributed ledgers. However, the technology still faces several important issues such as scalability, privacy, environmental costs and lack of support for micro-transactions. To address these issues so that this technology becomes mainstream on a large scale (e.g., Internet of Things (IoT)), new ledgers have emerged that not only solve the issue of network scalability but also enable micro-transactions.

IOTA is a public distributed ledger that enables fee-less micro-transactions for the Internet of Things (IoT) devices [2]. Created in 2015, it solves the core issues of scalability and fees in traditional distributed ledgers. The market cap of circulating IOTA tokens amount to nearly $3 billion as of June 2018 and it is among the top ten cryptocurrencies by market cap [3]. One of the major differences of IOTA from existing distributed ledgers is the consensus structure namely, the *Tangle* [2], which is based on a Directed Acyclic Graph (DAG) rather than a blockchain. The protocol removes the need of *miners*[1] from the network. Instead, all network participants are equally responsible for the network consensus. Each time they make a transaction, they validate two previous transactions [2]. This simple yet scalable approach increases the confirmation rates on the network as the number of network contributors grows which will yield more approved transactions.

The ultimate vision of IOTA is to enable the machine economy for smart devices. The number of smart connected devices around the globe will drastically increase to 75.44 billion by 2025 [4] and the distributed integrity layer of the tangle can replace centralized cloud storages for the data these devices will produce. However, with the versatile nature of the tangle, value can also be exchanged

* Corresponding author.
E-mail address: masoom.alam@gmail.com (M. Alam).

[1] Special participants responsible for validating and approving transactions.

https://doi.org/10.1016/j.comnet.2018.11.019
1389-1286/

on top of it and digital money can be securely tracked in a tamper-proof way. IOTA tokens are stored at addresses which are cryptographic public identities generated from a random seed[2] of 81 trytes.[3] Transactions[4] associated with these public identities stay anonymous as long as addresses cannot be linked to their holders.

However recent works [5–7] have demonstrated how transactions graph analysis can link transactions and addresses on a public ledger and since the IOTA ledger is transparent and has a similar transaction scheme like Bitcoin's *Unspent Transaction Output (UTXO)* [8,9], taint and transaction analysis can disclose private information such as transaction history and total balance. [10]. In the Bitcoin whitepaper [1], Satoshi Nakamoto recommended using a new pair of addresses for every transaction, strictly for the purpose of maintaining on-ledger privacy. But even if a user, (say Alice), transfers funds to a fresh unlinkable address, it would still link the new address back to his/her identity. Mixing services allow a way to transfer funds to a fresh address in an unlinkable manner. If Alice sends her funds to a mixing service, then the service will in return transfer funds of some randomly chosen user to Alice's desired fresh address. An external observer, (say Bob), cannot distinguish a mixing transaction from a normal transaction on the ledger. Alice, participating in the mix can plausibly deny her participation in the mix, hence adding a desirable property for privacy. As of today, there is only one mixing service [11] for the IOTA ledger and is only supported for the *devnet aka testnet.*[5]

However, there are two fundamental issues with centralized mixers. First, the consumers of the mixing service need to trust the service providers not to steal their funds. Second, the mixing service providers can keep track how the funds were mixed in the first place and hence can be forced or incentivized to reveal this information. These issues have led to propositions and implementations of new decentralized techniques [12–17] that not only solve the issues with centralized mixing but also offer better security and anonymity. However, all of these decentralized mixing techniques are proposed and evaluated for the Bitcoin protocol and because IOTA uses *Winternitz One-Time Signatures (WOTS)* [18,19] instead of Bitcoin's *Elliptic Curve Digital Signature Algorithm (ECDSA)* [20,21], none of the techniques can be directly incorporated into the IOTA protocol. The main reason for the inapplicability of existing decentralized mixing techniques with the IOTA ledger is because none of these techniques provide security guarantees against hash-based signatures.

The remainder of the paper is structured as follows: We provide an in-depth comparison of our proposed solution with previous related works in Section 2. Section 3 provides background information on the IOTA ledger (addresses, signature scheme, multi-signatures, transactions and IOTA mixing), anonymity and verifiable shuffling. Section 4 highlights the problem definition and design requirements for decentralized mixing. Section 5 describes our proposed solution and system design. In Section 6, we analyze, evaluate and discuss the system properties of our proposed solution. Section 7 outlines a detailed case study of smart devices running our proposed solution for mixing and creating unlinkability in their transactions. Finally, Section 8 concludes the paper.

2. Related work

In this section, we discuss and analyze previously proposed privacy enhancing protocols to improve transactions anonymity and highlight their differences compared to our proposed solution, as shown in Table 1. We also highlight the main contributions of this research work.

2.1. Related works

Centralized mixing services [22–24] were initially developed for Bitcoin and also for IOTA [11] to improve anonymity by mixing tokens. However, these services provide no guarantee that the tokens will ever return. Most of the mixing services have been based on a fee model and face issues of availability. Additionally, the owners of centralized mixing services can keep track of the knowledge of the linkage between input and outputs addresses and could reveal this information. However, the solution of centralized mixing services guarantees strong anonymity because the mixing transactions are indistinguishable on the ledger for outside observers.

Merge Avoidance Mike Hearn, former Bitcoin core developer proposed a technique (called Merge Avoidance) for Bitcoin [25] to protect transaction privacy. Recently, the proposed approach was also discussed in the context of the IOTA ledger [10]. The inapplicability of merge avoidance is because of the overhead of transaction fees. While it can be adopted by IOTA (with no transaction fees), it does not improve the anonymity of the transactions themselves, rather it avoids privacy leaking situations for end users. Furthermore, wallets adopting this approach could affect the availability of the funds for a user which clearly defeats the notion of instant payments.

The *CoinJoin* [12] protocol was the first to propose group transactions. The proposal was a great step towards improving the disadvantage and risk of a centralized mixer stealing funds during a mix. However, during the group transaction, the participants learn about the mapping of input to output addresses of each other. While it formed the basis of many privacy enhancing protocols for Bitcoin, it cannot be applied to the IOTA ledger because of IOTA's use of hash-based signatures when participants can steal portions of private key to forge address signatures and can bail out of the group transaction.

CoinShuffle [13] improves CoinJoin [12] by introducing a way to mix tokens that no participant learns about the mapping of input and output addresses of each other. The protocol utilizes decryption mixnets for output address shuffling. The use of decryption mixnets protects each participant link of input to output address. This shuffling technique also forms the basis of the outputs shuffling of our proposed solution as discussed previously. However, CoinShuffle compatibility is only for ledgers such as Bitcoin that use *ECDSA* signatures for transactions and hence fails to be applicable to the IOTA protocol. Our proposed solution, addresses this issue.

MixCoin [16] introduces a way to hold dishonest participants accountable during a mix. However, it still depends on a centralized mixing service. While it adds anonymity to transactions and reaps the benefit of indistinguishable transactions for external observers, it still faces drawbacks of mixing delays and significant mixing fees. Most importantly, the compatibility is only restricted to Bitcoin-like ledgers because of the signature scheme.

CoinSwap [17] mitigates the risk of a centralized service stealing funds by employing a multi-signature contract with a payment service. This is still subject to the risk of funds being lost because the centralized service can refuse to sign the contract. This risk has been eliminated in the Bitcoin protocol by the use of *Hashed Timelock Contracts* [29] but due to the lack of possibility of smart contracts being deployed on the tangle directly, it becomes inapplicable for the IOTA ledger. Furthermore, it still does not provide anonymity against participants (sender, receiver and the payment service).

[2] Random 81 character string consisting of only A-Z and the number 9.

[3] A tryte consists of 3 trits $1, 0, 1$, so the maximum value is $3^3 = 27$. Supported tryte alphabets are [A Z, 9].

[4] A transfer of tokens between addresses.

[5] A dedicated network mimicking the IOTA protocol by allowing transactions with fake tokens.

Please cite this article as: U. Sarfraz, M. Alam and S. Zeadally et al., Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions, Computer Networks, https://doi.org/10.1016/j.comnet.2018.11.019

Table 1
Comparison of our proposed solution with related works considering the primitive design requirements. *Anonymity level* is computed for 1) mixing peers 2) c / n malicious peers and 3) outside observers. *Time complexity* is measured for m mixing peers with n inputs.

Approach	Security	Anonymity level	Fees	Time complexity	Comp. w/ IOTA
Centralized Mixers [11,22–24]	None	$0 >> n >> n$	Mix fees	–	√
CoinShuffle [13]	Group transaction	$n \; n-c \; n$	TX fees	$O(m*n)$	✗
Merge Avoidance [25]	None	$0 >> n >> n$	TX fees	–	√
CoinJoin [12]	Group transaction	$0 \; n-c \; n$	TX fees	$O(n)$	✗
MixCoin [16]	Accountability	$n \; n-c \; n$	Mix fees	$O(n*log(n))$	√
CoinSwap [17]	Hashed Timelocks	$n > n-c > n$	TX fees	–	✗
Stealth Addresses [26]	ECDH	$\infty \; \infty \; \infty$	–	$O(n^2)$	✗
CoinParty [15]	67% honest peers	$n > n-c > n$	TX fees	$O(m*n)$	✗
ZeroCash [27,28]	ZKPs	$\infty \; \infty \; \infty$	TX fees	$O(n^2)$	✗
Our Approach	Multi-signatures w/ group transaction	$n \; n-c > n$	None	$O(n^2)$	√

The *Stealth Addresses* [26] approach removes the requirement to generate a fresh address to receive new payments. While it provides strong anonymity guarantees on top of the ledger, it suffers several drawbacks. First, the recipient will be dependent on scanning the ledger for new transactions to be able to filter out his/her transaction because he/she cannot have any knowledge about the one-time used payment address in advance. Additionally, the use of stealth addresses is computationally expensive. Stealth addresses can also be identified due to the presence of an ephemeral key on a transaction.

CoinParty [15] improves mixing through group transactions by employing threshold ECDSA signatures. CoinParty provides the distinctive property of *deniability* to mixing participants. CoinParty cannot be applied to the IOTA ledger because of its compatibility to just ECDSA signature schemes, and it also suffers from the disadvantages of mixing delays and security guarantees against 67% honest participants.

ZeroCoin [28] is an extension to Bitcoin. Essentially it replaces Bitcoin's public transaction history by Zero Knowledge Proofs (ZKPs). ZeroCoin was among the first approaches to create unlinkability between transactions without inclusion of any third party. However, ZeroCoin is associated with high communication and computation overheads. It also requires a trial decryption of every cipher text after scanning the ledger and additional storage for the size of proofs (approximately 25 KB). Later, *ZeroCash* [27] was proposed to mitigate and reduce the storage and computation requirements for ZeroCoin. These protocols do enable a distributed layer where privacy and integrity can co-exist but they have clear disadvantages such as adaptability and compatibility because using these protocols require significant changes to existing blockchain operations.

2.2. Our research contributions

In contrast to related privacy preserving protocols, our proposed solution made several improvements. We summarize the main contributions of our proposed decentralized mixing protocol for the IOTA ledger as follows:

- It leverages the multi-signature scheme to ensure a secure commitment between participants. The use of multi-signature addresses protects against signature forgery and guarantees that even in the presence of malicious adversaries during mixing, no participant can reveal portions of his/her private key of the input address.
- It provides a direct alternative to the deployed IOTA mixing service because with our proposed solution, the mixing operation can be executed in a fully decentralized fashion. Moreover, due to the decentralization of the protocol, it does not require any mixing fees from any participant.
- It achieves the desired anonymity property of unlinkability because no participant or outside observer would be able to transform a link between input and output addresses.
- It does not require any change to the existing IOTA protocol. We evaluate a proof-of-concept implementation in a real-world scenario to demonstrate that, unlike other decentralized privacy enhancing solutions, it works well with the current IOTA protocol.

3. Background

Next, we briefly cover relevant background on anonymity, IOTA ledger as well as IOTA mixing and verifiable shuffling as the foundation elements of our proposed decentralized mixing protocol.

3.1. IOTA

Addresses IOTA protocol implements a deterministic address generation process that starts with the *seed*. Every generated address has a corresponding *key index*, *security* and a *private key*. Since IOTA uses a ternary numeral system, the address length is 81 *trytes*. IOTA address generation utilizes cryptographic sponge functions [30]. Fig. 1 describes how address generation works for IOTA. First *subseed* (seed + index) is hashed into a *private key*. The length of the private key is determined by the *security*. It should be noted that the process of private key generation is deterministic and hence it removes the need for storing private keys anywhere. After the *private key* is generated, it is hashed into 27 key fragments that are each hashed 26 times. In the last step, the hashed key fragments are hashed together to generate an *address* (public key). Due to IOTA's choice of one-time signature scheme, spending from an address multiple times drastically reduces the security of the funds at that address, hence making it more challenging to employ decentralized anonymity enhancing protocols in IOTA.

Signature scheme IOTA uses Winternitz One-time signature scheme. WOTS produces significantly smaller signatures compared to other hash-based digital signature schemes. Since IOTA uses a ternary numeral system, there are slight modifications made to the original WOTS scheme. The security of WOTS depends on the hashing scheme used. Thus WOTS is unforgeable if the hash function H used is collision resilient.

$$H : \{-1, 0, 1\}^* \rightarrow \{-1, 0, 1\}^n \tag{1}$$

The IOTA variant of WOTS initially generates a *subseed*.

$$subseed = H(seed + keyindex) \tag{2}$$

where $seed \in \{A - Z, 9\}$ and *key index* $\in \{0, 1, \ldots\}$. The private key pr_i is computed by successively hashing *subseed*. The length of the private key depends on the security level $s \in \{1, 2, 3\}$. It will be

Please cite this article as: U. Sarfraz, M. Alam and S. Zeadally et al., Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions, Computer Networks, https://doi.org/10.1016/j.comnet.2018.11.019

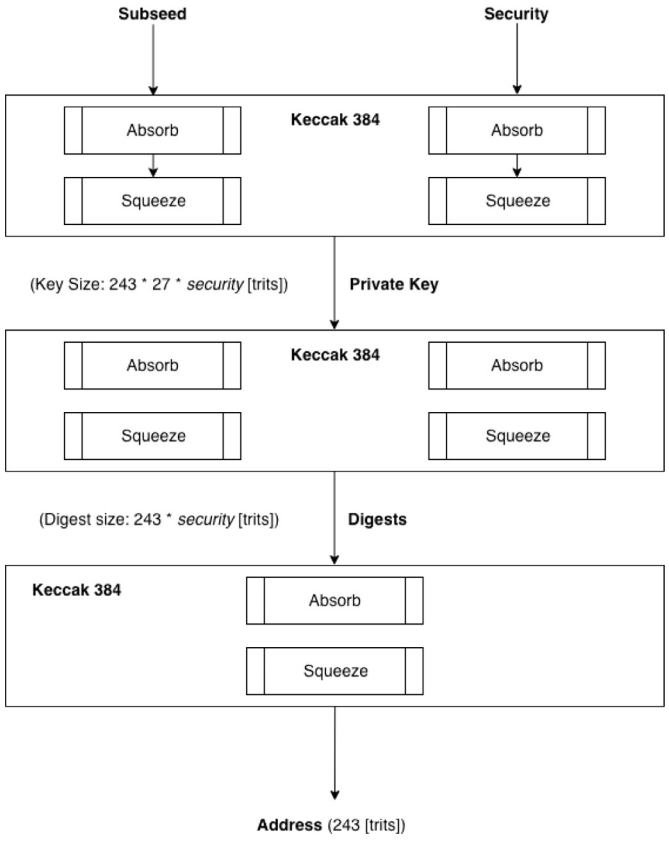

Fig. 1. Overview of the address generation process in IOTA.

hashed 27(=w)[6] times per security level, which results in a key length of s * 27 * 81 trytes, which is 2187, 4374, or 6561 *trytes*, respectively. Each 81 trytes hash generated is a *key fragment*. The private key pr_i is then used to generate the address pk_i by hashing each key fragment 26 times, and then hashing the results together into an 81 tryte address value.

$$pk_i = H^{3^w-1}(pr_i) \tag{3}$$

The final address pk_i therefore depends on s, which means that the same *subseed* can generate 3 different addresses, one for each security level. For signature generation, a digest $d = H(M)$ for a message M is computed. Then, we first take 27, 54, or 81 trytes of digest d depending on the security level s. This partial message hash is normalized such that the sum of all trytes equal to zero. The signature σ is computed by hashing private key d_i times.

$$\sigma_i = H^{d_i}(sk_i) \tag{4}$$

Multi-signatures Multi-signatures, in general is a digital signature scheme that allows a group of users rather than a single user to sign a document. Essentially, what multi-signature enables (in cryptocurrencies space) is to restrict a group of people to formally agree on spending. Ideally, all co-signers need to sign the transaction from a multi-signature address, but certain variations in the multi-signature scheme (M-N) [31] allow ownership to a few participants to spend from a multi-signature address. In order to create a multi-signature address all participants need to share their digests publicly, which are then concatenated to make a long single digest [32]. Then the same process (as shown in Fig. 1) is followed to create a multi-signature address of *81 trytes*.

[6] Winternitz parameter = 3, which determines the number of trits to be signed simultaneously.

Transactions A transfer of IOTA tokens between a set of IOTA addresses constitutes a transaction. To issue a transaction, a user needs to specify one or more input addresses $A_1,\ldots,A_n$ with a positive value z and output (receiving) addresses $O_1,\ldots,O_n$. There are also other information attributes embedded within an IOTA transaction [33]. IOTA also supports zero value transactions, which removes the need for specifying any input addresses. In order to spend from an input address during a transaction, a user needs to sign the transaction with the corresponding private key as a proof that the user owns the respective address. For a valid transaction to be stored and broadcasted on the IOTA ledger, it needs to validate the two previous valid transactions. The process of choosing two *tips*[7] involves doing a weighted random walk from the genesis towards the tips [34]. The validation requires doing proof of work [35]. Transactions that are part of the network become *sites* on the tangle graph.

IOTA Mixing One of the earliest and ubiquitous techniques proposed for improving anonymity on distributed ledgers is *centralized mixing*. The basic idea is: for users to remove the linkage between their input and output addresses from an outside observer, they exchange tokens of equal value with some other users. The most basic form of mixing is done in the presence of a trusted third party. A user interested in mixing his/her coins, sends his/her funds to the service, then the mix returns an equal value token to the output address provided by the user. Many mixing services have been deployed for the Bitcoin network [22–24] and while these have been incorporated into the IOTA ledger [11], they suffer from two major drawbacks: First, users have to trust a third party which in reality defeats the whole purpose of decentralization. Second, the mixing service can keep track of the input and output address mapping.

3.2. Anonymity

Anonymity is of paramount importance in distributed ledgers. Pfitzmann and Kohntopp define anonymity [36] as:

> "Anonymity is the state of being not identifiable within a set of subjects, the *anonymity set*".

To expand on the formal definition, a subject (initiator) is identifiable if an observer or any other subject is able to get information that can be linked to the initiator. Anonymity is certainly measurable and the use of entropy [37] lead to the foundation of formal measurement of anonymity [38,39]. The *degree of anonymity* considers the probability associated with each entity. Using the formal definition from [38]:

$$d = \frac{H(X)}{H_M} \tag{5}$$

$$H(X) := \sum_{i=1}^{N}\left[p_i.lg\left(\frac{1}{p_i}\right)\right] \tag{6}$$

$$H_M := H(X) \leftarrow lg(N) \tag{7}$$

$H(X)$ is the entropy of the network, N is the number of nodes (entities) active in the network, and p_i is the probability associated with the node i. H_M is the maximal entropy of the network and it occurs when there is uniform probability associated with each node $\left(\frac{1}{N}\right)$. The definition of the degree of anonymity shows that over the time an adversary can assign probabilities to each initiator as the original initiator of a message, based on the information leaking attributes of the system. A recent ledger analysis

[7] Valid transactions on the tangle.

[7] has shown that even through incomplete transaction information, anonymity of the users can be affected.

3.3. Verifiable shuffling

A shuffle is essentially a permutation and re-randomization of a set of cipher texts. Formally a verifiable shuffle of a set of cipher texts $\{c_1,\ldots,c_n\}$ is a new set of cipher texts $\{C_1,\ldots,C_n\}$ keeping the same plain texts in permuted order. The main application of shuffling is construction of mix-nets, a cryptographic system introduced by Chaum [40] for providing anonymity and unlinkability in communication. In the context of distributed ledgers, especially decentralized mixing techniques, verifiable shuffling techniques have been used [13,14] for shuffling the output addresses in an oblivious manner. This overcomes the drawback of a trusted third party being able to learn about the mapping of input and output addresses during a mix. Considering the security properties of a shuffle namely, *verifiability, unlinkability and robustness* [41], verifiable shuffling becomes a suitable candidate for improving anonymity and financial privacy in distributed ledgers.

3.4. Adversary model

Maximum efficiency cannot be achieved if an adversary is constantly hindering the communication between the participants. Constant disruption can lead to an inefficient execution of the protocol. Additionally, the lack of co-operation of any participant can make him/her a *passive adversary*. For *anonymity* and *correctness*, we assume external observers and malicious peers as passive adversaries because they can analyze the protocol to gain additional private information.

Our protocol does not depend on any trust assumptions from any peer. However, to achieve *anonymity* the protocol demands at least two honest participants for the mixing operation. Otherwise, oblivious mixing becomes impossible as the peers can easily learn about the mapping of input and output addresses.

4. Problem definition

Motivated by the recent work [10,11] on improving anonymity in IOTA transactions, we investigate how a set of entities can mix their IOTA tokens to maintain their financial privacy on the ledger without needing any trusted party. It is also worth pointing out that the use of WOT signature scheme for signing transactions in the IOTA protocol which is provably quantum-resilient, but once signed, it significantly reduces the security of the address because it exposes portions of the private key associated with the address.

Formally, a group of n peers with z number of IOTA tokens at public addresses $A_1,\ldots,A_n$ with corresponding private address keys $A_1^{'},\ldots,A_n^{'}$ want to mix the amount to destination addresses $B_1,\ldots,B_n$ in a way that (1) each peer securely and successfully receives z IOTA tokens on the corresponding output address, (2) except for the original input peer, no other peer learns about the correct mapping of input and output addresses and (3) no information about the corresponding private keys is revealed even in the presence of malicious peers. Fundamentally, for each peer i participating in the protocol signs off a transaction $A_i \xrightarrow{z} O_{a(i)}$ where a is a *verifiable* and *secret* shuffle over $\{1,\ldots, n\}$. Such a service should essentially have the following design goals:

- **Anonymity.** The mixing must be anonymous i.e. a malicious peer or an outside observer must not be able to create a link between A_i and O_i.
- **Correctness.** Any malicious participant should not be able to steal other participants' funds. No malicious participant should be able to steal the private key or portions of private keys from other participants in the protocol. Essentially, no malicious participant should be able to practically forge signatures of any other participant.
- **Compatibility.** The protocol should be fully compatible with the existing IOTA protocol and should not produce any invalid transactions.
- **No mixing fees.** The mixing operation should not introduce any additional fees.
- **Efficiency.** The protocol should scale to any number of participants and should not increase delays when the number of participants grows. In addition, it should not cause any additional overheads on the IOTA network.

Considering the existing solutions for improving transactions privacy on distributed ledgers, the initial solutions we identified were *centralized mixers* [22–24]. A similar centralized service is also developed specifically for the IOTA ledger [11]. These centralized techniques do not guarantee security against a malicious service. Apart from availability issues with these services, the mixing service can still keep track of the links between permuted addresses. Several improvements [12,16,17,25] have been proposed to address the issues associated with centralized mixes, but mixing entities still need to depend on a trusted party for the mix. Recently, decentralized mixing techniques [13–15] have been proposed. However, their applicability is restricted to only Bitcoin like distributed ledgers that use *Elliptic Curve Digital Signature (ECDSA)*. Some other techniques [26–28] with strong anonymity and security guarantees have also emerged, but they are incompatible with the IOTA ledger because they require significant changes to the core IOTA protocol. Thus, secure and anonymous mixing of IOTA tokens remains a challenge. In the following section, we propose a novel solution for IOTA mixing that utilizes the advantages of verifiable shuffling and multi-signatures to fulfill the stated requirements.

5. Protocol design

In this section, we describe our proposed solution. Our protocol takes a decentralized approach at IOTA mixing by removing the need for a centralized mixing service. The protocol takes into account the use of WOT signature scheme in IOTA protocol and ensures that during the mix no mixing peer[8] ends up losing their funds by revealing significant portions of their private key to malicious participants in the protocol. The protocol also ensures anonymity in a way that no malicious participant or an outside observer is able to make a link between any input and output address. Fig. 3 presents an overview of the protocol with three mixing peers. The protocol can be split into three stages, (1) settlement (Section *A*), (2) outputs shuffling (Section *B*) and (3) transaction (Section *C*). For cases where an adversary or several adversaries try to deviate from the protocol, there is an additional fall back phase. We describe the protocol design in depth in the following section. The analysis and in depth discussion of security, anonymity and other system properties are presented in section VI.

5.1. Settlement

The aim of settlement phase is to take a pledge from the mixing participants $i \in \{1,\ldots,n\}$ for the required funds. The settlement involves creating $n * M_i$ multi-signature addresses among participants and then transferring mixing funds to the newly created addresses. Ideally, a single multi-signature address M should be sufficient for the settlement phase, but generating a single address for n participants offers a weak security guarantee in a way

[8] Participant of the protocol. Section VII describes a detailed case study of three smart devices acting as a mixing peer.

Please cite this article as: U. Sarfraz, M. Alam and S. Zeadally et al., Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions, Computer Networks, https://doi.org/10.1016/j.comnet.2018.11.019

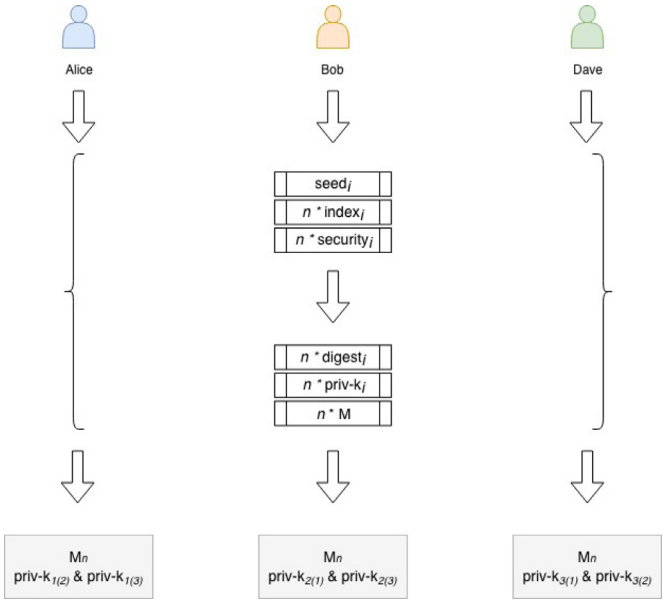

Fig. 2. Overview of the settlement phase with three participants. Generates three multi-signature addresses and assigns 1-N ownership of the address to each participant.

that there is no counter-measure for any participant who bails out from the settlement phase when all participants transfer mixing funds to the multi-signature address. This can lead to the possibility of locked funds because such funds cannot be further spent from M due to the missing signatures of the participant who bailed out.[9] Fig. 2 presents an overview of the settlement phase. Using the 1-N digital multi-signature scheme, we show how n participants generate $n * M_i$ multi-signature addresses and how ownership for spending z tokens will be distributed among the participants in a secure way:

(S1) Each participant $i \in \{1, \ldots, n\}$ determines key indexes $k_1, k_2 \ldots k_n \mid k_i, \ldots, k_n \in \{0, 1, 2 \ldots\}$ and security levels $s_1, s_2 \ldots s_n \mid s_i, \ldots, s_n \in \{1, 2, 3\}$ to generate digests $d_1, d_2 \ldots d_n$ and private keys $pr_1, pr_2 \ldots pr_n$ using a cryptographic sponge function.

(S2) Each participant shares the corresponding digests $d_1, d_2 \ldots d_n$ to generate $M_1, M_2 \ldots M_n$ multi-signature addresses.

$$\text{Addresses-Digests mapping} \begin{Bmatrix} M_1 \rightarrow D_{i(_1)}, \ldots, D_{n(_1)} \\ M_2 \rightarrow D_{i(_2)}, \ldots, D_{n(_2)} \\ \ldots \\ M_n \rightarrow D_{i(_n)}, \ldots, D_{n(_n)} \end{Bmatrix}$$

(S3) For each generated multi-signature address, every $(n - i)$ participants shares their corresponding private keys $pr_1, pr_2 \ldots pr_n$ to make a 1-N mapping for address ownership.

(S4) Every participant makes a transaction T_i to the corresponding multi-signature address M_i on which they have complete ownership.

Mixing participants validate the multi-signature address by publicly sharing digests. If the address validation fails, then it leads to the possibility of a malicious participant being present. In this case, the protocol aborts and all participants are notified.

5.2. Outputs shuffling

The aim of outputs shuffling is to obliviously randomize the set of output addresses declared by the participants so that no participant learns about the mapping of input addresses $A\_1, A_2, \ldots, A_n$ to output addresses $O_1, O_2, \ldots, O_n$. *Verifiable Shuffling* is a well-known problem in anonymous communications and one of the promising solutions to this problem is proposed in [42–44]. These techniques have inspired mixing techniques for Bitcoin [13,14]. This step in our protocol is based on the results described in [13] and the various steps of our shuffling technique are as follows:

(O1) Each participant creates an unused[10] IOTA address $O_1, O_2, \ldots, O_n$. These output addresses are generated so that they can be used in the mixing transaction.

(O2) Each participant creates a fresh transient encryption decryption key pair (Ek_i, Dk_i) and announces the relevant public encryption key.

(O3) Once every participant learns the encryption keys of each other, the first participant creates a layered encryption of his/her output address i.e., sequentially encrypts his/her output address with encryption keys of all participants. The first participant passes the cipher text to the next participant.

(O4) Each subsequent participant i up to the last one decrypts the outermost layer of encryption for all cipher texts with his/her corresponding decryption Dk_i. Each participant expects to receive a list of cipher texts with $i - 1$ size.

(O5) After decrypting one layer of encryption for all cipher texts, each participant i randomly shuffles the cipher texts he/she receives and then creates a nested encryption for his/her output address O_i.

(O6) The last participant decrypts all the cipher texts and then shuffles the final decrypted list with his/her own corresponding output address. Finally, the last participant shares the final list of output addresses with all the participants.

Fig. 3 includes an overview of a successful run of the *outputs shuffling* phase with three honest participants. Following the step (O1), the participants Alice, Bob and Dave generate a fresh output address O_1, O_2 and O_3 respectively. In the next step (O2), each participant (except for Alice) creates a fresh encryption decryption key pair. Bob and Dave announce their encryption keys Ek_B and Ek_D respectively. Following the step (O3), Alice encrypts her output address O_1 in a sequential manner i.e., Alice first encrypts for Dave obtaining $c_1 = \overline{E}(Ek_D, O_1)$ and then for Bob obtaining $c_2 = \overline{E}(Ek_B, c_1)$. Alice then adds c_2 to a list $L = (c_2)$ and passes L to Bob. Following the steps (O5) and (O6), Bob after receiving L from Alice, decrypts the outermost layer of the cipher text c_2 in list L, obtaining c_1. Bob also creates a new cipher text $c_3 = \overline{E}(Ek_D, O_2)$ for his output address O_2 and adds it to L. Bob then shuffles the list $L = (c_1, c_3)$ and passes L to Dave. Following the step (O6), Dave after receiving the list L from Bob, decrypts the outermost layer for both cipher texts c_1 and c_3 in L and obtains O_1 and O_2. Dave adds his address O_3 to the decrypted list, performs a shuffle over the final list and announces the final list after performing a shuffle.

Upon receiving the decrypted list, every participant verifies if his/her output address is in the list. If there is a duplication of output addresses or if there is any output address that is missing from the list, the protocol breaks and the participants enter the fall back phase. Additionally, if the decryption fails or multiple decryptions lead to the same output, the corresponding participant aborts the protocol, announces this anomaly to every participant and all the participants enter the fall back phase.

[9] Multi-signature addresses require signatures from all co-signers, hence in the absence of a missing participant the funds would be locked.

[10] An address with no transactions and zero balance.

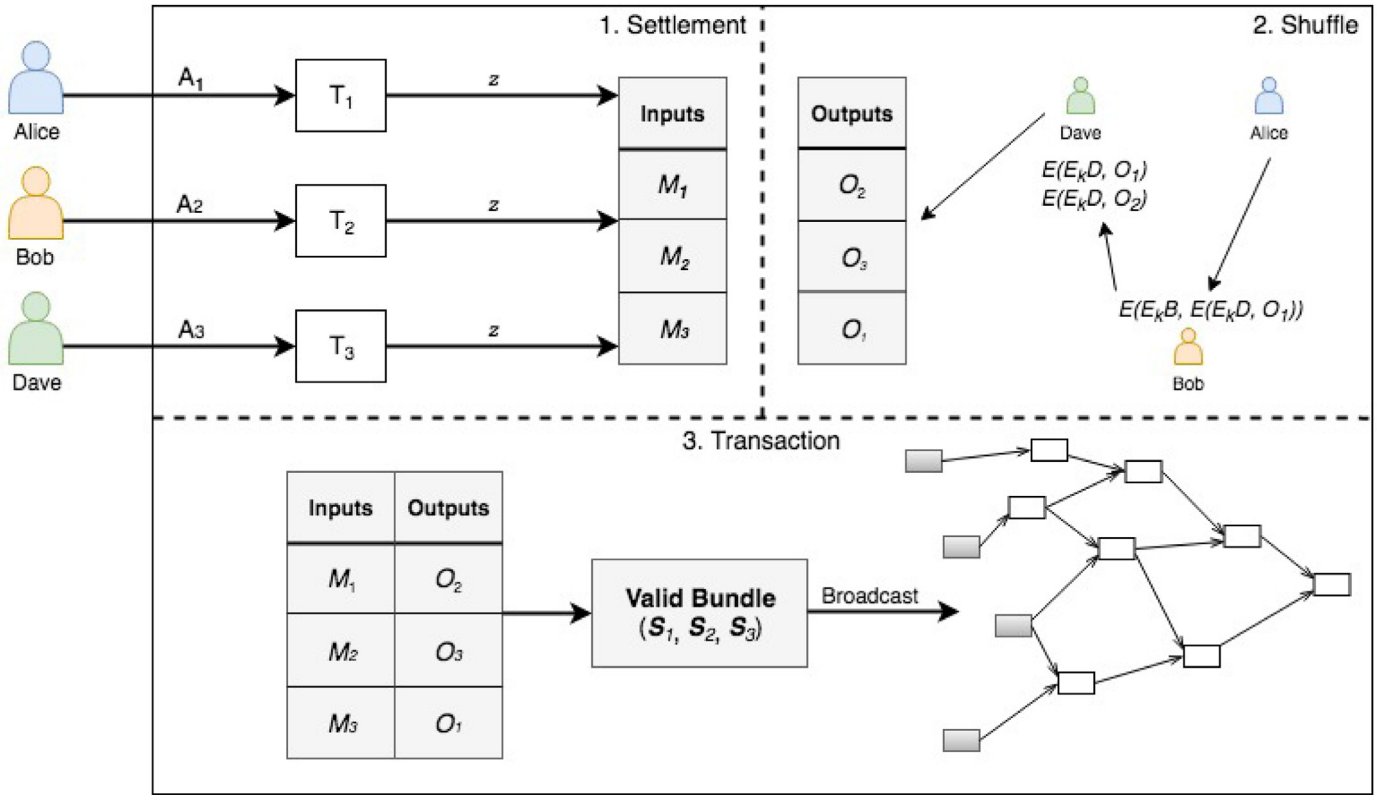

Fig. 3. Overview of the protocol with three honest participants.

5.3. Transaction

The aim of the transaction phase is to construct the final transaction bundle after the mix and broadcast the transaction to the network. Essentially, this means that the funds from each participant's input address $A_1, A_2, \ldots, A_n$ will be transferred to the corresponding output address $O_1, O_2, \ldots, O_n$. Since the input addresses are actually multi-signature addresses, this means a single signature of any participant would be insufficient for the transaction signing and hence the signature of all participants on each transaction is required.

(T1) Every participant deterministically creates a (not yet signed) mixing transaction that spends IOTA tokens from each of the input addresses in $L_{in} = (A_1, A_2, \ldots, A_n)$ to $L_{out} = (O_1, O_2, \ldots, O_n)$.
(T2) Participant i signs the transaction according to the specification of the IOTA protocol and broadcasts the signature.
(T3) Upon receiving a valid signature from each participant j, participant i adds all signatures to the bundle. Participant i checks if any of the other participants has spent his/her money reserved for mixing in the meantime. If this is the case, participant i aborts the protocol and enters the fall back phase.
(T4) Participant i gets two transactions to approve from the network, performs proof of work, and then generates the correct transaction hash and broadcast the bundle to the IOTA network.

5.4. Fall back

When an error or diversion from the normal operation of the protocol is detected, the fall back phase is entered. During the *settlement* phase, if any participant bails out and does not transfer the required funds to the newly created multi-signature address, then the operation of the protocol depends on the number of participants who transferred their funds to their respective multi-signature addresses that they own. If there is only one participant who transferred his/her funds, the normal operation of the protocol cannot be carried out and hence the participant will have to transfer the funds back to one of his/her new output addresses. If the IOTA network reports that the value of the coins at an input address is below v, or that the tokens at an input address A have already been spent, participant i broadcasts the transaction that sent the insufficient coins to the input address or the transaction that spent the coins. It is slightly more difficult to hold mixing peers accountable for malicious behavior during the *outputs shuffling* phase. However, as this has already been previously described in [13], in this work, we only present the basic idea. The output addresses do not need to be kept secret if an error occurs. If the mixing errors out, the honest mixing peers reinitialize the protocol by leaving out the malicious peers. The randomness used for the construction of layered encryption allow honest participants to trace down the malicious participants. Honest participants can sequentially reconstruct the intermediate shuffling and since all the messages (i.e., output addresses) are signed by participants, malicious adversaries can thus be held accountable for deviating from the protocol. It is worth noting that, as a consequence, the protocol needs to be rerun with new (unused) output addresses which is an acceptable good practice even for correct protocol executions.

6. Analysis of system properties

In this section we demonstrate that our proposed solution for decentralized mixing in IOTA ledger satisfies the design requirements we have presented in section IV. We explain how our solution increases the anonymity in section *A*. Section *B* provides an in-depth evaluation of how mixing correctness will be achieved even in the presence of malicious participants. Sections *C, D, E*

Please cite this article as: U. Sarfraz, M. Alam and S. Zeadally et al., Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions, Computer Networks, https://doi.org/10.1016/j.comnet.2018.11.019

briefly discuss *compatibility, cost efficiency* and *performance* of our proposed solution.

6.1. Anonymity

The anonymity property in our case implies 1) the unlinkability between input and output addresses of the mixing participants and 2) the anonymity level achieved by each mixing participant. We first explain how our proposed solution achieves unlinkability and then analyze the level of anonymity it guarantees to the mixing participants.

An IOTA mixing protocol achieves unlinkability (after a mixing operation) if an output address O_i that belongs to a participant i is indistinguishable from his/her input address I_i. To analyze the unlinkability property with our proposed solution, we omit the case where the normal operation of the protocol is disrupted and mixing participants enter the fall back phase. If participants enter the fall back phase, the output addresses of the mixing participants are discarded and the protocol is rerun with new output addresses. Unlinkability only becomes relevant if the mixing participants have entered the *outputs shuffling* phase because entering this phase require mixing the output addresses among the participants, hence successful or unsuccessful run of the *settlement* or *transaction* phase does not add anything to the unlinkability property. Our scheme for shuffling is inspired from [13,43] and thus we only provide a brief overview here. Due to the layered encryption of all participants, participant i receives a list L_{i-1} of $i-1$ cipher texts. These cipher texts do not reveal any link between the participant and output addresses because no participant knows about the corresponding private key D_{ki} and thus cannot conclude which output address is present in which layered cipher text.

Based on observations of the *outputs shuffling* run, a malicious participant can try to guess mapping between input and output addresses of other participants. The larger the set of addresses an attacker has to guess from, the higher the level of anonymity it provides to every participant because a larger set of addresses leads to a smaller probability of a correct guess. After a successful run of *outputs shuffling* phase, all mixing participants learn which output addresses are involved in the mixing, as they have to sign the transaction bundle before broadcasting it to the IOTA network. However, since shuffling guarantees *unlinkability* (as explained previously), the anonymity level against mixing participants is equal to the total number of mixing participants n.

6.2. Correctness

To prove the correctness of our proposed solution in the presence of malicious participants, we discuss each phase (i.e., settlement, outputs shuffling, and transaction). We provide a proof that a successful or an unsuccessful run of the protocol does not lead to loss of any participant funds. We also provide a proof that no malicious participant is able to steal portions of private key (signature forgery) from any other honest participant in the protocol.

In the *settlement* phase, after generating n number of multi-signature addresses where n is the number of mixing participants, the mixing participants transfer the required mixing funds to the multi-signature address M_i they own.[11] The correctness of our proposed solution in this phase depends on the correctness of the transaction $A_i \xrightarrow{z} M_i$. The correctness here is independent of our proposed solution and is ensured by the IOTA network. We note that the settlement phase ideally may only require a single multi-signature address M but since there are no countermeasures for malicious adversaries bailing out of the protocol, the participants need to generate n addresses. Each participant i gets a 1-N ownership over each multi-signature address M_i. A refusal to settling funds on corresponding multi-signature addresses or a refusal to provide address digests during multi-signature address generation does not affect the execution of the protocol in any way because the other participants can carry on mixing even in the absence of that particular participant. Hence *availability*[12] property on funds is preserved, making sure no funds of any participant are locked for spending even in the presence of malicious participants.

In the *outputs shuffling* phase, a malicious participant could substitute the encrypted output address with his/her own output address by encrypting it with the public keys of the remaining mixing participants. However, this malicious substitution is detected at the end of the *outputs shuffling* phase when the last mixing participant removes the last layer of encryption from the output addresses and announces the decrypted shuffled output addresses. The malicious substitution is detected by the affected participants as their output addresses are not in the decrypted shuffled list and all participants enter the fall back phase.

In the *transaction* phase, after steps (T1) and (T2) are performed a malicious participant could try to brute force the corresponding private key of the input address of any other participant and could refuse to sign the transaction. In this case, the protocol enters the *fall back* phase but at this point, the malicious adversary has now *50%* of the private key. In this case, it is unfeasible for an adversary to bruteforce the private key as it requires around 2^{256} tries to forge the signature but it becomes easier for an adversary to forge as he/she could get more fractions of the private key when the other honest participants transfer their funds back to their new output addresses. The average number of attackers tries to be able to forge the signature is approximately $(\frac{t+1}{t})^{s*27}$ where t is the number of outgoing transactions and s is the security level of the private key. The private key length for any address is s*6561 *trits* and non-multi-signature addresses have a limit of 3 on security levels. However, in the multi-signature scenario the security level that is used to generate the private key is the sum of all security levels $\sum_{s=i}^{n}$ provided by the participants. Even if the normal operations of our protocol are disrupted, we would ensure that $t \leq 2$ and hence even in the case where a malicious participant breaks the protocol by learning the signature of other participants, signature forgery is unfeasible.

At any point after a successful run of *settlement* phase, a malicious adversary with input address I can make a transaction $I \xrightarrow{z} O$, transferring all the funds to a new address. This anomaly is detected before broadcast, where the participant that is about to broadcast the final transaction bundle checks the latest balances on all the input addresses of the transaction bundle. If any input address is no longer funded, the broadcast of the final transaction is canceled and the participants enter the *fall back* phase.

6.3. Compatibility

A decentralized mixing protocol for the IOTA ledger ensures compatibility if it requires no changes or patches to the original IOTA protocol. Standard IOTA transactions are not multi-signature transactions and most of the users use a security level 2 for generating addresses.[13] While multi-signature addresses do bring an overhead of maintaining multiple keys, the compatibility boils down to the semantics of the generated multi-signature addresses and transactions broadcasted to the network. Our proposed solution thus require no changes to the IOTA protocol because 1)

[11] N-N and M-N ownership schemes are allowed in multisignatures. N-N require signatures from all participants while M-N requires sharing private keys in a way that signatures of M cosigners are sufficient.

[12] Availability of funds from an input address during the time a transaction remains unconfirmed.

[13] IOTA's official Trinity wallet uses security level 2 for all transactions.

Please cite this article as: U. Sarfraz, M. Alam and S. Zeadally et al., Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions, Computer Networks, https://doi.org/10.1016/j.comnet.2018.11.019

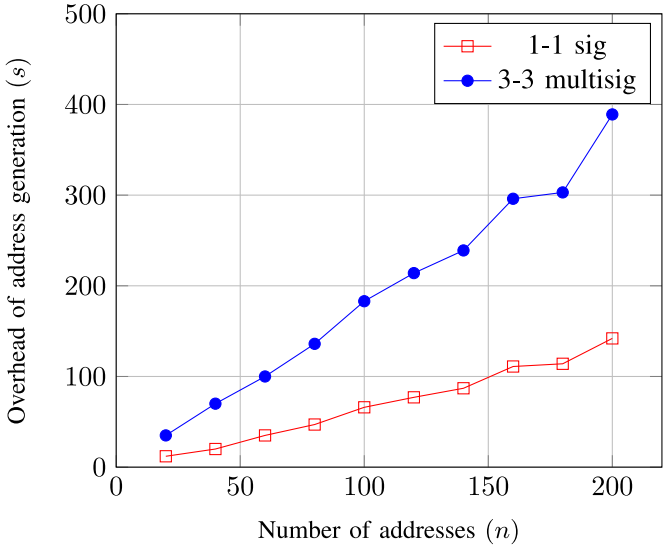

Fig. 4. Performance of Multi-signature vs Normal address generation.

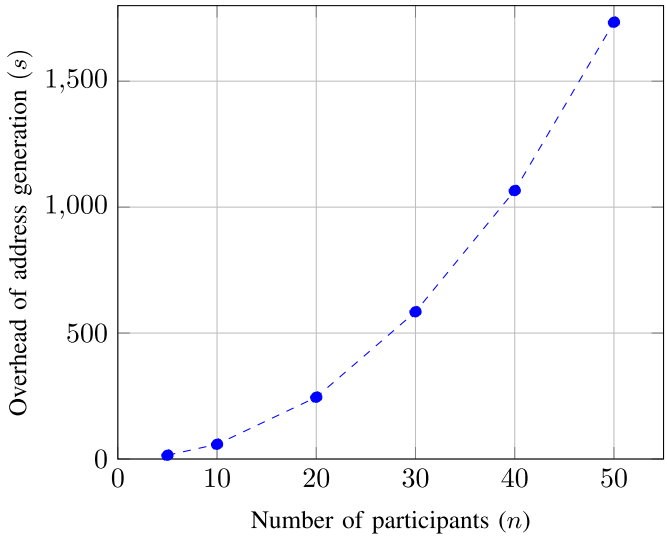

Fig. 5. Address generation for $n = 5, 10, \ldots, 50$ multi-signature addresses with $1 - n$ mapping (ownership).

multi-signature addresses are exactly like normal IOTA addresses and their size is 81 trytes and 2) the final transaction made after a successful run of our proposed solution does not differ semantically from normal IOTA transactions. This also adds a unique *deniability* property to our proposed solution because an external observer cannot distinguish a normal transaction from a mix transaction.

6.4. Fees

Since IOTA transactions are completely feeless, there are essentially only one type of fees that can be charged for mixing i.e., fees charged by centralized mixing services. Our proposed solution is cost efficient because any number of peers n can collaborate with other interested peers in making private transactions without involving any third party and hence avoid paying any additional fees.

6.5. Performance evaluation

In this section, we show that our proposed solution fulfills the *efficiency* requirement we have presented in section IV. We show that our proposed solution scales to a large number of mixing participants and does not add any prohibitive overhead to the mixing, the users or the IOTA network. We have implemented a proof-of-concept implementation of our proposed solution. In particular, we have implemented *settlement* and *transaction* phases of the protocol and have measured performance of each phase without disruption.

The implementation is based on *Node* version 8.11.1. We have tested our implementation on a local network setup with controlled network constraints. Functionality related to IOTA addresses, transactions and multi-signatures has been implemented using *iota.lib.js* [45], the official JavaScript library of IOTA ledger. Our evaluation has been carried out on a Mac system with 8GB 2133 MHz LPDDR3 RAM and 2.3 GHz Intel Core i5 processor.

Our proposed multi-signature scheme increases the size of the computed private key, hence we have benchmarked the generation time for different sets of addresses with size $n = 10, 20, \ldots, 200$ and security level $s = N * 2$ where N is the number of co-signers for each address. Fig. 4 presents a comparison of address generation time for $N = 1$ (single co-signer) and $N = 3$ (multiple co-signers). The generation time increases linearly with n for both mappings $N = 1$ and $N = 3$. For each set S of addresses with size

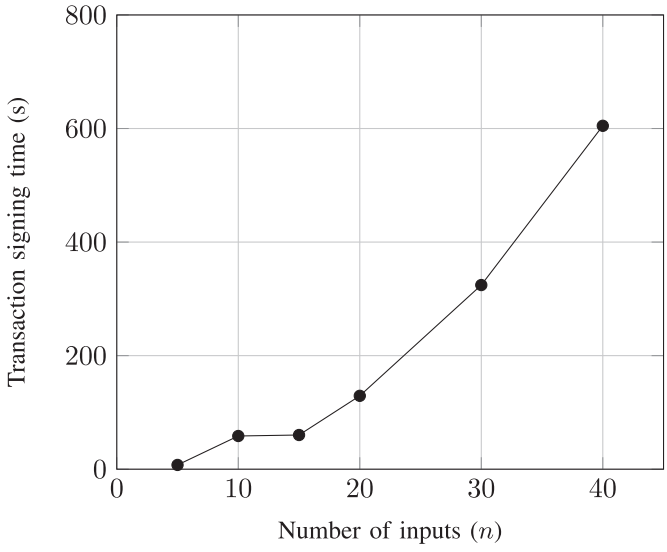

Fig. 6. Signing time for $n = 5, 10, \ldots, 50$ input addresses for $m = n$ participants.

n, the address generation time with $N = 3$ is always greater than the address generation time with $N = 1$ because of the increased size of computed private keys.

In the *settlement* phase, our proposed scheme requires generating n multi-signature addresses and forming a $1 - N$ multi-signature mapping for each generated address, hence we evaluated the performance of generating $n = 5, 10, \ldots, 50$ multi-signature addresses and assigning each participant i an ownership on each address. Fig. 5 shows the overall execution time for generating n addresses with $1 - N$ ownership mapping of each address M_i. Our performance evaluation uses a *security* parameter $s = 3$ in this scenario. The execution time increases approximately quadratically because for each address M_i, each participant i will attain ownership over $n - 1$ participants and most importantly because the length of the corresponding private keys will depend on the accumulated security of all participants n^*3.

In the *transaction* phase, our proposed scheme requires all participants to sign the input addresses $M_1, M_2, \ldots, M_n$. Fig. 6 displays the overall execution time for signing the final bundle for group transaction (after a successful mix). Our employed $1 - N$ scheme

Please cite this article as: U. Sarfraz, M. Alam and S. Zeadally et al., Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions, Computer Networks, https://doi.org/10.1016/j.comnet.2018.11.019

provides a great advantage at this step because each participant i can directly sign the input address M_i because he/she holds the private keys of other $n-1$ participants. To summarize, our experimental results depict the feasibility of our proposed solution even with a large number of participants and that it also fulfills the performance requirements.

7. CASE STUDY: IOTA smart air quality monitoring

Consider a smart air monitoring solution, designed to evaluate, visualize and function of the outdoor air quality enabled by tracking of environmental air pollutants. Consider a large number of these smart devices[14] are deployed for collecting climatic data using Tangle as a data integrity layer where all these information pieces are exchanged. With IOTA's data marketplace, different air quality improving companies can subscribe to the live stream of different devices in exchange for IOTA tokens. However, access to this data in exchange for iota tokens opens up privacy concerns because any subscriber can traverse the transaction history of the device. Moreover, a subscriber can actively monitor the device's transactions with other subscribers. These issues are undesirable for the firm owning these devices because it opens up the possibility for competitors to get private financial information.

Consider a smart air monitoring device D_1 that has IOTA tokens z at an address A_1 and wants to add anonymity to its transactional activity. With some bootstrapping mechanism, device D_1 discovers a smart vehicle charging station D_2 and a vending machine D_3 (both using IOTA for receiving payments) for the sheer purpose of performing a decentralized mix. Considering three devices n, Algorithm 1 requires $n = 3$ key indexes $k_1, k_2, k_3 \mid k \in \{0, 1, 2 \ldots\}$

Algorithm 1 Settlement.

```
1: procedure SETTLEMENT((seeds, indexes, security))  ▷ Generate n multi-signature addresses, get 1-N ownership and transfer funds
2:     size ← size of list
3:     n ← size(participants)
4:     i ← 0
5:     while i < n do  ▷ We will have n multi-signature address when i = (n − 1)
6:         for j = 0, j++, while j < n do
7:             s ← n_j(seed)
8:             idx ← n_j(indexes_i)
9:             sl ← n_j(security_i)
10:            iota.multisig.getDigest(s, idx, sl)
11:            if i ≠ j then
12:                iota.multisig.getKey(s, idx, sl)
13:        for each d in digests do
14:            address.absorb(d)
15:        address.finalize()
16:        if iota.multisig.validateAddress(M_i, digests then
17:            M_i ← address
18:        i++
19:    for each i in n do
20:        address ← Mi
21:        value ← v                          ▷ v > 0
22:        depth ← d                          ▷ d ∈ {0, ..., 15}
23:        mwm ← m                            ▷ m ≥ 14
24:        iota.api.sendTransfer()
```

and security levels $s_1, s_2, s_3 \mid s \in \{1, 2 \ldots\}$ from each device. Using Algorithm 1, each device generates n digests dig_1, dig_2, dig_3 and the corresponding private keys pk_1, pk_2, pk_3. Algorithm 1 generates a multi-signature address M for each digest dig generated by device.

$$\left\{\begin{array}{l}\text{Multi-signature address mapping}\\ M_1 \rightarrow dig1_{(D_1)}, dig1_{(D_2)}, dig1_{(D_3)}\\ M_2 \rightarrow dig2_{(D_1)}, dig2_{(D_2)}, dig2_{(D_3)}\\ M_3 \rightarrow dig3_{(D_1)}, dig3_{(D_2)}, dig3_{(D_3)}\end{array}\right\}$$

Each device needs to attain a 1-N ownership of a multi-signature address. Using Algorithm 1, each device would have the following mapping creating a 1-N scheme for each multi-signature address.

$$\left\{\begin{array}{l}\text{1-N multi-signature scheme}\\ D_1 \rightarrow M_1 \rightarrow pk1_{(D_2)}, pk1_{(D_3)}\\ D_2 \rightarrow M_2 \rightarrow pk2_{(D_1)}, pk2_{(D_3)}\\ D_3 \rightarrow M_3 \rightarrow pk3_{(D_1)}, pk3_{(D_2)}\end{array}\right\}$$

Algorithm 1 (L:16) validates $M_i, \ldots, M_n \mid i \in \{1, \ldots, n\}$. Each device D_i makes a transaction $A_i \xrightarrow{z} M_i$. Taking advantage of the feeless architecture of the IOTA protocol, the exact number of tokens will be transferred to the corresponding address M_i. Algorithm 2 requires each device D_i to create a fresh output address O_i and also a key pair of a public key encryption scheme, consisting of a public encryption key ek_i and a private decryption key dk_i. Using Algorithm 2, D_1 encrypts O_1 with ek_3 obtaining l_1 and again encrypts l_1 with ek_2 obtaining l_2. D_1 now passes l_2 to D_2 which first decrypts l_2 with dk_2 obtaining l_1. D_2 also encrypts O_2 with ek_3. Now D_2 has a list of two cipher texts (l_1, l_2). D_2 shuffles the list and passes it to D_3. When D_3 obtains the list (l_1, l_2), it decrypts both l_1 and l_2 to obtain O_1 and O_2 respectively. It also adds O_3 to the list and after performing a random shuffle over the final list announces the list with D_1 and D_2. The mapping in the final list after a random shuffle of all devices could look like the following:

$$\text{Output mapping after random shuffle}\left\{\begin{array}{l}D_1 \rightarrow M_1 \rightarrow O_2'\\ D_2 \rightarrow M_2 \rightarrow O_3'\\ D_3 \rightarrow M_3 \rightarrow O_1'\end{array}\right\}$$

Algorithm 2 Outputs Shuffling.

```
1: procedure SHUFFLE(ek_2, ek_3)  ▷ Shuffling for n = 3 participants
2: P1:
3:     l_1 ← Enc(O_1, Ek_3)
4:     l_2 ← Enc(l_1, Ek_2)
5:     goto P2.
6: P2:
7:     C ← Enc(Ek_2, Enc(Ek_3, O_1))
8:     D ← Enc(Ek_3, O_1)  ▷ Removes one layer of encryption Dk_2(Enc(Ek_2, Enc(Ek_3, O_1)))
9:     l_1 ← Enc(O_2, Ek_3)
10:    l_2 ← D
11:    a(l_1, l_2)  ▷ where a is a random permutation over list of ciphers l_1 and l_2
12:    goto P3.
13: P3:
14:    O_1 ← Dk_3(Enc(Ek_3, O_1))
15:    O_2 ← Dk_3(Enc(Ek_3, O_2))
16:    verify(a(O_2, O_1, O_3)).
```

After verification of each output address O_i' in the final decrypted list, Algorithm 3 makes the final transaction $M_i \xrightarrow{z} O_i'$. Algorithm 3 constructs the final bundle by requiring signatures from each device. D_1 possessing $(pk1_{(D_2)}, pk1_{(D_3)})$ adds

[14] Bosch deployed micro-climate monitoring system Climo in Las Vegas. However, these devices do not yet use a distributed integrity layer for data storage.

Please cite this article as: U. Sarfraz, M. Alam and S. Zeadally et al., Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions, Computer Networks, https://doi.org/10.1016/j.comnet.2018.11.019

Algorithm 3 Final Transaction Construction.

```
1: procedure TRANSACTION(inputs, outputs) ▷ Construction of final transaction to be broadcasted to the Tangle
2:     n ← size(participants)
3:     O ← O_1, ..., O_n
4:     M ← M_1, ..., M_n    ▷ List of all multi-signature addresses as inputs
5:     for each address in O do
6:         value ← v                                   ▷ v > 0
7:         transfers.add(address, value)
8:     for each address in M do
9:         b ← v                                       ▷ v > 0
10:        securitySum ← s
11:        inputs.add(address, balance, securitySum)
12:    bundle ← iota.crypto.Bundle()
13:    for each i in inputs do
14:        bundle.addEntry()
15:    bundle.finalize()
16:    for each tx in bundle do
17:        a ← tx.address
18:        b ← bundle
19:        k ← key
20:        if a = M_i then
21:            for j = 0, j++, while j < n do
22:                if ≠ j then
23:                    iota.multisig.addSignature(b, a, k_j)   ▷ j holds all private keys so can add all signatures
```

signature on its input address M_1. Similarly D_2, D_3 possessing $(pk_{2_{(D_1)}}, pk_{2_{(D_3)}})$, $(pk_{3_{(D_1)}}, pk_{3_{(D_2)}})$ respectively, add their signatures on M_1, M_2. The final transaction bundle is then broadcast to the IOTA network. After a successful mixing operation all devices participating in the mix are able to add anonymity by achieving unlinkability in their transaction chain.

8. Conclusion

The IOTA public distributed ledger is designed to enable instant micro-transactions for Internet of Things (IoT) devices. The Directed Acyclic Graph (DAG) based approach to design a scalable distributed ledger with zero fees enables numerous use-cases for smart devices. However, the linkable pseudonymity provided by the IOTA protocol raises privacy concerns. For example, transaction activity may be continuously monitored by adversaries to plan thefts or private financial information may be leaked to competitors (we described a detailed case study earlier). Many privacy enhancing solutions have been proposed for Bitcoin and its derived currencies to protect financial privacy of the end users but they fail compatibility with the IOTA ledger due to IOTA's use of quantum resilient hash-based signature scheme. While a dedicated mixing service for IOTA to improve anonymity [11] exists, but it still depends on a trusted third party and does not even provide anonymity against weaker passive adversaries (Mixing service has knowledge about the permutation applied to output addresses). In this paper, we have thus presented a novel mixing technique for IOTA tokens, which is secure, robust and completely compatible with the IOTA protocol. Sticking to the IOTA ledger's ideology, our proposed solution is fully decentralized and it neither requires any third party, nor introduces any additional mixing fees. Our scheme employs a combination of digital multi-signature scheme [31,32] and decryption mix-nets [13,42]. A comprehensive analysis of our implementation shows that our protocol only introduces computation overhead if the number of participants increases and most importantly, our quantitative security analysis provides a proof that no malicious adversary can practically forge signatures by stealing portions of private key of any participant. Hence, our presented solution for decentralized mixing makes it difficult to traverse payment histories, adds sufficient level of uncertainty, improves on the benefits of a centralized mixer, and thereby enhances the anonymity of transactions on IOTA ledger.

Acknowledgement

We thank Jian Liu for his advice and support during the revision of this paper. We also thank the anonymous reviewers for their valuable comments which helped us to improve the content, organization and presentation of this paper.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.comnet.2018.11.019.

References

[1] S. Nakamoto, Bitcoin: a peer-to-peer electronic cash system, 2008, (https://bitcoin.org/bitcoin.pdf).
[2] S. Popov, IOTA: the tangle (2016). http://iotatoken.com/IOTA_Whitepaper.pdf.
[3] Crypto currency market capitalizations, 2018, https://coinmarketcap.com/, accessed 05/07/2018.
[4] The Statistics Portal, 2018, https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide, accessed 05/07/2018.
[5] F. Reid, M. Harrigan, An analysis of anonymity in the bitcoin system, Privacy, Secur., Risk Trust (PASSAT) (2011).
[6] D. Ron, A. Shamir, Quantitative Analysis of the Full Bitcoin Transaction Graph, FC Springer, 2013.
[7] M. Fleder, M.S. Kester, S. Pillai, Bitcoin Transaction Graph Analysis, CoRR abs/1502.01657 (2014).
[8] S. Delgado-Segura, C. Pérez-Solà, G. Navarro-Arribas, J. Herrera-Joancomartí, Analysis of the bitcoin UTXO set, IACR Cryptol. ePrint Arch. 2017 (2017) 1095.
[9] Bundles, IOTA guide, (https://domschiener.gitbooks.io/iota-guide/content/chapter1/bundles.html). (Accessed on 08/05/2018).
[10] L. Tennant, Improving the anonymity of the IOTA cryptocurrency, 2017. Available at https://bit.ly/2vr0Tsb.
[11] L. Tennant, IOTA mixer v1 released IOTA @ UCL Medium, 2017b, (https://medium.com/iota-ucl/iota-mixer-91f3d39735c1b). (Accessed on 08/05/2018).
[12] CoinJoin: bitcoin privacy for the real world, 2013, (https://bitcointalk.org/index.php?topic=279249). (Accessed on 08/05/2018).
[13] T. Ruffing, P. Moreno-Sanchez, A. Kate, Coinshuffle: practical decentralized coin mixing for bitcoin, ESORICS (2014).
[14] T. Ruffing, P. Moreno-Sanchez, A. Kate, P2P mixing and unlinkable bitcoin transactions, NDSS (2017).
[15] M.H.F.G. J. H Ziegeldorf, R. Matzutt, K. Wehrle, Coinparty: secure multi-party mixing of bitcoins, CODASPY (2015).
[16] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J.A. Kroll, E.W. Felten, Mixcoin: anonymity for bitcoin with accountable mixes, in: Proc. of the 17th International Conference on Financial Cryptography and Data Security, 2014.
[17] G. Maxwell, CoinSwap: transaction graph disjoint trustless trading, 2013, (https://bitcointalk.org/index.php?topic=321228).
[18] J.B. D. J Bernstein, E. Dahmen, Post-Quantum cryptography, Springer Berlin Heidelberg, pp. 35–36.
[19] S. Rohde, T. Eisenbarth, E. Dahmen, J. Buchmann, C. Paar, Fast hash-based signatures on constrained devices, vol. 8, Springer, pp. 104–117.
[20] V.Sonny, V. Kapoor, R. Singh, Elliptic curve cryptography, ACM Ubiquity (2008).
[21] A.Menezes, D. Johnson, S. Vanstone, The elliptic curve digital signature algorithm (ECDSA), Int. J. Inf. Secur. (2001).
[22] Bitcoin mixer. bitcoin blender., (https://bitcoinmix.org/). (Accessed on 08/06/2018).
[23] Cryptomixer, (https://cryptomixer.io/). (Accessed on 08/06/2018).
[24] Chipmixer - bitcoin tumbler, (https://chipmixer.com/). (Accessed on 08/06/2018).
[25] M. Hearn, Merge avoidance, (https://medium.com/@octskyward/merge-avoidance-7f95a386692f). (Accessed on 08/06/2018).
[26] [Bitcoin-development] Stealth addresses, 2014, (https://lists.linuxfoundation.org/pipermail/bitcoin-dev/2014-January/004020.html). (Accessed on 08/06/2018).
[27] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, Zerocash: decentralized anonymous payments from bitcoin. in security & privacy, in: Proceedings of IEEE Symposium on Security and Privacy, 2014.
[28] M.G. I. Miers, C. Garman, A.D. Rubin, Zerocoin: anonymous distributed E-cash from bitcoin, Security & Privacy. IEEE, 2013.
[29] Hashlock - bitcoin wiki, (https://en.bitcoin.it/wiki/Hashlock). (Accessed on 08/06/2018).
[30] M.P. Guido Bertoni, J. Daemen, G. Assche, Cryptographic sponge functions, 2011, (https://keccak.team/files/CSF-0.1.pdf).

[31] IOTA multi-signature scheme, (https://github.com/iotaledger/wiki/blob/master/multisigs.md).(Accessed on 08/06/2018).
[32] A.B. Mushi, IOTA: multisig explained, (https://medium.com/@abmushi/iota-multisig-explained-bca334d250a2). (Accessed on 08/06/2018).
[33] D. Schiener, The anatomy of a transaction, IOTA Guide, (https://domschiener.gitbooks.io/iota-guide/content/chapter1/transactions-and-bundles.html).
[34] Tip selection - IOTA docs, (https://docs.iota.org/introduction/tangle/tip-selection).(Accessed on 08/06/2018).
[35] PoW on the tangle - IOTA docs, (https://docs.iota.org/introduction/tangle/proof-of-work). (Accessed on 08/06/2018).
[36] A. Pfitzmann, M. Hansen, Anonymity, unlinkability, unobservability, pseudonymity, and identity management. A consolidated proposal for terminology (2005).
[37] T.M. Cover, J.A. Thomas, Elements of Information Theory, John Wiley & Sons, Inc, pp. 12–21.
[38] C. Diaz, S. Seys, J. Claessens, B. Preneel, Towards measuring anonymity, in: Proc. 3rd Int. Workshop on Privacy Enhancing Technologies (PET 2003) (LNCS 2482), 2003.
[39] A. Serjantov, G. Danezis, Towards an information theoretic metric for anonymity, in: Proc. Int. Workshop Privacy Enhancing Technology, 2002, pp. 41–53.
[40] D. Chaum, Untraceable electronic mail, return addresses, and digital pseudonyms, CACM (1982).
[41] R.S.-N. Lan Nguyen, K. Kurosawa, A provably secure and efficient verifiable shuffle based on a variant of the paillier cryptosystem, J. Univers. Comput. Sci. (2005).
[42] H. Corrigan-Gibbs, B. Ford, Dissent: accountable anonymous group messaging, CCS (2010).
[43] J. Brickell, V. Shmatikov, Efficient anonymity preserving data collection, SIGKDD (2006).
[44] D. Chaum, The dining cryptographers problem: unconditional sender and recipient untraceability, J. Cryptol. (1988) 65–75.
[45] iotaledger/iota.lib.js: Iota javascript library, (https://github.com/iotaledger/iota.lib.js). (Accessed on 08/06/2018).

Dr. Masoom is heading the cyber security lab comsats islamabad.

Please cite this article as: U. Sarfraz, M. Alam and S. Zeadally et al., Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions, Computer Networks, https://doi.org/10.1016/j.comnet.2018.11.019