MARS: Monetized Ad-hoc Routing System (A Position Paper)

Bernardo David* Rafael Dowsley† Mario Larangeira*

ABSTRACT

A Mobile Ad-Hoc Network (MANET) automatically reorganizes itself, allowing moving nodes to join or leave the network at any point in time without disrupting the communication. An essential element of such systems is a routing protocol able to quickly restructure existing routes when nodes leave and provide routes to new nodes. In most MANET routing protocols, nodes are assumed to be altruistic, that is, forward incoming packets to the next node in the route. However, as pointed out in a number of previous works, nodes are often selfish in real world scenarios, refusing to forward incoming packets from their peers but still using the MANET to route their own packets. In this work, we propose MARS, a blockchain-based reputation system that acts as an overlay on top of existing MANET routing protocols (*e.g.* AODV and OLSR). The main goal of MARS is to keep a publicly available (and verifiable) record of node behavior that can be used to both select good routes and incentivize nodes to actively participate in routing to earn rewards. As a building block, we propose a compact "proof-of-routing" that allows a node to prove that it has participated in the routing of a batch of packets. Upon presenting such a proof, the node earns a reputation point, which is stored as an asset in the blockchain, and as such can be traded for enhanced network services among MANET nodes or for other assets (*e.g.* cryptocurrencies) with third parties.

1. INTRODUCTION

A mobile ad-hoc network (MANET) allows mobile devices to communicate without any pre-established infrastructure or centralized management. In a MANET, nodes cooperate among themselves to route messages, dynamically adjusting routes as they join, leave and physically move. Such flexibility makes MANETs attractive for applications such as campus networks, disaster relief, providing internet access networks in areas without infrastructure and *Internet-of-Things* (IoT) applications. However, a consequence of node mobility and lack of central infrastructure is an unknown and constantly changing network topology, which makes it unfeasible to deploy traditional routing protocols [7].

Providing efficient and reliable routing for MANETs is a challenging task for which a number of protocols has been developed [9]. These protocols can be classified into two main categories [10]: reactive routing protocols, where nodes discover routes only when needed, and proactive routing protocols, where nodes perform a constant route discovery process by periodically exchanging topology information. The Ad-Hoc On Demand Distance Vector (AODV) protocol [15], a reactive routing protocol, and the Optimized Link State Routing Protocol (OLSR) [6], a proactive routing protocol, are well known examples of MANET routing protocols and will be used as examples in this work.

MANET routing protocols are usually not designed with security in mind and are indeed subject to several threats and attacks [11, 9, 10]. Nodes that intentionally misbehave in a MANET routing protocol can be classified into two main categories: *malicious* nodes or *selfish* nodes [13]. Malicious nodes aim at actively disrupting routing operations by subverting routing operations or overloading the network. On the other hand, selfish nodes do not purposefully disrupt network operations but refuse to route incoming messages while using other nodes' resources to route their own messages. Detecting and mitigating selfish behavior has proven to be a hard problem, since selfish nodes do not actively deviate from the protocol.

A number of heuristics for detecting and isolating selfish nodes have been proposed [5, 13, 2, 1, 17]. Most of them are *reputation*-based solutions, basically providing ways for nodes to measure how much their peers are contributing to routing and keep local records of each other's reliability (*i.e.* reputation). Given this data, nodes can choose which peers to cooperate with. Notice that, in addition to observing the behavior of

*Tokyo Institute of Technology and IOHK. Emails: {bernardo,mario}@c.titech.ac.jp. This work was supported by the Input Output Cryptocurrency Collaborative Research Chair, which has received funding from IOHK.

†Aarhus University and IOHK. Email: rafael@cs.au.dk. This project has received funding from the European research Council (ERC) under the European Unions's Horizon 2020 research and innovation programme (grant agreement No 669255).

peers in their vicinity, nodes also rely on external advice for building their reputation records (specially for peers that cannot be reached directly). However, in current reputation-based schemes, each node keeps its own reputation records locally, allowing dishonest nodes to falsely accuse their peers of misbehavior. These issues affect the accuracy and effectiveness of current reputation systems, which employ complicated heuristics to mitigate false claims of misbehavior and build a cohesive view of reputation among honest nodes. The solutions presented in [1, 17] also employ financial incentives, proposing a "central bank" entity that financially rewards nodes who participate in routing.

1.1 Our Contributions

In this work, we introduce MARS, a system that uses cryptographic tools to build a decentralized and *publicly verifiable* record of nodes' reputations in MANET routing protocols that can be accessed and verified by any third party (including new nodes that join the network). Moreover, MARS allows nodes to trade their reputation points for other assets, such as (improved) network services and cryptocurrencies. MARS works as an overlay extension to any MANET routing protocol. It stores reputation information in a blockchain-based public ledger [14, 12], which allows nodes to publicly share data with the added guarantee that data becomes immutable once written [8]. In order to provide publicly verifiable reputation information, we introduce the concept of *Proof-of-Routing*, which allows any entity to verify whether a node has participated in the routing of a given message (or batch of messages) and prevents dishonest nodes from claiming extra reputation points or keeping other nodes from claiming their points. MARS' achieves the following main characteristics: **(1) Decentralization and Public verifiability:** No central authority is needed for awarding reputation points and any third party can verify the decentralized reputation record's consistency; **(2) Resistance against dishonest nodes:** Dishonest nodes cannot abuse MARS to artificially increase their own reputation or decrease a node's reputation; **(3) Trading reputation:** Reputation in MARS is treated as a digital *asset* that can be traded for other assets, services or cryptocurrencies.

Instead of relying on expensive Proof-of-Work based blockchain protocols (*e.g.* Bitcoin [14, 8]) MARS relies on lightweight Proof-of-Stake based protocols [12]. MARS works by awarding a number of reputation points to a node every time a proof-of-routing showing that this node participated in routing is posted to the public ledger. A proof-of-routing cannot be forged by a node who didn't participate in routing nor can a dishonest node disavow another node's proof-of-routing. Hence, well-behaved nodes are guaranteed to receive their reputation points while dishonest nodes cannot artificially receive more points. We construct a proof-of-routing scheme based on *composite signatures* [16]. As in a cryptographic currency (*e.g.* Bitcoin), any entity can determine how many reputation points a node has received by inspecting the blockchain and counting how many proofs-of-routing including that node have been posted. Moreover, using standard cryptocurrency techniques, a node can *transfer* reputation points to another party in exchange for cryptocurrency coins, services (*e.g.* increased network speed) or other assets. This flexibility of dealing in reputation points, allows the system to offer financial incentives for altruistic behavior. Moreover, it can also be used as part of a billing system where nodes can trade reputation points for discounts in using network services (*e.g.* an Internet gateway) if they are actively helping run the network.

1.2 Related Works

Out of the many works proposing heuristics for mitigating selfish behavior in MANETs, e.g. [5, 13, 2, 1, 17], Ad hoc-VCG [1] and Sprite [17] stand out due to their use of financial incentives. Sprite [17] introduced the idea of incentivizing nodes to actively participate in routing messages through financial rewards accrued for each message that is routed. Moreover, a game theoretic analysis is presented, showing that their reward strategy results in rational players collaborating in routing, considering the nodes' cost for participating in the protocol versus the rewards. An important issue in Sprite is that nodes are assumed to have access to a "central bank" entity, which provides the financial rewards upon receiving a cryptographic proof that a given node participated in routing a message. This proof consists solely of a signature on the message and the identity of the nodes in the route by the source node, which a malicious node (who deviates from the protocol) can present to the central bank in order to receive a reward even it decides to drop the message. Ad hoc-VCG [1] improved on the game-theoretic analysis of Sprite and proposed that nodes pay rewards directly to each intermediate node in the route to their messages. However, Ad hoc-VCG does not propose an efficient way to implement such payment mechanism nor guarantees that nodes who deviate from the protocol cannot dishonestly obtain rewards.

Biryukov and Pustogarov [3] also proposed a blockchain based reward mechanism for nodes in the semi-decentralized Tor anonymous routing network. In this scheme, Tor relays (powerful nodes that aggregate large volumes of traffic) are compensated by users through block mining. Basically, the nodes connected to a given Tor relay act as members of a mining pool, trying to solve proof-of-work puzzles and sending their (partial) solutions to the Tor relay. In this scheme, the Tor relay is compensated by either using this information to mine blocks itself or

participating itself of a large mining pool. In contrast to MANETs, this scenarios assumes a pre-established and centralized network infrastructure, *i.e.* the Tor relays are known beforehand and nodes are assigned to a specific Tor relay for a long period of time, instead of dynamically changing routes as in a MANET.

2. PRELIMINARIES

We denote concatenating a string x with a string y by $x \mid y$.

Composite Signatures: A central cryptographic building block used in our solution is a *composite signature scheme*, as defined by Saxena et al. [16]. This primitive was derived from aggregate signature scheme [4]. The main similarity between these primitives is that in both cases multiple signatures can be combined into one single and short (aggregated) signature. The feature introduced by [16] is that the aggregation process is *one-way*, namely, given the aggregated signature, it is very hard to compute the individual signatures (or the signatures on any proper subset). We adopt the definition of Composite Signature Schemes given by Saxena et al. [16]. Let a message-descriptor ℓ consist of pairs of message/verification key, i.e., $\ell = \{(m_1, vk_1), \ldots, (m_i, vk_i)\}$. A composite signature scheme $\mathsf{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Verify}, \mathsf{SIG.Compose})$ works as follows:

- $\mathsf{SIG.Gen}$ gets as input the security parameter λ and output a pair of signing sk and verification vk keys.
- $\mathsf{SIG.Sign}$ takes as input a pair of keys (sk, vk) and a message m and outputs a signature σ on the message-descriptor $\ell = \{(m, vk)\}$.
- $\mathsf{SIG.Verify}$ takes as input a message-descriptor $\ell = \{(m_1, vk_1), \ldots, (m_i, vk_i)\}$ and a signature σ and outputs a decision bit about the validity of the signature. If a single message is signed under multiple signing keys, the message-descriptor is denoted as $\ell = \{m, vk_1, \ldots, vk_i\}$.
- $\mathsf{SIG.Compose}$ takes as input two pairs of message-descriptor/signature, (ℓ_1, σ_1) and (ℓ_2, σ_2). If $\ell_1 \cap \ell_2 \neq \emptyset$ or any of the message-descriptor/signature pairs is invalid according to $\mathsf{SIG.Verify}$, it outputs $\perp$; otherwise, it outputs a composite signature σ on the message-descriptor $\ell = \ell_1 \cup \ell_2$.

The correctness requirement for such schemes is the straightforward one. The compactness requirement is that the composite signature has the same size as a single signature. The key security property is that given the set of valid composite signatures that are available to an adversary, he should be able to compute valid composite signatures only on unions of their message-descriptors. An interesting consequence of this property is that the adversary cannot remove an honest party's signature from a composite signature. For more details on the formal security definition for composite signatures, we refer the reader to [16], where it was also proven that the aggregate signature scheme of Boneh et al. [4] can be transformed into a secure composite signature scheme by appending the verification-key and a random string to the message.

Blockchain ledgers and their properties: Blockchain based systems have been employed in several different scenarios beyond its original financial goal. In a nutshell, what these applications have in common is that they rely on a distributed ledger kept by a network of users, *i.e.* the blockchain. Arguably the most famous blockchain protocol is Bitcoin [14], which achieves global reach. Several other cryptocurrencies following a similar blockchain based design exist, showcasing the felxibility of such systems. Naturally, the security of a blockchain based distributed ledger depends on computational assumptions and on the fact that honest parties control the majority of a certain resource (*e.g.* hash power or stake). Specifically, in the context of Proof-of-Work based blockchains, Garay et al. [8] have shown that given a random oracle, a digital signature scheme and that hones tparties control a majority of the hash power, the following properties can be achieved:

- **Common-Prefix:** When pruning a large enough amount of the newest blocks, the local views of the remaining chain by honest parties have exactly the same blocks.
- **Chain-Quality:** Honest parties are guaranteed to generate at least a constant fraction of blocks in a long enough sequence of blocks.
- **Chain-Growth:** The blockchain is guaranteed to grow at a minimal rate of blocks per elapsed time.

In the context of Proof-of-Stake, the Ouroboros [12] protocol is proven to achieve the same security guarantees given that honest parties control a majority of the stake. MARS is implemented over a Proof-of-Stake blockchain since its much less resource intensive than Proof-of-Work based ones. More interestingly, in [8] it is proven that a blockchain that achieves these properties implements a *distributed ledger*. Such a system works as a robust distributed database that keeps a set of *records*, which in the case of a cryptocurrency are transactions. Informally, a distributed ledger has two properties:

- **Liveness:** Given enough time, a record created by an honest party will be inserted into the ledger;
- **Persistence:** Honest parties agree on records that have been registered in the ledger for long enough with overwhelming probability.

MANET Routing Protocols: MANET routing protocols can be classified into two main categories [10]:

reactive routing protocols, where nodes discover routes only when needed, and proactive routing protocols, where nodes perform a constant route discovery process by periodically exchanging topology information. Several MANET routing protocols have been developed and are discussed in surveys such as [9, 10]. We will exemplify our solution in terms of the Ad-Hoc On Demand Distance Vector (AODV) protocol [15] and the Optimized Link State Routing Protocol (OLSR) [6].

3. THREAT MODEL

We consider that the nodes can be selfish in relation to the execution of the routing protocol. This means that they do not disrupt the network operations on purpose. However, each node potentially tries to avoid using its own resources to route messages from other nodes while using the resources from other nodes' to route its own messages. This is the case of traditional selfish behavior where the nodes have no wish to disrupt communications but do wish to save their own resources, as defined in [13].

In addition, we do consider that the nodes can try to act dishonestly in the execution of the reputation system in order obtain advantages without being flagged as selfish. The nodes are modeled as probabilistic polynomial-time (PPT) Turing machines. With these added adversarial powers, we augment the usual model of selfish behavior [13] to capture real world situations where dishonest nodes still do not wish to disrupt communications but will adopt adversarial strategies to subvert systems that detect and isolate/punish selfish behavior [5, 13, 2, 1]. We call such adversarial nodes *dishonest*, in contrast to fully malicious or simply selfish nodes, as defined in [13].

We consider the case in which there is no collusion among the different dishonest nodes, meaning that dishonest nodes act individually to gain advantages but do not perform coordinated attacks. We argue that this restriction on collusion seems natural in highly mobile scenarios where nodes join and leave the network (or move around) very frequently, without having sufficient time to discover other dishonest nodes and coordinate. We leave an analysis with collusions as a future work.

We wish to build a reputation system that awards reputation points to nodes that truly cooperate in the routing protocol with security against dishonest nodes as defined above. The first property we wish to obtain in a secure reputation system is that no dishonest node can gain reputation points without presenting a valid proof-of-routing in which he was part of the route. The second property is that no dishonest node can decrease another node's reputation (*e.g* by presenting false data).

4. BUILDING MARS

The main goal of MARS is to keep a publicly verifiable record of reputation information for MANET nodes while resisting attacks by dishonest nodes who aim at maliciously increasing their own reputation or decreasing other nodes' reputations. Reputation points are stored in a lightweight Proof-of-Stake based blockchain in such a way that they can be used in transactions between users of the ledger as in a cryptocurrency. To that end, we introduce a proof-of-routing scheme that allows for nodes (and third parties) to verify whether a given node has participated in the routing of a given batch of messages. Each node receives a number of reputation points proportional to the amount of valid proofs-of-routing posted on the blockchain. MARS is run in conjunction with an arbitrary MANET routing protocol, requiring that each node participates in generating a proof-of-routing after routing a batch of messages. Once a proof-of-routing showing participation is posted in the public ledger, the nodes who participated in the routing are awarded reputation points. MARS runs on top of a Proof-of-Stake based blockchain [12] using a composite signature scheme $\mathsf{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Verify}, \mathsf{SIG.Compose})$ and proceeds as follows:

- **Node Registration (Avoiding Sybil Attacks):** Upon joining the MANET, a new node N_i proceeds as follows to register its verification key for SIG with a solution to a Proof-of-Work puzzle (in order to avoid Sybil attacks): (1) run $\mathsf{SIG.Gen}(1^\lambda)$ to generate (sk_i, vk_i), (2) find nonce_i such that $H(vk_i|\mathsf{nonce}_i) < d$, where $H(\cdot)$ is a cryptographic hash function and d is a difficulty paramter, (3) post (vk_i, nonce_i) to the blockchain. A node is only allowed to join the network if itregisters a pair (vk', nonce') such that $H(vk'|\mathsf{nonce}') < d$.
- **Earning Reputation Points with Proof-of-Routing:** A node earns reputation points by providing a Proof-of-Routing, showing that it has actively particiapted in routing. The proof-of-routing is basically a composite signature by each node that participates in the route of a batch of messages that is being routed, along with a signature by the source node on the hash of the batch, its own address and the destination node's address. In order to save space and achieve the desired security guarantees, each node in the route composes its signature with the initial signature generated by the source node, forming a small composite signature that contains the signature of each node involved in the proof-of-routing. First, each node routes individual messages until a large enough batch is transmitted. Next, the source node generates a signature on the batch of messages, which is forwarded to each node on the route, who in turn generate the composite signature containing the previous signatures along with its own. Additionally, each node checks the validity of the composite signature it has received from the previous node, aborting if it is not valid. By the security properties of

composite signatures and since there is no collusion among the nodes, no node can remove the signatures of the previous nodes from the composite signature it receives, and thus cannot decrease the number of reputation points received by honest nodes. For a given route from node N_0 to node N_d, we denote the source node as N_0, the destination node as N_d and the intermediary nodes in the route as $N_1, \ldots, N_{d-1}$. More specifically a proof of routing is generated and verified as follows:

- **Source node:** N_0 signs a hash of its batch of messages $h = H(m_1|\cdots|m_\ell)$ concatenated with its address N_0 and the address of the destination node N_d, obtaining a signature $\sigma_{0,h} = \mathsf{SIG.Sign}(sk_0, vk_0, N_0 \mid N_d \mid h)$. N_0 sends both $N_0 \mid N_d \mid h$ and $\sigma_{0,h}$ to N_d.
- **Intermediary nodes:** For $i = 2, \ldots, d-1$, each node N_i verifies previous nodes' signature by computing $\mathsf{SIG.Verify}(N_0 \mid N_d \mid h, vk_0, \ldots, vk_{i-1}, \sigma_{i-1,h})$, computes a signature $\sigma'_{i,h} = \mathsf{SIG.Sign}(sk_i, vk_i, N_0 \mid N_d \mid h)$, and composes it with the previous signature $\sigma_{i-1,h}$, obtaining a composite signature $\sigma_{i,h} = \mathsf{SIG.Compose}(N_0 \mid N_d \mid h, vk_{i-1}, \sigma_{i-1,h}, vk_i, \sigma'_{i,h})$. N_i forwards $N_0 \mid N_d \mid h$ along with its fresh composite signature $\sigma_{i,h}$ to N_{i+1}.
- **Destination node:** N_d performs the same actions as the intermediary nodes. Additionally it posts its final composite signature $\sigma_{d,h}$ along with $N_0 \mid N_d \mid h$ to the blockchain. N_d is also required to post a list of the nodes that participated in routing the batch of messages, which can be accomplished in different ways as discussed later in this section.
- **Verifier:** Any party V who wants to verify a proof of routing uses the list of nodes that participated in the routing of the batch of messages along with $N_0 \mid N_d \mid h$ to verify the validity of the composite signature $\sigma_{d,h}$. All nodes whose signatures are included in $\sigma_{d,h}$ earn an amount of reputation points that can be adjusted proportionally to the size of the message batch concerned in the proof.

- **Transactions with Reputation Points:** A node earns reputation points when a valid proof-of-routing containing its verification key is posted to the blockchain. Since reputation points are connected to a node's verification key, transactions with reputation points can be carried out using the same transaction scheme as Bitcoin. In order to transfer a reputation point to another entity, a node uses the signing key corresponding to his verification key to sign a Transfer Transaction specifying both the verification key of the entity who will receive the points and the proof-of-routing that awarded those points. This transaction is considered valid if the proof-of-routing specified as the source of reputation points is valid and contains that user's verification key; and the transaction is signed under the corresponding signing key.

Security Analysis and Collusion: First we argue that a dishonest node cannot prevent a honest node (who participated in routing a message) from receiving its reputation points. It is clear that dishonest nodes cannot remove an honest node's signature from the intermediate composite signature they receive during the routing process or from the final composite signature posted in the blockchain, which follows from the security definitions of composite signatures (see Section 2) and from the fact that there are no collusion among nodes. Notice that this also means that a dishonest node cannot decrease the number of reputation points another node has received in the future, since the proofs-of-routing awarding points to a node become immutable once posted to the blockchain (by the Persistence property of public ledgers presented in Section 2). Next, notice that dishonest nodes cannot gain reputation points without participating in routing messages. In our threat model we assume that dishonest nodes cannot collude and perform coordinated attacks, having to act alone in trying to subvert the reputation system. In this scenario, all nodes are forced to actively participate in routing and generating a correct proof-of-routing. If a dishonest node fails to route or provides an invalid partial proof-of-routing, the transmission is aborted. Hence a node has to follow the protocol so that the message reaches the destination in order to receive reputation points. In case of collusion, a group of dishonest nodes who coordinate an attack would be able to simply share their secret-keys and sign all of the messages routed by each of them in the names of each other, effectively adding all of them to the proof-of-routing. However, this strategy requires that the secret-keys of all of these nodes are revealed to each other, allowing these dishonest nodes to steal each others' reputation points.

Deploying MARS with Reactive and Proactive Routing Protocols: The proof of routing mechanism requires any parties who want to verify a proof to know the nodes that participated in routing the message (or batch of messages) associated to that proof. Hence, this information must be posted along with the proof itself in the blockchain. When using MARS on top of reactive routing protocols (*e.g.* AODV), where routes might change for every message that is transmitted between a source node and a destination node, it might be necessary to post data describing all nodes that participated in routing to the blockchain, which can cause a high storage overhead. However, if MARS is deployed on top a proactive routing protocol (*e.g.* OLSR), a routing table is known for each node. In this case, the protocol can be modified to require each node to first post its routing table to the public ledger, later including

a reference to specific routes in the table in each message that it sends. Intermediary nodes will then only route a message if they are part of the route referenced in the message. When the routing table changes, the node can simply post an update to the public ledger specifying only the new route and the old route that it supersedes. Such a modification amortizes the storage overhead in the public ledger, since individual route information does not have to be posted for every single proof of routing.

5. REFERENCES

[1] Anderegg, L., Eidenbenz, S.: Ad hoc-vcg: A truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. In: Proceedings of the 9th Annual International Conference on Mobile Computing and Networking. pp. 245–259. MobiCom '03, ACM, New York, NY, USA (2003)

[2] Balakrishnan, K., Deng, J., Varshney, V.K.: Twoack: preventing selfishness in mobile ad hoc networks. In: IEEE Wireless Communications and Networking Conference, 2005. vol. 4, pp. 2137–2142 Vol. 4 (March 2005)

[3] Biryukov, A., Pustogarov, I.: Proof-of-work as anonymous micropayment: Rewarding a tor relay. In: Böhme, R., Okamoto, T. (eds.) FC 2015. LNCS, vol. 8975, pp. 445–455. Springer, Heidelberg (Jan 2015)

[4] Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and verifiably encrypted signatures from bilinear maps. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (May 2003)

[5] Buchegger, S., Le Boudec, J.Y.: Performance analysis of the confidant protocol. In: Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing. pp. 226–236. MobiHoc '02, ACM, New York, NY, USA (2002)

[6] Clausen, T., Jacquet, P.: Optimized link state routing protocol (olsr) (2003)

[7] Corson, S., Macker, J.: Mobile ad hoc networking (manet): Routing protocol performance issues and evaluation considerations (1999)

[8] Garay, J.A., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol: Analysis and applications. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 281–310. Springer, Heidelberg (Apr 2015)

[9] Hu, Y.C., Perrig, A.: A survey of secure wireless ad hoc routing. IEEE Security and Privacy 2(3), 28–39 (May 2004)

[10] Kannhavong, B., Nakayama, H., Nemoto, Y., Kato, N., Jamalipour, A.: A survey of routing attacks in mobile ad hoc networks. IEEE Wireless Communications 14(5), 85–91 (October 2007)

[11] Karlof, C., Wagner, D.: Secure routing in wireless sensor networks: attacks and countermeasures. In: Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, 2003. pp. 113–127 (May 2003)

[12] Kiayias, A., Russell, A., David, B., Oliynykov, R.: Ouroboros: A provably secure proof-of-stake blockchain protocol. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10401, pp. 357–388. Springer (2017)

[13] Marti, S., Giuli, T.J., Lai, K., Baker, M.: Mitigating routing misbehavior in mobile ad hoc networks. In: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking. pp. 255–265. MobiCom '00, ACM, New York, NY, USA (2000)

[14] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008)

[15] Perkins, C., Belding-Royer, E., Das, S.: Ad hoc on-demand distance vector (aodv) routing (2003)

[16] Saxena, A., Misra, J., Dhar, A.: Increasing anonymity in bitcoin. In: Böhme, R., Brenner, M., Moore, T., Smith, M. (eds.) FC 2014 Workshops. LNCS, vol. 8438, pp. 122–139. Springer, Heidelberg (Mar 2014)

[17] Zhong, S., Chen, J., Yang, Y.R.: Sprite: a simple, cheat-proof, credit-based system for mobile ad-hoc networks. In: IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428). vol. 3, pp. 1987–1997 vol.3 (March 2003)