Vulnerabilities on Hyperledger Fabric

Article *in* Pervasive and Mobile Computing · July 2019

DOI: 10.1016/j.pmcj.2019.101050

CITATION
1

READS
471

5 authors, including:

Nitish Andola
Indian Institute of Information Technology Allahabad
7 PUBLICATIONS **8** CITATIONS
SEE PROFILE

Raghav Gahlot
Indian Institute of Information Technology Allahabad
2 PUBLICATIONS **1** CITATION
SEE PROFILE

Manas Gogoi
Indian Institute of Information Technology Allahabad
1 PUBLICATION **1** CITATION
SEE PROFILE

Sharannya Venkatesan
Saveetha University
28 PUBLICATIONS **185** CITATIONS
SEE PROFILE

Pervasive and Mobile Computing 59 (2019) 101050

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

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

Vulnerabilities on Hyperledger Fabric

Nitish Andola *, Raghav, Manas Gogoi, S. Venkatesan, Shekhar Verma

Network Security and Cryptography Lab, Department of Information Technology, Indian Institute of Information Technology Allahabad, Allahabad, U.P., 211012, India

ARTICLE INFO

Article history:
Received 6 December 2018
Received in revised form 27 May 2019
Accepted 5 July 2019
Available online 11 July 2019

Keywords:
Zero knowledge
Anonymity
Wormhole attack
Bilinear pairing
Hyperledger Fabric

ABSTRACT

In this paper, we have precisely analysed Hyperledger Fabric and pointed out two security limitations with possible solutions. First, the identity of an endorser is known to all members within a channel, which opens a gateway for DoS attack on endorsers in order to either block transaction pertaining to a client, or to degrade network efficiency. Second, the technology is prone to wormhole attack i.e. within a channel, compromising a member leads to leakage of ledger information of all members, to everyone outside the channel. We have proposed a solution to remove the above mentioned weaknesses. We have proposed two different mechanisms to eliminate the first weakness. The first approach uses a random verifiable function to randomize endorsers, while the second approach uses pseudonyms to anonymize endorsers. To address the second weakness, we have anonymized sender and receiver identity inside a channel. We have used a group signature approach using bilinear pairing to hide the sender identity and a zero knowledge approach using bilinear pairing to anonymize the receiver. The approach is immune to a malleability attack. Proper security proofs have been provided for Signature Unforgeability and Unlinkability in Ciphertext (UN-C). We have provided experimental results to measure the impact of DoS attack on hyperledger network using Hyperledger Caliper. After applying DoS attack on two peers, the throughput is reduced from 125 *tps* to 100 *tps* at send rate 123 *tps*. The latency in increased from 1.396 *s* to 2.44 *s* at send rate 123 *tps*.

1. Introduction

Blockchain is one of the most disruptive technologies in the modern era, due to its use in achieving decentralization while maintaining integrity in the system. Organizations are gradually turning in to this technology, to facilitate trade in a private, secure and trusted manner across platforms, in an otherwise untrusted scenario. This technology can be partitioned in three types i.e. public, private and permissioned blockchain. In public blockchain [1–5], no one has control over the network. This ensures that the data cannot be changed once validated on the ledger. However, public blockchain is limited in a sense that in a certain situation, an organization may want to employ blockchain technology with restricted members. In private blockchain, [6] a peer can perform any action in private network i.e. deploy contracts, make transactions, validate blocks, and interact with contracts. Ethereum [2] has provided this feature by the name of private ethereum. But, in this network all the peers are treated equally, so no access mechanism is provided. To overcome this issue, Permissioned blockchain [7] comes into the picture in which proper access control is maintained. Hyperledger fabric technology [8] has leveraged the benefits of both private and permissioned blockchain. In this framework, peers are

* Corresponding author.
E-mail address: andola.nitish18@gmail.com (N. Andola).

https://doi.org/10.1016/j.pmcj.2019.101050
1574-1192/

of four types i.e. transaction proposer, endorsers, orderers and validators. The transaction proposer proposes a transaction; endorsers endorse a transaction; orderer job is to bundle a transaction into a block and broadcast it; validators validate and update blocks on the ledger. For maintaining the access control mechanism, separate channels are maintained in hyperledger technology. However, hyperledger has some drawbacks namely (a). Identity of the endorser is known to all participants in a channel; this information makes DoS attack possible on selective endorsers to block certain transactions or degrade the efficiency of the network. (b). In a channel, the identity of each peer is known to every other peer. This renders Hyperledger Fabric prone to wormhole attack i.e. compromising any peer in the channel leads to leakage of transaction information of the entire channel. Such an attack can be avoided if the privacy of the peers is preserved. Privacy preservation against the internal attacker is proposed in [9]. However, preventive model against inside adversary specific to hyperledger fabric is needed.

We have proposed two different approaches to eliminate the first drawback. In the first approach, we have used a sortition algorithm inspired from algorand [10] in which the endorsers are randomly selected from a verifiable random function (VRF). In the second approach, the anonymity of endorsers is maintained using pseudo identity i.e. endorser endorses a transaction using pseudonyms. Further, to remove the second drawback, we have proposed two mechanisms. The first approach aims to anonymize sender's identity using a group signature algorithm. The second approach provides receiver anonymity using bilinear pairing. We have used searchable encryption technique for anonymizing receiver. In other related work [11], a secure channel free certificateless searchable public key encryption with multiple keywords scheme for IIoT is deployed.

There have been various schemes in the literature that claims to be secure but later proved to be flawed due to the weak security and adversary model. In [12], authors have attempted to break the undesirable cycle of "break–fix–break–fix" and proposed a systematical evaluation framework for schemes to be assessed objectively. In [13], 7 attacking scenarios in which a two factor authentication may fail along with the comparative evaluation of 26 representative schemes have been demonstrated. In [14], a security model is defined which can capture the practical capabilities of an adversary along with a set of twelve properties is framed for comparative evaluation. In [15] two factor authentication has been upgraded to three factor authentication while retaining the practice friendly properties for two-factor authentication. [16] proposes a scheme which facilitates efficient login, mutual authentication, session key agreement, verification and password update phases along with password recovery. A lightweight scheme in [17], provides biometric verification along with password verification for authorization and access to various application servers. A new framework for the handshake scheme based on hierarchical identity-based cryptography is given in [18]. However, all the above schemes lack anonymous authentication of users. We propose an authentication mechanism using group signature scheme [19] which maintains complete anonymity of sender.

Organization: This subsection describes the workflow of paper. Section 2 enlightens the contribution of the paper. Section 3 covers the prerequisites of bilinear pairing followed by wormhole attack, VRF and few other definitions. In Section 4, the system infrastructure and adversarial model is discussed. Section 5 describes the vulnerability of hyperledger fabric against DoS attack and wormhole attack. In Section 6 we have proposed various approaches to eliminate the weaknesses in hyperledger. Section 7 describes the security proofs for Signature Unforgeability and Unlinkability in Ciphertext (UN-C). Section 8 analyses the effect of DoS attack on the performance of hyperledger network followed by conclusion in Section 9.

2. Our contribution

1. We have precisely identified and analysed the weaknesses on hyperledger fabric, and a new framework is proposed for permissioned blockchain.
2. We have provided anonymity mechanisms in terms of identity the of sender and the receiver inside a channel in hyperledger fabric.
3. We have provided anonymity mechanisms for endorsers on hyperledger fabric.
4. We have provided experimentation results to analyse the effect of DoS attack on the performance of hyperledger fabric network.

3. Preliminaries

Let G_1, G_2 be two additive cyclic groups of prime order q and G_T another cyclic group of order q written multiplicatively. A pairing is a map: $e : G_1 \times G_2 \rightarrow G_T$, which satisfies the following properties:

1. Bilinearity $\forall a, b \in F_q^*,\ \forall P \in G_1, Q \in G_2 :\ e(aP, bQ) = e(P, Q)^{ab}$
2. Non-degeneracy: $e \neq 1$
3. Polynomial Time Computable: The algorithm to compute pairing must take a polynomial amount of time.

Wormhole Attack: Within a private network, a malicious peer creates a virtual private network with the outside network and leaks the information of its own private network. This attack can be launched without any knowledge of honest members of the private network.

Verifiable Random Function (VRF): In a distributive environment, the input to VRF is a seed along with the public key of the user. The output is a random value with the following property. Later, it can be verified that the random value corresponds to a specific user by using its public key i.e. no other user without the corresponding secret key can claim the ownership of random value.

Malleability Attack: When a sender broadcasts his transaction to other peers, the adversary can modify the content of a transaction i.e. the signature or even modify the receiver identity and then recalculate the transaction hash and further broadcast the transaction. Now, the sender waits for the endorsement of his transaction, which never happens as the transaction hash was modified by the adversary. In this situation, the sender being confused, resend the transaction to the receiver.

Fixed Argument Pairing Inversion 1 (FAPI-1): FAPI-1 problem [20] is stated as "Given $P_1 \in G_1$ and $P_3 \in G_T$, compute $P_2 \in G_2$ such that $e(P_1, P_2) = P_3$".

Computational Diffie Hellman (CDH): Let G be a finite cyclic group generated by a group element P and $\forall a, b \in F_q^*$. CDH problem is stated as "Given set of parameters: P, aP, bP, compute abP".

4. System architecture and adversary model

4.1. System architecture

The hyperledger network mainly consists of 5 entities:

1. Membership Service Provider (*MSP*): The *MSP* is a peer which issues and validates certificates of the user after performing user authentication.
2. Client: A client is a peer which proposes a transaction to be updated in ledger.
3. Endorser: When a client sends the transaction proposal request to endorser, authentication of the client and simulation of the proposed transaction is performed by the endorser.
4. Orderer: After receiving a bunch of unconfirmed transaction from clients, an orderer collects a set of transactions in a block, and broadcast it to the network.
5. Validator: The role of the validator is to validate the unconfirmed transactions inside a block and update it in the ledger in case of successful validation.

The block updation consists of 4 phases (a). Transaction proposal phase consists of the transaction proposal by a client to the endorser. (b). The endorsing phase consists of authentication of the client followed by validation of transaction by the endorser. (c). The creation of block by collecting the unconfirmed transactions sent by clients; and broadcasting the block by the orderer comes under the ordering phase. (d). The committing phase consists of the validation of unconfirmed transaction followed by commitment of transactions in the ledger.

4.2. Adversary model

In hyperledger network, the ledger is accessible to all the members of the network excluding the interference of peers outside the network. Further, the ledger of a channel inside the hyperledger limits the accessibility to only members that are part of the channel. Adversary $\mathcal{A}$ is a member of the channel on hyperledger, i.e also a member of hyperledger network. As in conventional data sharing schemes, $\mathcal{A}$ can eavesdrop all the network traffic inside a channel of hyperledger fabric, $\mathcal{A}$ has access to every transaction present in the ledger. It is assumed that the identity of a member inside a channel can be determined by $\mathcal{A}$. It is practical to assume that $\mathcal{A}$ can modify the unconfirmed transactions sent by the client to endorser during the transaction proposal phase. The identity of an endorser is assumed to be known to the adversary. This makes the endorser prone to DoS attack.

5. Problem statement

In Hyperledger Fabric, the client can choose endorser E of its choice during the transaction proposal phase, due to which the identity of E is disclosed to everyone in the network including an insider adversary. Also, during the entire process from transaction proposal to updation of transaction in hyperledger, the identity of the client is known to everyone. Therefore, attacks on the system as mentioned below are possible.

1. **DoS on endorsers**: As the identity of endorsers in known to everyone in the network, an insider malicious peer can launch a DoS attack on endorsers to achieve two objectives. The first motive of adversary $\mathcal{A}$ is to block client transactions for updating into ledger. $\mathcal{A}$ will constantly eavesdrop the network traffic; whenever the target client pk_{tc} proposes transaction T to chosen endorsers during the transaction proposal phase, endorser sends a response

back to pk_{tc} after endorsing T. $\mathcal{A}$ can modify or dump a certain number of responses by the endorsers so as to defeat the policy requirement of the chain code for T i.e. if chain code policy requires endorsement of transaction by at least 10 endorsers, and the client chooses 15 endorsers for endorsement, assuming 15 successful responses being sent back to client, $\mathcal{A}$ will dump or modify 6 responses intended to pk_{tc}. Now only 9 responses are sent back to the client which means failing of transaction proposal phase. In the above case, pk_{tc} will have to again repeat the transaction proposal. Similar attacks by $\mathcal{A}$ in every attempt of pk_{tc} will prevent pk_{tc} to propose T.
The second motive of $\mathcal{A}$ is to degrade the overall network efficiency. Targeting specific endorsers in the network will lead to failure of endorsement of transactions, which will directly affect the rate of block generation in the ledger. In Section 7, experimental results are provided to support our claim.

2. **Impact of Wormhole attack**: In Hyperledger Fabric, the channel is a medium via which we can separate a group of identities from rest. Everyone inside the channel has access to the ledger. If a single member in channel acts maliciously and colludes with the outer adversary, wormhole attack is possible. This will lead to leakage of confidential information of all members of the channel. We conclude the problem by stating the fact that the access control mechanism of hyperledger entirely depends on trusting every member inside the channel.

6. Proposed work

This section is partitioned into two subsections. The first subsection focuses on providing anonymity of endorsers using two different techniques. In the first technique, the anonymity of an endorser is guaranteed by randomizing endorser i.e. every user in the channel is equally probable to act as endorser for every new transaction. The second technique covers the anonymization of endorsers using pseudo identities. In the second subsection, our aim is to anonymize the transactions inside a channel i.e. the members of a channel can avail the benefits pertaining to that channel, but no member can view transaction communicated between two members within that channel. This anonymization is done in two steps (a). The sender is anonymized using group signature approach (b). The receiver is anonymized using the bilinear pairing based approach. In this approach, no world state is maintained to check the validity of a transaction, alternatively, the validation check can be done by backtracking similar to bitcoin.

6.1. Anonymizing endorsers

There are two ways to anonymize endorsers:

1. **Using VRF Sortition**: To avoid the pre-knowledge of endorsers in the chaincode, alternatively, we define a VRF similar to [10] in the chaincode, via which the endorsers will be selected randomly for endorsing a specific transaction. The steps involved in the process are as follows:

Algorithm 1 procedure Sortition(sk, T_{id}, τ, N)

(hash, π) $\leftarrow VRF_{sk}(T_{id})$
p $\leftarrow \frac{\tau}{N}$
j $\leftarrow 0$
if $\frac{hash}{2^{hash\,1en}} \in \left[B(0;1,p), \sum_{k=0}^{1} B(k;1,p) \right)$ do
$j = 1$
return$(hash, \pi, j)$

Algorithm 2 procedure VerifySort(pk, hash, π, τ, T_{id}, N)

if $VerifyVRF_{pk}(hash, \pi, T_{id})$ **then return** 0;
$p \leftarrow \frac{\tau}{N}$
$j \leftarrow 0$
if $\frac{hash}{2^{hash\,1en}} \in \left[B(0;1,p), \sum_{k=0}^{1} B(k;1,p) \right)$ do
$j = 1$
return j

(a) Client application broadcasts the transaction proposal request to all peers in the channel.

(b) Each peer individually runs the sortition algorithm. This algorithm outputs if a peer is selected as endorser or not.

(c) As shown in Algorithm 1, a peer computes $\langle hash, \pi \rangle \leftarrow VRFsk(T_{id})$ using VRF function, where sk is the secret key of peer, T_{id} represents transaction to be endorsed. $hash$ is pseudo-random value and π as its proof. τ represents the expected range of endorsers among N peers in the channel. $hash$ determines if a peer is selected or not and is calculated using a binary variable j. Selection of peers follows the binomial distribution, $B(k; w, p) = \binom{N}{k} \cdot p^k q^{N-k}$,

(d) If a peer is selected as an endorser, it endorses the transaction. Along with the endorsed transaction, the peer also sends $hash$ and π, which acts as a part of the proof that peer can act as an endorser.

(e) Later, the client and other peers can verify if a peer is a valid endorser or not using Algorithm 2. The pseudocode for verifying a sortition proof, as shown in Algorithm 2, follows the same structure to check if that peer was selected. The function returns if the peer is selected or not. This algorithm is run by the client, and the above selected peer(endorser) is publicly verified using the proof (from the VRF output).

2. **Using Pseudonyms**: Initially, the client broadcasts its transaction proposal request, as the identity of endorsers is not known, to provide anonymity of endorsers, each endorser E generates a set of pseudonyms certified by the MSP, and signs each transaction with a newly certified pseudonym. However, the protocol to certify a pseudonym by the MSP requires 4 steps which are given as follows:

(a) Initially, E generates a set of public private key pair (pk_E, sk_E). Here, pk_E acts as a pseudonym which is sent to MSP. It is assumed that the interaction between the endorser and MSP is authentic and is done via a secure medium.
Remark: Different pseudonym is used for endorsing a new transaction.

(b) MSP generates a certificate $Cer_{msp}(pk_E)$ corresponding to pk_E i.e. signs pk_E by its own secret key.

$$Cer_{msp}(pk_E) = \sigma_{sk_{MSP}}(pk_E)$$

(c) E uses its sk_E to endorse a transaction T i.e. signs T using its sk_E to generate Sig_E

$$Sig_E = \sigma_{sk_E}(T)$$

(d) To enable validation, psuedonym pk_E and $Cer_{msp}(pk_E)$ are attached in each endorsed transaction.

6.2. Anonymizing sender and recipient

In this subsection, we will describe approaches to anonymize sender and receiver. Consider in a channel, client/sender pk_s sends a transaction to a receiver pk_R.

6.2.1. Anonymizing sender

To anonymize sender we have employed an identity based group signature scheme [19]. A typical group signature scheme consists of 5 phases:

1. Setup: This algorithm is run by the group manager. The input to this algorithm is a security parameter k, and the output is system parameters $SP = (p, q, G_1, G_2, \mathrm{e}, H, H_1)$ along with master secret key a. The system parameter acts as the group public key and are known to all group members while master secret key is only known to the group manager.
2. Extract(SP, f_i, a): This protocol is run between user and group manager. The interaction between group manager and the user is assumed to be private and authenticated. The input to this protocol is identity of user f_i and master secret key a of group manager. At the end of the protocol, the user becomes the member of the group.
3. Sign(SP, m): This algorithm is run by the group member to sign a message m. The input to this algorithm is group public key SP, and a message m. The output is a group signature $\{A, B\}$.
4. Verify(A, B, f_i): This algorithm is run for the verification of group members. The input to this algorithm is A, B and f_i and the output is 1 in case of valid user and 0 in case of invalid user.
5. Reveal(a, A, B): This algorithm is run by the group manager to reveal the identity of group member. The input to this algorithm is master secret key a and group signature $\{A, B\}$. The output is identity of group member f_i.

Remark. In the context of hyperledger fabric, the group manager job will be replaced by the Membership Service Provider *MSP*, and the group will be formed by the client to generate a transaction during transaction proposal phase. We assume that the sender and dummy members used by the sender to form group are certified by the MSP to perform actions in the channel.

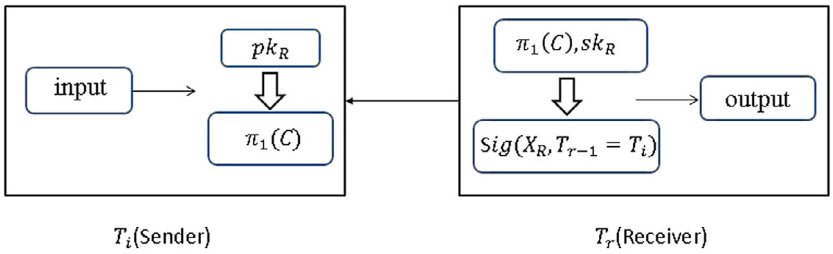

Fig. 1. Transaction of sender and receiver.

Authentication of sender: Group signature is used to anonymize the actual identity of the sender. Endorser contains the list of members authorized to access a specific channel, which is provided by the *MSP*. To authenticate the sender during the endorsing phase, the endorser will have to check if the members used to form the group are members of the channel. Even if a single member is not authorized to take part in channel activity, the endorser will not endorse the transaction. In this way, our approach enables anonymization of sender along with the authentication of the sender. Thus, the authentication of the sender to the endorser implies its authenticity to the receiver as well.

6.2.2. Anonymizing receiver

1. **Approach 1**

This Approach 1 aims to provide perfect unlinkability of public keys of users.

Global Setup

This algorithm first picks a large prime number p randomly , and then generates:

(a) Let G_1 be an additive cyclic group, and G_2 be a multiplicative cyclic group (both with prime order p)

(b) a bilinear pairing function $e : G_1 \times G_1 \rightarrow G_2$

(c) two hash functions as $H_1 : \{0, 1\}^* \rightarrow G_1$ and $H_2 : G_2 \rightarrow \{0, 1\}^{\lambda}$, where λ is the security parameter.

(d) a random generator $P \in G_1$. Finally, the algorithm outputs the System Parameter SP, where $\{e, G_1, G_2, H_1, H_2, P\} \in SP$

User Keys

Each user in the system generates their public private key pair as (pk, sk) where $pk = P^{sk}$. (pk_S, sk_S) denotes sender public private key pair and (pk_R, sk_R) denotes receiver public private key pair.

Transaction

As shown in Fig. 1, let us consider that there is one input and one output in a transaction T_i. Now we shall proceed to explain the transfer of the right of ownership of a transaction from sender to receiver.

(a) The sender generates a random number r, takes pk_R and sender's previous unspent transaction T_{i-1}. It now performs the following operations:

i. It computes $U = P^r$

ii. It computes $V_1 = {pk_R}^{r^2}$

iii. It computes $V = H_2(e(pk_R, V_1))$

Now, it forms a $C = (U, V)$. Now a problem π_1 is set with C as the input of the problem. The problem π_1 is defined as 'Knowing C find a suitable X_R so that given equality holds i.e. $H_2(e(U, X_R)) = V$.
Note: Unique r is chosen for every transaction.

(b) Firstly, R after receiving π_1 with the inputs C generates $X_R = {U^{sk_R}}^2$. Finally, it tries to solve the equation $H_2(e(U, X_R)) = V$. If R is able to solve the problem then it is clear that the transaction was intended for him.

(c) Later, receiver traverses each transaction and in polynomial time finds T_r intended to him. To spend T_i receiver generates a new transaction T_r and input to T_r is a signature $Sig = X_R, T_i$ to claim the ownership of the transaction T_i.
Note: In this case $T_{r-1} = T_i$ i.e. receiver's previous unspent transaction.

(d) Miner tracks T_{r-1} and then verify whether X_R is the appropriate solution to π_1 or not.

Correctness

$$H_2(e(U, X_R)) = V$$
$$H_2(e(U, {U^{sk_R}}^2)) = H_2(e(pk_R, V_1))$$

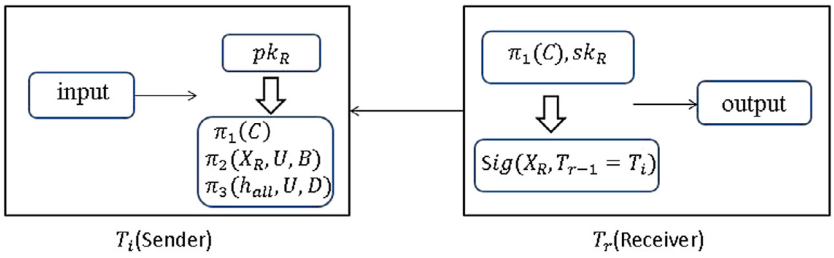

Fig. 2. Transaction of sender and receiver.

$$H_2(e(U, P^{r.sk_R{}^2})) = H_2(e(pk_R, pk_R{}^{r^2}))$$
$$H_2(e(P^{r_2}, P^{r.sk_R{}^2})) = H_2(e(P^{sk_R}, P^{sk_R.r^2}))$$
$$H_2(e(P, P)^{sk_R{}^2.r^2}) = H_2(e(P, P)^{sk_R{}^2.r^2})$$

Unlinkability

In approach 1, any adversary $\mathcal{A}$ cannot link the public keys of sender and receiver (pk_S, pk_R) in a transaction. The detailed security proof of Searchable Ciphertext Indistinguishability $(SC - IND)$ and Signature Indistinguishability are provided in Section 6.

Attacks

Approach 1 provides perfect unlinkability of public keys. However, it is still not immune to a malleability attack.

Malleability

$\mathcal{A}$ can modify the problem π_1 by retargeting the receiver's address, while keeping the signature of S same. Suppose $\mathcal{A}$ chooses a target transaction T_i, where $C_i = (U_i, V_i)$ where $U_i = P^r, V_i = H_2(e(pk_R, pk_R{}^{r^2}))$. Now $\mathcal{A}$ chooses a random number r' and a new public key $pk_{R'}$. Now $\mathcal{A}$ generates another $C' = (U', V')$ where $U' = (P^{r'})$ and $V' = H_2(e(pk_{R'}, pk_{R'}{}^{r'^2}))$. This modification from C_i to C' modifies T_i to T'. Later by providing signature $X_{R'}$ by using $sk_{R'}$ claims the ownership of the modified T'.

Hence, we can conclude that Approach 1 perfectly tackles with the weakness of leaking identity of receiver from X_R, however, it is still suffers from malleability attack. In Approach 3 we have tied the input of a transaction with its output, hence including a parameter which distinguishes output generated by the sender with an adversary.

2. **Approach 2**

This Approach 2 aims to provide perfect unlinkability and is resistant against malleability attacks.

Global Setup

The approach is similar as of Approach 1

User Keys

The approach is similar as of Approach 1.

Transaction

As shown in Fig. 2, let us consider that there is one input and one output in a transaction T_i. Now we shall proceed to explain the transfer of the right of ownership of a transaction from sender to receiver.

(a) The sender generates a random number r_2, takes pk_R and sender's previous unspent transaction T_{i-1}. It now performs the similar operations as in Approach 1 to generate (U, V_1, V, C). Now to provide non malleability property, it further performs a series of operations:

i. The signature *Sig* in this approach is similar as of Approach 2. Let us assume the signature of the sender to claim the ownership of its input be $X_S = P^{sk_S{}^2.r_1}$, which was generated with the help of secret key of sender sk_S and the public entity U_S, where $U_S = P^{r_1}$.

ii. Using (sk_S, U_S, r) as input sender generates an equality check problem π_2 which is stated as: 'Given the set of parameters (X_S, U, B) as input, check if the given equation holds'

$$e(X_S, U) = B$$

where $U = P^r, B = e(P^{sk_S{}^2}, U_S{}^r)$

iii. S computes $h_all = H_1(Data)$, where *Data* represent all content in T_i until now.

iv. S computes $D = h_{all}{}^r$. Now, using (U, h_{all}, D) as input sender generates an equality check problem π_3 which is stated as: 'Given set of parameters (h_{all}, U, D) as input, check if the given equation holds'

$$e(U, h_{all}) = e(P, D)$$

Problems π_1, π_2, π_3 are set by S in a transaction.

Note: h_{all} is not to be confused with T_i, as T_i is generated by the hash of all contents which include π_3 also.

(b) Later, receiver traverses each transaction and in polynomial time finds T_r intended to him. To spend T_i receiver generates a new transaction T_r and input to T_r is a signature $Sig = X_R, T_i$ to claim the ownership of the transaction T_i.
Note: In this case $T_{r-1} = T_i$ i.e. receiver's previous unspent transaction.

(c) Miner job is to track T_{r-1} and verify problems π_1, π_2, π_3, if any of the problem is not satisfied, then the validity of T fails and it is not appended to the blockchain.

Correctness

Correctness for π_1 is similar as given in Approach 1. Correctness for π_2 and π_3 are given below:

$$
\begin{aligned}
e(X_S, U) &= B \\
e(P^{{sk_S}^2.r_1}, P^r) &= e(P^{{sk_S}^2}, {U_S}^r) \\
e(P^{{sk_S}^2.r_1}, P^r) &= e(P^{{sk_S}^2}, P^{r_1})^r \\
e(P, P)^{{sk_S}^2.r_1.r} &= e(P, P)^{{sk_S}^2.r_1.r}
\end{aligned}
$$

$$
\begin{aligned}
e(U, h_{all}) &= e(P, D) \\
e(P^r, h_{all}) &= e(P, {h_{all}}^r) \\
e(P, h_{all})^r &= e(P, h_{all})^r
\end{aligned}
$$

Our Approach 2 provides perfect unlinkability of public keys and is immune to malleability attack.

Unlinkability

The proof is similar as given is approach 1.

Non Malleability

In our scenario, $\mathcal{A}$ cannot perform the malleability attack, if it tries to modify the content of T, then T will not be accepted by the verifiers, hence will not be appended to the blockchain. We have achieved this property in two steps:

(a) By validating the equation in π_2,

$$e(X_S, U) = B$$

we can ensure the correctness of U, as if suppose $\mathcal{A}$ attempts to create a fake $U' = P^{r'}$, but $\mathcal{A}$ does not know sk_S to create fake B' so that the equation gets satisfied. Also, $\mathcal{A}$ cannot modify X_S as the modification of X_S means the unsatisfaction of π_1.

(b) Now as U is fixed, miners will check the validity of the equation in π_3,

$$e(U, h_{all}) = e(P, D)$$

If $\mathcal{A}$ tries to modify h_{all} and create fake $h_{all}{}'$, then the equation will not satisfy. To satisfy the equation, $\mathcal{A}$ will have to modify D by recovering sk_S which is a discrete logarithmic problem which is assumed to be NP hard.

Hence, we conclude that Approach 2 satisfies unlinkability of public keys and provides non-malleability by tying input of transaction with its output. However, the approach is limited in a sense that apart from miners and adversary, the receiver is also not able to break the anonymity of sender. In Approach 3, we have included an additional parameter formed by sk_S and pk_R which helps the receiver to recover pk_S in polynomial time.

3. **Approach 3**

In this approach 3, we will aim to resolve the issue of tracking S by R, in such a way that proper anonymity of sender is maintained for everyone except R.

Global Setup

The approach includes all the parameters as of approach 1 in addition to another hash functions $H_3 : \{0, 1\}^* \rightarrow \{0, 1\}^*$ and $H_4 : G_1 \rightarrow \{0, 1\}^*$

User Keys

The approach is similar as of Approach 1.

Transaction

Let us consider that there is one input and one output in a transaction T_i. Now, we shall proceed to explain the transfer of the right of ownership of a transaction from sender to receiver.

(a) S along with the parameters as given in approach 2, also sets another value K_{trac} in the output of transaction which is computed as follows:

i. $Y = {pk_R}^{sk_S}$
ii. $k_1 = H_4(Y)$
iii. $K_{trac} = H_3(T_{i-1} \parallel k_1)$

(b) R with the help of K_{trac} and T_{i-1} performs the following operations:

i. R chooses a tentative sender's public key $pk_{S'} = P^{sk_{S'}}$.
ii. R computes $W = {pk_{S'}}^{sk_R}$
iii. R computes $k_2 = H_4(W)$
iv. R computes ${K_{trac}}' = H_3(T_{i-1} \parallel k_2)$
v. R compares ${K_{trac}}' = T_{i-1}$. The equation will only hold when $S = S'$. Hence S is identified.

Note: We assume that size of sender's list with R varies polynomially. Hence, with high probability R will guess relevant S in polynomial time.

Hence, we conclude that Approach 3 tracks S in polynomial time.

Wormhole Attack Prevention: From the above approaches, we can conclude that the sender and receiver anonymity is preserved. So with high probability, an inside adversary will not be able to guess the link between the transaction information and the end user identity. It cannot, therefore cannot leak the user information with any outside member of the channel.

7. Security proofs for anonymizing receiver

7.1. Security goals

Firstly, $\mathcal{A}$ should not be able to perform DoS attack on endorsers in order to block client transactions or to reduce network efficiency. Secondly, only modification of unconfirmed transactions by $\mathcal{A}$ must be discovered i.e. unconfirmed transactions should not be modified. Thirdly, $\mathcal{A}$ should not be able to leak the information of members inside the network to any member outside the channel or the network.

7.2. Assumption

In our security proofs, we have assumed that it takes exponential complexity to solve discrete logarithm is G_1 or G_T. Also, all the security proofs mentioned below are assumed to be secure under the random oracle model.

7.3. Signature unforgeability

There are two approaches for $\mathcal{A}$ to forge X_R. First is to recover sk_R which is as hard as solving discrete logarithm problem and another approach is to compute X_R using parameters U and V and applying $(FAPI - 1)$ as mentioned in [20]. To prove the signature unforgeability we present Theorem 1 along with its proof. Before presenting the theorem, we describe a random oracle used:

- $FAP - 1$ oracle: Let there be two elements P_1 and P_2, where $P_1 \in G_1$ and $P_2 \in G_1$ and P_3 be the output of $e : G_1 \times G_1 \rightarrow G_2$. The input to this oracle is P_1 and P_3 and the output is P_2 if P_2 exists, else output is 0.

Theorem 1. *Let us assume $\mathcal{A}$ solves $FAP - 1$ with non negligible advantage ${Adv_{\mathcal{A}_1}}^{FAP-1}(k)$ then, one can solve CDH problem with non negligible advantage ${Adv_{\mathcal{A}_1}}^{CDH}(k)$, where k is sufficiently large security parameter.*

Proof. Let (P, aP, bP) be the inputs to CDH problem.

1. $\mathcal{A}$ initially computes $P_3 = e(aP, bP)$.
2. $\mathcal{A}$ now sends aP and P_3 as input to $FAP - 1$ oracle and retrieves $P_2 = abP$.

Therefore, CDH is solved with non negligible advantage ${Adv_{\mathcal{A}_1}}^{CDH}(k)$, but we know that ${Adv_{\mathcal{A}_1}}^{CDH}(k)$ is negligible as it is a hard problem. Hence, our assumption is wrong and ${Adv_{\mathcal{A}_1}}^{FAP-1}(k)$ is negligible.
This concludes the proof. □

7.4. Unlinkability in ciphertext (UN-C)

$UN-C$ guarantees that the searchable ciphertexts $V = H_2(e(pk_R, V_1))$, where $V_1 = {pk_R}^{r^2}$ hides pk_R. We will use Theorem 1 prove $SC-IND$.

Theorem 2. *Let us assume $\mathcal{A}$ solves $UN-C$ with non negligible advantage ${Adv_{\mathcal{A}_1}}^{UN-C}(k)$ then one can solve CDH problem with non negligible advantage ${Adv_{\mathcal{A}_1}}^{CDH}(k)$, where k is sufficiently large security parameter.*

Proof. Let us assume $\mathcal{A}$ knows $P_3 = e(pk_R, V_1)$. $\mathcal{A}$ chooses a random public key $P_1 = pk_{R'}$. $\mathcal{A}$ sends P_1 and P_3 to $FAP-1$ oracle. If a point P_2 is returned then it is clear that the point is V_1 and $pk_{R'} = pk_R$(As there exists only single key pair which maps to P_3). If the output is 0, then $\mathcal{A}$ repeats the process until it finds a valid key pair which satisfies $e(P_1, P_2) = P_3$. In this way, $\mathcal{A}$ guesses pk_R with non negligible advantage ${Adv_{\mathcal{A}_1}}^{UN-C}(k)$.

Let (P, P_1, P_2) be the inputs to CDH problem, where $P_1 = aP, P_2 = bP$. $\mathcal{A}$ now sends aP and P_3 as input to $FAP-1$ oracle and retrieves $P_4 = abP$. Therefore, CDH is solved with non negligible advantage ${Adv_{\mathcal{A}_1}}^{CDH}(k)$, but we know that ${Adv_{\mathcal{A}_1}}^{CDH}(k)$ is negligible as it is hard problem. Hence, our assumption is wrong and ${Adv_{\mathcal{A}_1}}^{UN-C}(k)$ is negligible.
This concludes the proof. □

7.5. Limitations of the security model

In work [21], the authors revisited two foremost known schemes and systematically examined the unavoidable tradeoffs in the design criteria. The weaknesses and limitations of existing security models have been brought forth. This highlights the following. (a). Care needs to be exercised at the time of group formation while sending transactions, as smaller the group the higher the probability for the attacker to guess the sender. Hence, the scheme becomes vulnerable if the channel size is small. (b). Even if the sender is hidden in a group, the adversary can use the side channel information i.e. the specific user is more likely to send transaction during a specific time duration to deanonymize the sender.

8. Experiments and results

We have conducted various experiments to evaluate the performance of hyperledger network (a). before the DoS attack i.e. when DoS attack was not been performed on endorsers (b). After committing the DoS attack on endorsers. The performance parameters used are Throughput and Latency. The throughput represents the number of transactions updated in the ledger per second whereas the latency represents the time taken for a transaction from transaction proposal phase to transaction commit phase. The overall performance is the product of performance of throughput and latency.

8.1. Experimental setup

All the experiments were performed using the caliper tool to analyse the performance of hyperledger fabric 1.0.5. Our experimental setup consists of 2 organizations A and B. Each organization contains 2 peers. It is assumed that both peers can act as endorsers. Also, setup includes 1 orderer which uses solo ordering service. Couch DB is used as our state database to store the state of clients. All experiments we performed on Intel(R) Core(TM) i3-3220 CPU @ 3.30 GHz with 2GB RAM running Ubuntu 14.04 LTS with 3.13.X kernel version. The procedure used to calculate each timing was to run each execution 200 times and get the mean (μ) and standard deviation (σ) of the performance parameters(throughput and latency). The mean is calculated to improve the accuracy of the results while standard deviation signifies how far values are spread out from their average value. We used a chain code open() to analyse our experiments. Initially, the client proposes a transaction to open a new account, later the endorser runs open() and sends back the response to the client. After this, the transaction is updated after being passed to orderer and the validator. The policy of chain code used in our setup is "at least 1 peer in an organization must endorse a transaction". We launched the DoS attack in three scenarios (a). DoS on 1 peer of organization A. (b). DoS on 2 peers of organization A. (c). DoS on 2 peers of organization A and 1 peer of organization B. The tool used to perform DoS attack is hping3 in which we continuously flood syn packets to a target peer. The unit used to represent throughput and send rate is transaction per second(tps).

8.2. Results and discussion

1. **Before DoS attack**: In Fig. 3, we have shown the variation of throughput with the increase in send rate/tpr. Here x-axis represents transaction proposal rate tpr i.e. rate at which transactions are being proposed whereas y-axis represents throughput i.e. rate at which transactions are committed to ledger. As we can see the throughput is 47 $tps(\mu = 47\ tps, \sigma = 0.47\ tps)$ when send rate is 50 tps while 125.5 $tps(\mu = 125.5\ tps, \sigma = 2.17\ tps)$ when send rate is 150 tps. This shows the increase of throughput with the increase in send rate, which is obvious as the transaction proposal rate increases, the quantity of transaction being updated in the ledger.
   Fig. 4 shows the variation of latency with the increase in send rate/tpr. Here, x-axis represents tpr whereas y-axis represents latency i.e. time taken from application sending the transaction proposal to the transaction commit. As

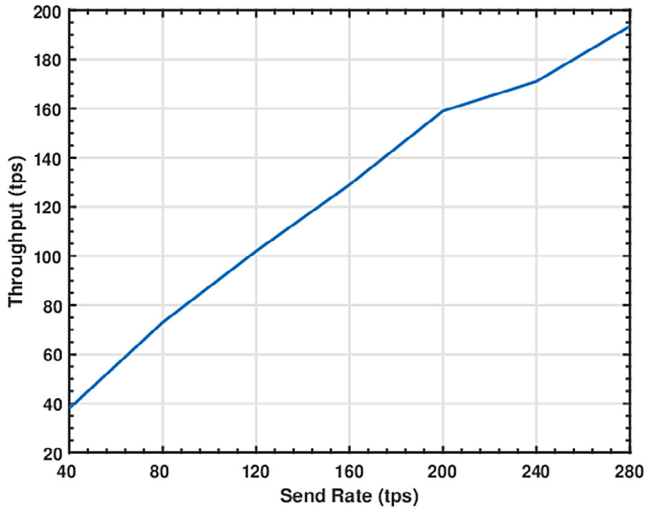

Fig. 3. Impact of *tpr* on throughput.

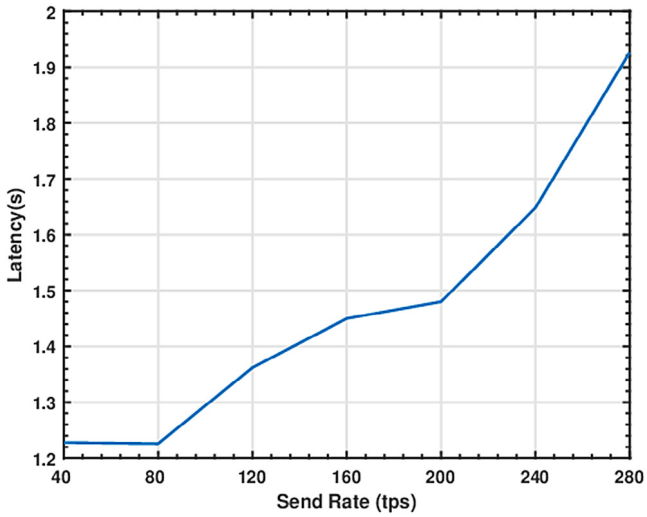

Fig. 4. Impact of *tpr* on latency.

it can be observed, latency is 1.318 s($\mu = 1.318\,s, \sigma = 0.052$ s) when send rate is 50 *tps* while 1.442 *tps*($\mu = 1.442\,s, \sigma = 0.049$ s) when send rate is 150. This shows the increase in latency with the increase in send rate which is logical as the transaction rate increases, the endorsing process and the queue of transaction formed in ordering service delays the transaction updation process.

2. **After DoS**: Fig. 5 *x*-axis represents the send rate/*tpr*, whereas *y*-axis depicts the effect in throughput while performing DoS on different peers. It can be seen that throughput at send rate 123 *tps* is 125 *tps*, while the value is dropped to 123 *tps*, 100 *tps*, and 96 *tps* after applying DoS on 1 peer of A, 2 peers of A, 2 peers of A and 1 peer of B respectively. Hence, the observation is (a). applying DoS attack on peers reduces the throughput, (b). increasing the target peers for DoS further reduces the throughput drastically.
   In Fig. 6, the *x*-axis represents the send rate/*tpr*, and the *y*-axis represents the effect in latency while performing DoS on different peers. It can be seen that latency at 125 *tps* is 1.396 s in the normal case, while the latency rises to 1.79 *s*, 2.44 *s*, 2.5 *s* after applying DoS attack on 1 peer of A, 2 peers of A, 2 peers of A and 1 peer of B respectively. We can conclude that (a). impact of DoS on peers increases the latency, (b). increase in target peers for DoS will result in the increase in latency.

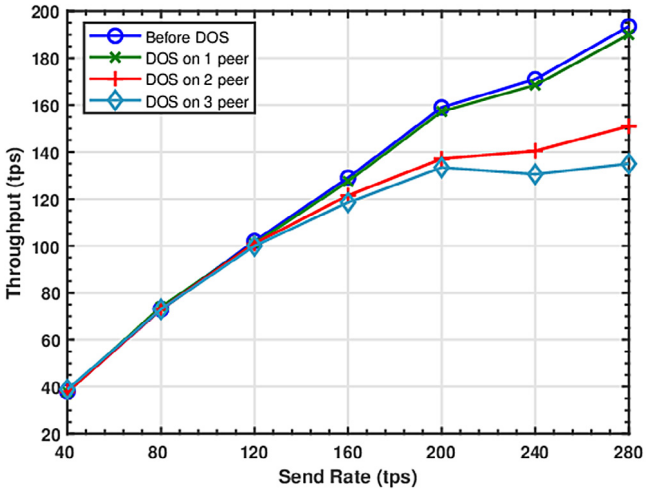

Fig. 5. Impact of DoS on throughput.

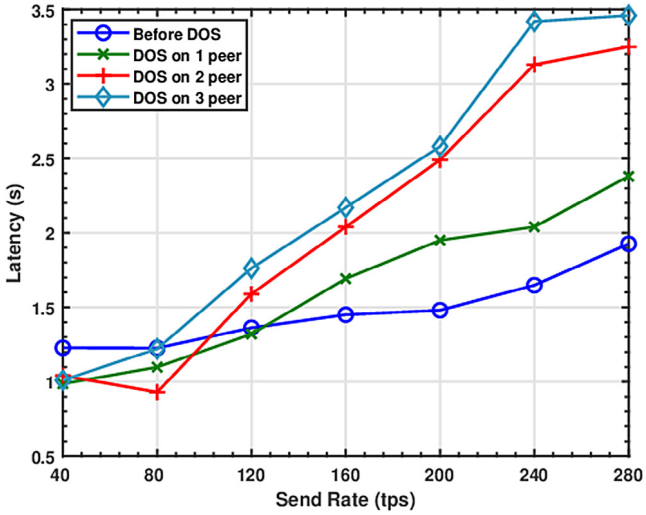

Fig. 6. Impact of DoS on latency.

9. Conclusion

In this paper, we evaluated the effect of the DoS attack on the endorsers; the results indicate that DoS attack has a significant effect on the network efficiency. The throughput is reduced followed by the increase in latency. We proposed and evaluated two alternative approaches to anonymize the endorsing process, (a). using VRF, (b). using pseudonyms. In the first approach, there is a tradeoff between security and efficiency, as all the users will act as tentative endorsers, therefore every user must have chain code with it which reduces network efficiency. The second approach follows the first approach with respect to the tradeoff, as endorsers will have to interact offline with MSP for certification of pseudonyms, thereby reducing network efficiency. One more level of anonymization is provided in hyperledger by anonymizing sender and receiver inside a private channel. Hence, our successive approaches ensure that hyperledger fabric is immune against DoS attack and wormhole attack. This is also shown by security proofs of Signature Unforgeability and Unlinkability in Ciphertext(UN-C). To increase the efficiency of network in the scheme is still for future work.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008.
[2] G. Wood, Ethereum: A Secure Decentralised Generalised Transaction Ledger, Ethereum project yellow paper, 151, 2014, pp. 1–32.
[3] E.B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, M. Virza, Zerocash: Decentralized anonymous payments from bitcoin, in: 2014 IEEE Symposium on Security and Privacy (SP), IEEE, 2014, pp. 459–474.
[4] I. Miers, C. Garman, M. Green, A.D. Rubin, Zerocoin: Anonymous distributed e-cash from bitcoin, in: Security and Privacy (SP) 2013 IEEE Symposium on, IEEE, 2013, pp. 397–411.
[5] K.M. Alonso, Monero-Privacy in the blockchain.
[6] G. Greenspan, MultiChain private blockchain—White paper, 2015. URl: http://www.multichain.com/download/MultiChain-White-Paper.pdf.
[7] J. Sousa, A. Bessani, M. Vukolic, A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform, in: 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN, IEEE, 2018, pp. 51–58.
[8] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, Hyperledger fabric: a distributed operating system for permissioned blockchains, in: Proceedings of the Thirteenth EuroSys Conference, ACM, 2018, p. 30.
[9] D. He, N. Kumar, J.H. Lee, Privacy-preserving data aggregation scheme against internal attackers in smart grids, Wirel. Netw. 22 (2) (2016) 491–502.
[10] S. Micali, ALGORAND: the efficient and democratic ledger, CoRR abs/160701341 (2016).
[11] M. Ma, D. He, N. Kumar, K.K.R. Choo, J. Chen, Certificateless searchable public key encryption scheme for industrial Internet of Things, IEEE Trans. Ind. Inf. 14 (2) (2017) 759–767.
[12] D. Wang, W. Li, P. Wang, Measuring two-factor authentication schemes for real-time data access in industrial wireless sensor networks, IEEE Trans. Ind. Inf. 14 (9) (2018) 4081–4092.
[13] D. Wang, Q. Gu, H. Cheng, P. Wang, The request for better measurement: A comparative evaluation of two-factor authentication schemes, in: Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, ACM, 2016, pp. 475–486.
[14] D. Wang, P. Wang, Two birds with one stone: Two-factor authentication with security beyond conventional bound, IEEE Trans. Dependable Secure Comput. 15 (4) (2016) 708–722.
[15] X. Huang, Y. Xiang, A. Chonka, J. Zhou, R.H. Deng, A generic framework for three-factor authentication: Preserving security and privacy in distributed systems, IEEE Trans. Parallel Distrib. Syst. 22 (8) (2010) 1390–1397.
[16] R. Amin, S.H. Islam, G.P. Biswas, M.K. Khan, N. Kumar, An efficient and practical smart card based anonymity preserving user authentication scheme for TMIS using elliptic curve cryptography, J. Med. Syst. 39 (11) (2015) 180.
[17] S. Chatterjee, S. Roy, A.K. Das, S. Chattopadhyay, N. Kumar, A.V. Vasilakos, Secure biometric-based authentication scheme using chebyshev chaotic map for multi-server environment, IEEE Trans. Dependable Secure Comput. 15 (5) (2016) 824–839.
[18] D. He, N. Kumar, H. Wang, L. Wang, K.K.R. Choo, A. Vinel, A provably-secure cross-domain handshake scheme with symptoms-matching for mobile healthcare social network, IEEE Trans. Dependable Secure Comput. 15 (4) (2018) 633–645.
[19] S. Han, J. Wang, W. Liu, An efficient identity-based group signature scheme over elliptic curves, in: Universal Multiservice Networks, Springer, Berlin, Heidelberg, 2004, pp. 417–429.
[20] S. Galbraith, F. Hess, F. Vercauteren, Aspects of pairing inversion, IEEE Trans. Inform. Theory 54 (12) (2008) 5719–5728.
[21] D. Wang, D. He, P. Wang, C.H. Chu, Anonymous two-factor authentication in distributed systems: certain goals are beyond attainment, IEEE Trans. Dependable Secure Comput. 12 (4) (2014) 428–442.