Liu, Gao; Dong, Huidong; Yan, Zheng

B4SDC: A Blockchain System for Security Data Collection in MANETs

Published in:
2020 IEEE International Conference on Communications, ICC 2020 - Proceedings

DOI:
10.1109/ICC40277.2020.9149192

Published: 01/06/2020

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:
Liu, G., Dong, H., & Yan, Z. (2020). B4SDC: A Blockchain System for Security Data Collection in MANETs. In *2020 IEEE International Conference on Communications, ICC 2020 - Proceedings* Article 9149192 (IEEE International Conference on Communications). IEEE. https://doi.org/10.1109/ICC40277.2020.9149192

B4SDC: A Blockchain System for Security Data Collection in MANETs

Gao Liu[a], Huidong Dong[a] and Zheng Yan[a,b]
[a]State Key Laboratory on Integrated Services Networks and School of Cyber Engineering, Xidian University, Xi'an, China
[b]Department of Communications and Networking, Aalto University, Espoo, Finland
zyan@xidian.edu.cn

***Abstract*—Security-related data collection is an essential part for attack detection and security measurement in Mobile Ad Hoc Networks (MANETs). Due to no fixed infrastructure of MANETs, a detection node playing as a collector should discover available routes to a collection node for data collection. Notably, route discovery suffers from many attacks (e.g., wormhole attack), thus the detection node should also collect security-related data during route discovery and analyze these data for determining reliable routes. However, few literatures provide incentives for security-related data collection in MANETs, and thus the detection node might not collect sufficient data, which greatly impacts the accuracy of attack detection and security measurement. In this paper, we propose B4SDC, a blockchain system for security-related data collection in MANETs. Through controlling the scale of RREQ forwarding in route discovery, the collector can constrain its payment and simultaneously make each forwarder of control information (namely RREQs and RREPs) obtain rewards as much as possible to ensure fairness. At the same time, B4SDC avoids collusion attacks with cooperative receipt reporting, and spoofing attacks by adopting a secure digital signature. Based on a novel Proof-of-Stake consensus mechanism by accumulating stakes through message forwarding, B4SDC not only provides incentives for all participating nodes, but also avoids forking and ensures high efficiency and real decentralization at the same time. We analyze B4SDC in terms of incentives and security, and evaluate its performance through simulations. The thorough analysis and experimental results show the efficacy and effectiveness of B4SDC.**

***Index Terms*—MANETs, security-related data collection, incentive mechanism, blockchain**

I. Introduction

MANET suffers from different attacks due to self-organization [1]. In order to provide a secure and high-quality networking service, security-related data collection becomes essential for network attack detection and security measurement. Security-related data, in short security data, are the data that can be used to discover network threats and measure its security. In data collection, after receiving the request of a collector or detection node, collection nodes send it sensed security data [2]. Due to no fixed infrastructure in MANETs, nodes cooperate to forward the request and security data. In order to ensure efficient collection, a collector should discover available routes to collection nodes through route discovery, so that the request and security data can be transmitted via these routes. However, the route discovery suffers from various attacks, e.g., wormhole and rushing attacks [1]. Existing detection mechanisms can help a collector to detect these attacks and select reliable routes, but the collector should analyze security data provided by the forwarders of control information for making decisions, e.g., the timestamps of receiving and sending a message, the location of forwarders. Unfortunately, few studies provide incentives for security data collection in MANETs. Forwarders might not be willing to sense and provide security data due to extra overload, selfishness, etc., so that the collector could not collect sufficient security data for detection. As a result, the accuracy of threat detection and security measurement cannot be ensured.

Current incentive mechanisms mainly make use of reputation and micropayment systems [3]. However, the present studies in this field are still facing a number of issues to be applied into security data collection in MANETs. First, these mechanisms do not consider spoofing attacks that an attacker launches for maximizing its profits. Second, reputation systems might not resist collusion attacks raised by the selfish nodes with a high reputation and do not specify the type of incentive (e.g., what profit can be brought by a high reputation.) Third, micropayment systems allow a collector to pay to collection nodes and forwarders for their cooperation and contributions, but most of them require a trusted third party to manage debiting and crediting accounts. This kind of design is obviously infeasible for MANETs since such a trusted third party is hard to be deployed. Fourth, in some micropayment systems for route discovery, a source node pays control information forwarders, but cannot constrain its payment, which is caused by an uncontrolled RREQ forwarding scale [3]. In addition, some existing systems allow the source node to only pay the forwarders in discovered routes, which is unfair to the nodes that have participated in route discovery but are not in the routes [4].

In order to solve above issues, a distributed incentive system for security data collection is highly expected, which should resist spoofing, collusion and excessive forwarding, and feature fairness, and should not rely on a trusted third party. We found that blockchain is a candidate technology to help achieving the above goals due to its advantages, e.g., transparency, immutability, and self-organization. In a blockchain system, a miner collects transactions, generates a block and provides it to other miners with a proof of work (e.g., the proof of computing and storage) to gain the majority of acceptance, which is called consensus. In general, the incentive for miners is provided in a form of digital tokens, e.g., bitcoin [5]. At the same time, the transaction can help

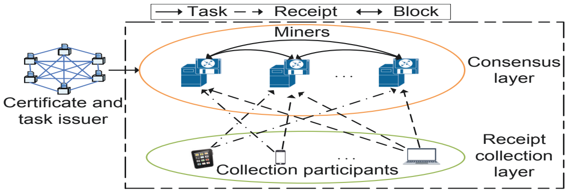


Fig. 1. System model

incenting security data collection.

However, the blockchain itself is still facing many technical challenges, namely forking, low efficiency and a trend of centralization. Based on the current literature review [6], consensus mechanisms mainly include Proof of Work (PoW), Byzantine Faulty Tolerant (BFT), Proof of Sake (PoS), Proof of Useful Work (PoUW) and Trees and Directed Acyclic Graphs (DAGs). PoW and PoUW take the risk of temporary forking due to network latency. PoW wastes a lot of resources since it is a meaningless task. Its transaction confirmation time is long, thus negatively impacting its throughput. PoW and PoUW take the risk of centralization due to the outsourceability of tasks. Because PoS consumes almost no resources, a miner might create two blocks to cause forking. Many BFT based consensus mechanisms focus on scalability, but they provide no incentives for miners. Trees and DAGs can replace the chain structure of blockchain for ensuring a high throughput and avoiding double spending, but some trees and DAGs based consensus mechanisms [7] also employ PoW, thus suffering from the same problems of PoW, namely forking, low efficiency and the risk of centralization.

In this paper, we propose B4SDC, a blockchain system that provides incentives for security data collection in a distributed way in MANETs. Through controlling the scale of RREQ forwarding in route discovery, a collector can constrain its payment and simultaneously make each forwarder of control information obtain rewards as much as possible to ensure fairness. At the same time, B4SDC avoids collusion attacks with cooperative receipt reporting, and spoofing attacks by adopting a secure digital signature. Based on a novel Proof-of-Stake consensus mechanism by accumulating stakes through message forwarding, B4SDC not only provides incentives for all participating nodes, but also avoids forking and ensures high efficiency and real decentralization at the same time. With above ways, B4SDC successfully avoids collusion and spoofing attacks, allows collectors to control their maximum payments, and ensures fairness for collection participants as much as possible. It also solves current blockchain systems' main problems, namely forking, low efficiency and centralization. Specifically, the contributions of this paper can be summarized as follows.

(1) B4SDC provides incentives for collection participants. It encourages nodes to forward control information that includes security data in route discovery. After routes are discovered, the nodes in selected routes are incented to forward the request of a collector and the security data of collection nodes.

(2) B4SDC removes the need for a trusted third party in many micropayment systems by adopting blockchain. It adopts a secure digital signature for signing sent messages, thus avoiding spoofing attacks. At the same time, B4SDC allows collection participants to cooperatively report their received receipts to miners for gaining rewards, thus resisting against collusion attacks and ensuring fairness for all collection participants as much as possible. In addition, a collector can constrain the scale of RREQ forwarding in the route discovery, thus it can balance its budget.

(3) B4SDC provides a novel consensus mechanism. Block creation is proposed to ensure the distribution and efficiency of blockchain by avoiding the simultaneous generation of many valid blocks and reducing communication burdens. Single block winner selection is performed to make B4SDC free from forking when multiple valid blocks are created at the same time.

II. Problem Statements

A. An Overview of Route Discovery

Mainstream routing protocols adopt route discovery [8] for data transmission, such as DSR and AODV protocols. In the route discovery, a source node floods a RREQ including its and destination node's addresses. When a node receives the RREQ and it is not the destination node and has no routes to the destination node, it adds its address into the RREQ and then forwards this RREQ. If the destination node or an intermediate node having routes to the destination node receives the RREQ, it creates a RREP including a whole route path and sends the RREP to the source node in the reverse of the path.

B. System Model

As shown in Fig. 1, B4SDC involves three types of entities: certificate and task issuer, collection participant and miner.

The certificate and task issuer is responsible for issuing certificates and tasks, which can be a blockchain system for key management, PKI or identity management to ensure decentralization [9]. It does its work honestly for rewards since the published certificates and tasks should obtain the consensus of B4SDC blockchain.

In the receipt collection layer, collection participants involve collectors, forwarders and collection nodes. The collectors discover routes to collection nodes with route discovery, and select some reliable routes to publish requests and receive security data from collection nodes, since the route discovery allows the collectors to collect security data from each forwarder for detecting potential attacks and obtaining reliable routes. The forwarders help forwarding received messages including security data. Each collection node and forwarder can use received messages as receipts and share these receipts to miners. In this layer, each node is not fully trusted since it might forge receipts for rewards.

In the consensus layer, after collecting sufficient receipts and receiving a task, the miners create and publish blocks for consensus. The majority of miners accept one block as the next block of blockchain. It is hard to ensure each miner is trusted since the miner might be hacked or compromised.

C. Assumptions

Suppose an adversary can only control a small number of miners. The time of all the miners is supposed to be synchronized with the help of public GPS signals [10] or a

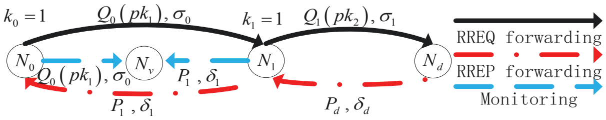


Fig. 2. A simple example of route discovery

public time blockchain [11]. Note that time synchronization can be achieved in the level of nanoseconds. Although this assumption restricts B4SDC, it is justified compared to secure incentive provisions for security data collection. In addition, miners cannot forge timestamps, since miners in the network are monitored by their neighbors and nodes can employ some lightweight methods (e.g., wormhole attack detection [10]) to detect this kind of misbehavior.

III. B4SDC Design

In this section, we describe the design of B4SDC, which contains receipt collection, and a novel consensus mechanism that contains block creation, block winner selection and an incentive mechanism.

A. Receipt Collection

Before the consensus, receipt collection should be performed. It consists of *Node Registration*, *Receipt Generation*, and *Receipt Sharing*.

1) Node Registration

Each node N_i generates a pair of private/public keys (sk_i, pk_i) based on ECC, and uses pk_i to register at the certificate and task issuer N_{TI}.

N_{TI} equipped with a pair of public/private keys (sk_{TI}, pk_{TI}) generates the certificate $CE_i = \{pk_{TI}, sig_{sk_{TI}}(pk_i)\}$ with ECDSA and issues it to N_i.

2) Receipt Generation

Route Discovery

1. Through neighbor discovery, each node N_i records the public key list NL_i of its neighbors, and periodically publishes $SNL_i = \{CE_i, pk_i, NL_i, t_i, sig_{sk_i}(NL_i, t_i)\}$.

2. Based on collection strategies, the collector N_0 attempts to discover routes to the selected collection node N_d for security data collection. In order to avoid the collector's unreliable payment (e.g., double spending), N_0 should send a blockchain (not N_{TI}) the deposit message $TX_{ct} = \{CE_0, pk_0, UID, NU_0\}$ to deposit NU_0 tokens for the unique identity UID of RREQ in advance. If N_0 makes sure that its message TX_{ct} is inserted into blockchain, it can use NU_0 tokens as the budget for route discovery. Then it determines appropriate Time To Live (TTL) and $k = \{k_h : h = 0, \ldots, TTL-1\}$ due to its budget [12]. k_h is the number of neighbors that each node at hop h selects as the receivers of RREQ.

3. N_0 randomly selects its k_0 neighbors $N_{0_1}, \ldots, N_{0_{k_0}}$ as the receivers of its RREQ, whose public keys are $pk_{0_1}, \ldots, pk_{0_{k_0}}$ respectively. Then it computes $BI = \{CE_0$, pk_0, UID, pk_d, TTL, k, $sig_{sk_0}(UID, pk_0, pk_d, TTL, k)\}$, and obtains its RREQ $Q_0 = \{BI$, $h = 0$, RL, pk_{0_1}, $\ldots$, $pk_{0_{k_0}}$, SD, $t_0'\}$ with the signature $\sigma_0 = sig_{sk_0}(Q_0)$, where pk_d is the public key of destination node N_d, RL the route list, SD the security data list including N_0's security data, and t_0' the time of sending Q_0. If $|NL_0| < k_0$, N_0 selects its all neighbors as RREQ receivers.

4. After receiving $Q_i = \{BI, CE_i, pk_i, h, RL, pk_{i_1}, \ldots, pk_{i_{k_i}}, SD, t_i'\}$ and $\sigma_i = sig_{sk_i}(Q_i)$ from N_i for the first time, N_{i+1} can check that the collector does not make an unreliable payment by accessing the blockchain, its public key pk_{i+1} belongs to $\{pk_{i_1}, \ldots, pk_{i_{k_i}}\}$ and $h < TTL$, and verifies CE_0 in BI, CE_i and σ_i. If all verifications hold, N_{i+1} saves Q_i and σ_i as receipts, increases h by 1, inserts its public key into RL, selects k_{i+1} neighbors whose public keys are $pk_{i+1_1}, \ldots, pk_{i+1_{k_{i+1}}}$, adds its sensed security data into SD, and generates $Q_{i+1} = \{BI, CE_{i+1}, pk_{i+1}, h, RL, pk_{i+1_1}, \ldots, pk_{i+1_{k_{i+1}}}, SD, t_{i+1}'\}$ and $\sigma_{i+1} = sig_{sk_{i+1}}(Q_{i+1})$. Finally, it broadcasts Q_{i+1} and σ_{i+1}.

In order to prevent N_i from selecting few receivers by deliberately inserting less than k_i public keys in the RREQ for saving resources, each node N_v can serve as an observer for monitoring the behaviors of its neighbor. If N_v finds that the neighbor list NL_i of its neighbor N_i satisfies $|NL_i| \geq k_i$ and N_i forwards Q_i to less than k_i neighbors, it can send the blockchain the report $TX_v = \{CE_v$, pk_v, $RP_v = (SNL_i$, $Q_i, \sigma_i)$, $SIG_v = sig_{sk_v}(RP_v)\}$, thus obtaining rewards by deducting N_i's rewards. Therefore, nodes are incented to provide security data and forward the RREQ.

5. When the destination node N_d or the intermediate node N_w having a route to the destination node receives the RREQ with a signature, it can check that the collector makes a reliable payment by accessing the blockchain, its public key belongs to the public key list in the RREQ and $h < TTL$. If all verifications hold, N_d or N_w saves the received RREQ and corresponding signature as receipts. Then N_d inserts its sensed security data into SD, and generates the RREP $P_d = \{BI, CE_d, pk_d, RL, SD, sig_{sk_d}(BI, RL)\}$ with the signature $\delta_d = sig_{sk_d}(P_d)$. N_w adds its sensed security data into SD, and creates the RREP $P_w = \{BI, CE_w, pk_w, RL, SD, sig_{sk_w}(BI, RL)\}$ with $\delta_w = sig_{sk_w}(P_w)$. Note that RREP's RL is a whole route path from N_0 to N_d, and SD has all security data recorded in the received RREQ. N_d or N_w forwards its RREP with the signature in the reverse direction of route from N_0 to N_d.

6. When N_i en route (derived from RL) receives a RREP and a signature from N_{i+1}, namely $P_{i+1} = \{BI, CE_w, CE_{i+1}, pk_w, pk_{i+1}, RL, SD, sig_{sk_w}(BI, RL)\}$ or $P_{i+1} = \{BI, CE_d, CE_{i+1}, pk_d, pk_{i+1}, RL, SD, sig_{sk_d}(BI, RL)\}$ with $\delta_{i+1} = sig_{sk_{i+1}}(P_{i+1})$, it verifies the involved signatures.

If all verifications hold, N_i saves P_{i+1} and δ_{i+1} as receipts, then inserts its sensed security data into SD, and generates and sends P_i and δ_i to N_{i-1} en route. When N_0 receives valid P_1 and δ_1, the common neighbor N_v of N_0 and N_1 can collect P_1 and δ_1, and share $TX_v = \{CE_v, pk_v, RP_v = (SNL_0, SNL_1, P_1, \delta_1), SIG_v = sig_{sk_v}(RP_v)\}$ to the blockchain for rewards. Fig. 2 shows a simple example of route discovery. N_0 sets $TTL = 2$.

Security Data Item Collection

1. When the collector N_0 has discovered routes to collection nodes, it can determine the reliable route PA_{d_j} to each collection node N_{d_j} for security data collection (e.g., by

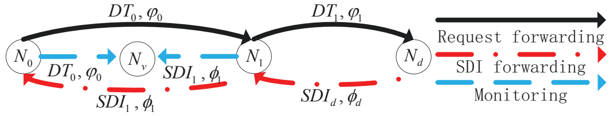


Fig. 3. A simple example of security data item collection

adopting wormhole attack detection). These collection nodes and corresponding routes are denoted as $\{N_{d_0}, \dots, N_{d_c}\}$ and $PA = \{PA_{d_0}, \dots, PA_{d_c}\}$ respectively. The collector also deposits some tokens in advance in order to avoid unreliable payments. In detail, it sends the blockchain the deposit message $TX_{cc} = \{CE_0, pk_0, DI, NU_1\}$ to deposit NU_1 tokens. If the collector knows TX_{cc} has been added into the blockchain, it can start to generate and broadcast the request with the unique identity DI.

2. N_0 generates and broadcasts the request $DT_0 = \{CE_0, pk_0, DI, PA, sig_{sk_0}(DI, PA)\}$ with the signature $\varphi_0 = sig_{sk_0}(DT_0)$, where DI is the unique identity of request.

3. After receiving $DT_i = \{DT_0, CE_i, pk_i\}$ and $\varphi_i = sig_{sk_i}(DT_i)$ from the neighbor N_i, the node N_{i+1} can check if the collector makes a reliable payment by accessing the blockchain, uses DT_i's PA to check it is a legitimate receiver, and verifies involved signatures. If all verifications hold, N_{i+1} considers DT_i and φ_i as receipts, then creates and broadcasts the new request $DT_{i+1} = \{DT_0, CE_{i+1}, pk_{i+1}\}$ with the signature $\varphi_{i+1} = sig_{sk_{i+1}}(DT_{i+1})$.

4. When the destination node N_{d_j} receives the request from N_0, it checks the payment of collector is reliable by accessing blockchain, then checks it is a legitimate receiver and verifies the validity of request through involved signatures. If all verifications hold, N_{d_j} considers the received request and signature as receipts, then creates the Security Data Item (SDI) $SDI_{d_j} = \{DT_0, CE_{d_j}, pk_{d_j}, SD_{d_j}, sig_{sk_{d_j}}(DT_0, SD_{d_j})\}$ with the signature $\phi_{d_j} = sig_{sk_{d_j}}(SDI_{d_j})$, and then forwards SDI_{d_j} and ϕ_{d_j} to N_0 in the reverse of route from N_0 to N_{d_j}.

5. If N_i receives $SDI_{i+1} = \{SDI_{d_j}, CE_{i+1}, pk_{i+1}\}$ and $\phi_{i+1} = sig_{sk_{i+1}}(SDI_{i+1})$ from N_{i+1}, it verifies the validity of SDI_{i+1} and ϕ_{i+1}. If all verifications hold, N_i saves SDI_{i+1} and ϕ_{i+1} as receipts, then generates and forwards $SDI_i = \{SDI_{d_j}, CE_i, pk_i\}$ with $\phi_i = sig_{sk_i}(SDI_i)$ in the reverse of route from N_0 to N_{d_j}. As a result, N_0 can obtain the security data SD_{d_j} from N_{d_j}. When N_0 receives valid SDI_1 and ϕ_1, the common neighbor N_v of N_0 and N_1 can collect SDI_1 and ϕ_1, and send $TX_v = \{CE_v, pk_v, RP_v = (SNL_0, SNL_1, SDI_1, \phi_1), SIG_v = sig_{sk_v}(RP_v)\}$ to the blockchain for rewards. In Fig. 3, a simple example of security data item collection is shown. DT_i includes the route $N_0 \to N_1 \to N_d$.

Receipt Sharing

1. Each node N_v broadcasts $TX_v = \{CE_v, pk_v, RP_v, SIG_v\}$, where $RP_v = \{Q_{i_v}, \sigma_{i_v}\}$, $\{P_{i_v}, \delta_{i_v}\}$, $\{DT_{i_v}, \varphi_{i_v}\}$, $\{SDI_{i_v}, \phi_{i_v}\}$ or $\{SNL_i, Q_i, \sigma_i\}$, and $SIG_v = sig_{sk_v}(RP_v)$ is N_v's signature on RP_v.

2. Miners collect and verify TX_{ct} with new UID, TX_{cc} with new DI and TX_v, since a miner will obtain rewards in the case that it successfully adds these messages into the blockchain.

B. Consensus Mechanism

We propose a novel consensus mechanism for B4SDC, which aims to solve the three main technical challenges of blockchain. Based on the last block of blockchain, miners in different locations compete to create the next block, and only one of published valid blocks is accepted as the next block. The consensus mechanism consists of block creation, block winner selection and an incentive mechanism.

Block Creation

N_{TI} publishes the task $T = \{pk_{TI}, TID, n_b, n_{bthr}, n_{thr}, TAG, \theta, sig_{sk_{TI}}(TID, n_b, n_{bthr}, n_{thr}, TAG, \theta)\}$ periodically, which triggers the block creation. TID is the unique identity of T. n_b is the number of the latest referenced blocks of the blockchain. n_{bthr} represents the threshold of the number of blocks that a miner creates in the latest n_b blocks of blockchain. n_b and n_{bthr} prevent a powerful miner from generating the majority of blocks of blockchain for controlling the blockchain, thus ensuring the decentralization. n_{thr} is the threshold of the number of receipts that a miner should insert into its created block, which ensures sufficient receipts can be inserted into the blockchain. TAG is a difficulty value, which is set to avoid the simultaneous creation of many valid blocks and thus helps reducing communication burdens. θ represents the time window that a miner waits for more valid blocks after receiving the first valid block.

Algorithm 1 shows the creation of block K. TX_{mr} means that the miner uses the age of AM tokens for making it become the creator of the next block. We denote the age of one token as the blockchain length from the block involving this token to the block K, and the miner uses tokens whose ages are the biggest. If TX_v is inserted into blockchain, this suggests that a collector gives rewards to others and thus the age of related tokens goes to 0. $w(AM)$ represents the weight of age of AM tokens for adjusting the difficulty of the miner to create a valid block. The bigger the age of used AM tokens is, the easier the miner generates a valid block, since it is applied to control the difficulty of block generation. If a miner generates a valid block successfully and has not received valid blocks from others, it publishes its created block. When another miner receives this block, it can verify the validity of block by running **Algorithm 1**.

Block Winner Selection

Multiple valid blocks might be created at the same time. We design how to choose one of them as the block winner (i.e., the next block). Aiming to summarize selection rules, we show determining the winner from two valid block candidates B_{c0} and B_{c1}. They have two timestamps t_{c0} and t_{c1}, and $n0_{c0}$ and $n0_{c1}$ receipts, respectively. In addition, both blocks' creators have created $n_{pk_{c0}}$ and $n_{pk_{c1}}$ blocks within the latest n_b blocks of blockchain. If $t_{c0} \neq t_{c1}$, the block whose timestamp is earlier is considered as the winner for ensuring the efficiency of block creation. When $t_{c0} = t_{c1}$ and $n_{pk_{c0}} \neq n_{pk_{c1}}$, the block whose creator has generated fewer blocks within the latest n_b blocks of blockchain is regarded as the winner for preventing a miner from controlling the blockchain by generating the majority of blocks of blockchain. If $t_{c0} = t_{c1}$,

```
Input: $T = \{pk_{TI}, TID, n_b, n_{bthr}, n_{thr}, TAG, \theta, sig_{sk_{TI}}\ (TID, n_b, n_{bthr}, n_{thr}, TAG, \theta)\}$, the hash value $h_{K-1}$ of the header of block $K-1$, $TX_v, v = 0, \ldots, n0$, $TX_{ct_q}$, $q = 0, \ldots, n1$, $TX_{cc_p}$, $p = 0, \ldots, n2$, the ECC public key and corresponding certificate of a miner (i.e., $pk$ and $CE_{mr}$), the timestamp $t$
Output: block $K$
if the miner has created $n_{pk}(n_{pk} < n_{bthr} < n_b)$ blocks within the latest $n_b$ blocks of blockchain then
    Generate a transaction $TX_{mr} = \{pk, AM\}$;
    Obtain $mt$, i.e., the root of Merkle tree generated with $TX_v, v = 0, \ldots, n0$, $TX_{ct_q}$, $q = 0, \ldots, n1$ and $TX_{cc_p}$, $p = 0, \ldots, n2$, where $n0 \geq n_{thr}$ ;
    Compute the hash value $H_{pk} = h(TID \| h_{K-1} \| mt \| pk \| CE_{mr} \| t)$;
    if $H_{pk} \leq TAG * w(AM)$ then
        Insert $T, h_{K-1}, TX_{mr}, TX_v, v = 0, \cdots, n0$, $TX_{ct_q}$, $q = 0$, ..., $n1$, $TX_{cc_p}$, $p = 0$, ..., $n2$, $mt, pk$, $CE_{mr}$ and $t$ into block $K$;
    end
end
return block $K$;
```

Algorithm 1: Creation of block K

$n_{pk_{c0}} = n_{pk_{c1}}$ and $n0_{c0} \neq n0_{c1}$, the block including more receipts is the winner. When $t_{c0} = t_{c1}$, $n_{pk_{c0}} = n_{pk_{c1}}$ and $n0_{c0} = n0_{c1}$, the block whose relative hash value H_{pk} is smaller is regarded as the winner.

The valid blocks received by each miner might be different due to network latency. Thus, we show how to mitigate the inconsistency of block reception in order to avoid forking. If a miner receives the first valid block, it stops mining and sets the time window θ for receiving valid blocks from other miners. When a miner succeeds in generating a valid block but has not received valid blocks from other miners, it publishes its created block, waits θ for other valid blocks, and considers its block as the first received valid block. Because creating a valid block successfully before receiving other valid blocks can be considered as receiving the block ahead of the other blocks. θ can be adjusted and published by N_{TI} due to the network variation. The miner refuses to receive future blocks once θ expires. Finally, the miner performs block winner selection on all received valid blocks for determining the winner.

Incentive Mechanism

B4SDC should not only provide incentives for security data collection in route discovery and security data item collection, but also encourage the issuer to periodically publish a task and miners to insert receipts and deposit messages into the blockchain.

In order to resist the collusion attacks of collector and intermediate nodes (i.e., intermediate nodes do not report their receipts deliberately,) N_v's public key and TX_v's Q_{i_v} recorded in blockchain can be used to construct a virtual tree [3]. Except the tree root, each non-leaf and leaf node obtains α and β ($\alpha > \beta$) tokens from pk_0, respectively. As for each isolated node outside of the tree, the public address pa gets $\alpha - \beta$ tokens, where pa can be recorded in the genesis block. P_{i_v}, DT_{i_v} and SDI_{i_v} can help constructing continuous virtual route paths originating from the first forwarding node in real routes. In the path, each node obtains α tokens from pk_0, but the last one only obtains β tokens. Except all the nodes in the path, if there are remaining l nodes in the real route that have reported involved receipts, pa obtains $l * (\alpha - \beta)$ tokens from pk_0 in order to avoid the collusion of intermediate nodes and collector [3]. In Fig. 2, N_1, N_d and N_v obtain 2α, α and β tokens respectively. In Fig. 3, N_1, N_d and N_v also obtain 2α, α and β tokens respectively.

TABLE I
TRANSACTION CONFIRMATION TIME AND THROUGHPUT

n_b	0	1	2	3
TC(s)	36.9	37.2	37.5	38.1
TH(tx/s)	27.1	26.88	26.67	26.25

TC: transaction confirmation time; TH: throughput.

When the first receipt with UID or DI is inserted in one block of blockchain, miners will not collect and insert receipts with the same UID or DI after lasting n_r blocks from the block, thus redeeming the unexpended tokens of collector. pa might have some tokens, which can be distributed to nodes whose receipts are inserted into blockchain in a specific distribution. If a new TX_v, TX_{ct} or TX_{cc} is inserted into blockchain successfully, N_{TI} and the involved miners can obtain fixed tokens.

IV. Analysis and Performance Evaluation

We analyze B4SDC in terms of incentive and security, and evaluate the performance of B4SDC by simulations.

A. Analysis

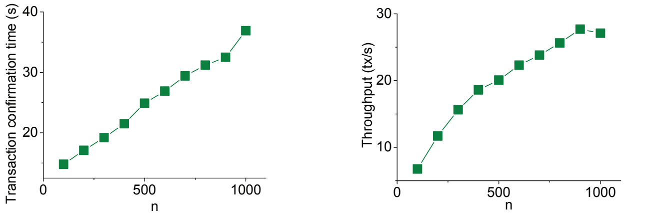
B4SDC not only incents security data collection, but also encourages N_{TI} to publish tasks and miners to insert more receipts or deposit messages into blockchain. In order to resist spoofing attacks, each collection participant should sign its sent message based on ECDSA. Receipts recorded in blockchain help constructing a virtual tree and continuous virtual route paths. In order to resist collusion attacks, each leaf node or the last node of a continuous virtual route path only obtains β tokens, and for each isolated node outside of the tree and paths, the collector transfers $\alpha - \beta$ tokens to pa [3]. In order to resist excessive forwarding, the collector constrains the scale of RREQ forwarding by adding TTL and k into this RREQ. Each forwarder follows the principle given in the RREQ, otherwise it obtains no rewards. Each node shares its receipts to miners, and miners compete to insert them into the next block for rewards. As long as a node participates in receipt generation and shares its receipts, it can obtain rewards with a high probability, thus ensuring the fairness. The liveness, safety, fault tolerance and decentralization of B4SDC can refer to the literature [13].

B. Performance Evaluation

In this part, we evaluated B4SDC through simulations. With respect to experimental environments, we simulated B4SDC using NS3 and C++ in a laptop that runs ubuntu 18.04 with Intel Core i5-6300HQ CPU @2.3Ghz and 12GB memory. We simulated a node as the certificate and task issuer, 1000 nodes for security data collection, and 4 miners. In terms of network settings, we simulated a real network by introducing the network delay between two miners that follows the exponential distribution with the rate parameter $\lambda = 12.6$s [14]. With respect to receipt generation, we adopted the DSR route discovery and used discovered routes to transmit the request of collector and the security data of collection nodes.

Transaction Confirmation Time: This time represents the average time that B4SDC takes to insert a receipt or deposit message into the blockchain. As shown in Fig. 4(a), we observe the transaction confirmation time when the task frequency is set to 4 tasks/min, $n_b = 0$, and we change the number $n = n0 + n1 + n2$ of receipts and deposit messages that should be inserted into the blockchain. The time increases linearly with n increasing. The bigger n suggests a miner collects and verifies more receipts and deposit messages, which takes more time. When we set the task frequency to 4 tasks/min and $n_{bthr} = 1$, Table I shows the transaction confirmation time by changing n_b. The time grows with the increase of n_b, since fewer miners participate in mining.

Throughput: Throughput is the average number of receipts and deposit messages that are successfully recorded in the blockchain per second. When the task frequency is set to 4 tasks/min and $n_b = 0$, we observe the throughput by changing n. As shown in Fig. 4(b), the throughput increases linearly when n increases. When $n_{bthr} = 1$ and the task frequency is 4 tasks/min, we observe the throughput with different n_b, as shown in Table I. When n_b increases, the throughput decreases due to the longer transaction confirmation time.



(a) Transaction confirmation time with 4 tasks/min, $n_b = 0$ and different n

(b) Throughput with 4 tasks/min, $n_b = 0$ and different n

Fig. 4. Experimental results

Comparison: In Table II, we compared the performance of B4SDC and mainstream blockchain systems. In the simulation, if the task frequency is 4 tasks/min and $n_b = 0$, the transaction confirmation time and throughput are $TC = 36.9$s and $TH = 27.1$tx/s, respectively. They are rational since a miner consumes almost no computing resources for solving a puzzle. Suppose there exist M miners. In PoW, BFT, PoS, PoUW and trees and DAGs based blockchian systems and B4SDC, a message should be shared to all miners for making them insert it into the blockchain, thus the communication complexity is $O(M)$. In PoW, PoUW, PoS, and trees and DAGs based blockchain systems and B4SDC, the next block should be sent to all the miners. Therefore, their communication complexity is $O(M)$. BFT requires information exchange between any two miners, thus its communication complexity is $O(M^2)$. The scalability is the ability of system to achieve a high throughput when many miners participate in. Only B4SDC supports the high scalability.

V. Conclusion

In this paper, we proposed B4SDC, a blockchain system for security data collection in MANETs. It not only incents security data collection in route discovery, but also motivates the collection at collection nodes. Analysis and simulation based experiments show that B4SDC can resist spoofing attacks, collusion attacks, and excessive forwarding, guarantee fairness as much as possible, and also solve the main problems of current blockchain technologies, namely forking, low efficiency, and centralization.

TABLE II
Performance Comparison of B4SDC with Mainstream Blockchain Systems

	PoW	BFT	PoS	PoUW	Trees and DAGs	
Criteria	Bitcoin	Algorand [6]	Ethereum	Permacoin [15]	GHOST [7]	B4SDC
TC(s)	3600	40	72	3600	-	36.9
TH(tx/s)	7	875	30	-	15.5	27.1
CPB	High	Low	Low	High	High	Low
CMB	$O(M)$	$O(M^2)$	$O(M)$	$O(M)$	$O(M)$	$O(M)$
SC	Low	Low	Low	Low	Low	High

CPB: computational burden; CMB: communication burden; SC: scalability; -: not given.

Acknowledgment

The work is supported in part by the National Natural Science Foundation of China under Grants 61672410 and 61802293, the Academy of Finland under Grants 308087 and 314203, the Key Lab of Information Network Security, Ministry of Public Security under grant No. C18614, the open grant of the Tactical Data Link Lab of the 20th Research Institute of China Electronics Technology Group Corporation, P.R. China under grant CLDL-20182119, the Shaanxi Innovation Team project under grant 2018TD-007, and the 111 project under grant B16037.

References

[1] G. Liu, Z. Yan, and W. Pedrycz, “Data collection for attack detection and security measurement in Mobile Ad Hoc Networks: A survey,” J. NETW. COMPUT. APPL., 2018, 105: 105-122.

[2] Y. Liu, Y. Wang, X. Wang, et al., “Privacy-preserving raw data collection without a trusted authority for IoT,” COMPUT. NETW., 2019, 148: 340-348.

[3] S. Zhong, J. Chen, and Y.R. Yang, “Sprite: A simple, cheat-proof, credit-based system for Mobile Ad-Hoc Networks,” In Proc. of IEEE INFOCOM 2003, 1987-1997.

[4] C. Li, B. Yu, and K. Sycara, “An incentive mechanism for message relaying in unstructured Peer-to-Peer systems,” ELECTRON. COMMER. R. A., 2009, 8(6): 582-592.

[5] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.

[6] Y. Gilad, R. Hemo, S. Micali, et al., “Algorand: Scaling byzantine agreements for cryptocurrencies,” In Proc. of ACM SOSP 2017, 51-68.

[7] Y. Sompolinsky and A. Zohar, “Accelerating bitcoins transaction processing. fast money grows on trees, not chains,” IACR Cryptology ePrint Archive, 2013.

[8] S. Qazi, R. Raad, Y. Mu, et al., “Securing DSR against wormhole attacks in Multirate Ad Hoc Networks,” J. NETW. COMPUT. APPL., 2013, 36(2): 582-592.

[9] J. Chen, S. Yao, Q. Yuan, et al., “CertChain: Public and efficient certificate audit based on blockchain for TLS connections,” In Proc. of IEEE INFOCOM 2018, 2060-2068.

[10] Y.C. Hu, A. Perrig, and D.B. Johnson, “Wormhole attacks in wireless networks,” IEEE J. SEL. AREA. COMM., 2006, 24(2): 370-380.

[11] K. Fan, S. Wang, Y. Ren, et al., “Blockchain-based secure time protection scheme in IoT,” IEEE INTERNET THINGS, 2018, https://doi.org/10.1109/JIOT.2018.2874222

[12] G. Liu, Z. Yan, and Y. Fu, “A micropayment-based incentive mechanism for security-related data collection in route discovery of DSR protocol,” J. CYBE. SECU., 2019, 4(1): 1-13.

[13] W. Feng and Z. Yan, “MCS-Chain: Decentralized and trustworthy mobile crowdsourcing based on blockchain,” FUTURE GENER. COMP. SY., 2019, 95: 649-666.

[14] C. Decker and R. Wattenhofer, “Information propagation in the bitcoin network,” In Proc. of IEEE P2P 2013, 1-10.

[15] A. Miller, A. Juels, E. Shi, et al., “Permacoin: Repurposing bitcoin work for data preservation,” In Proc. of IEEE S&P 2014, 475-490.