Improving the Efficiency of Anonymous Routing for MANETs[☆]

Jiefeng (Terence) Chen[a,b,*], Roksana Boreli[a,b], Vijay Sivaraman[b]

[a]*National ICT Australia (NICTA), Locked Bag 9013, Alexandria, NSW 1435, Australia*
[b]*School of Electrical Engineering and Telecommunications, University of New South Wales (UNSW), Sydney, NSW 2052, Australia*

Abstract

Anonymity of mobile devices and their data is an essential requirement to facilitate the deployment of future Mobile Ad hoc Networks. The currently proposed anonymous routing mechanisms for such networks inherently include a trade-off between a high level of anonymity, performance and scalability. In this paper[1], we propose an anonymous routing protocol that provides improved anonymity and security while achieving similar or better performance, as compared to existing proposals. Our proposal achieves anonymity using an efficient solution for invisible implicit addressing based on *keyed hash chain* and security via a novel application of one-to-many Diffie-Hellman mechanism, used to exchange keys for symmetric encryption. The final contribution includes a mechanism to facilitate selection of a trusted route by verifying connections between intermediate nodes. We demonstrate the benefits of our proposal in comparison with previous approaches using analysis and simulation.

Keywords: MANET, anonymous routing, onion routing, security, pseudonym

[☆]This research work has been funded by National ICT Australia (NICTA).
[*]Corresponding author. Tel.: +61 02 93762195. Fax: +61 02 93762024.
Email address: `terence.chen@nicta.com.au` (Jiefeng (Terence) Chen)
[1]Parts of this paper have been published in TrustCom 2010.

1. Introduction

The widespread popularity of mobile applications and content and the growing adoption of smartphones creates a need for increased network capacity, in addition to what may be provided by the evolution of standard mobile networks. The increasing availability of wireless technologies like Wi-Fi on smartphones, which have a capability for direct mobile-to-mobile connections, makes Mobile Ad hoc Networks (MANETs) a likely candidate to provide the required capacity increase. In MANETs, mobile nodes communicate without the aid of any pre-existing infrastructure, either directly or indirectly through peers. Due to the openness and cooperative nature of such networks, MANETs are vulnerable to a wide range of threats [16] to both the identity of users and to their data, therefore anonymity and security are essential in such networks.

Anonymity has been defined by Pfitzmann and Hansen [17] as "the state of being not identifiable within a set of subjects, the *anonymity set*". The size of this anonymity set is a quantifiable measurement of anonymity. In MANET data communication, anonymity relates to the the requirement that the identities of the source, destination and the route of a data message cannot be linked to any node within the network. An additional requirement which relates to the anonymity of data is *unlinkability* [17], defined as the notion of a third party (attacker) being unable to distinguish whether any two or more items of interest (in the case of MANETs, data packets) are related. When applied to routing, unlinkability will ensure that data packets from a single flow cannot be linked in order to trace the origin and the destination of this flow.

Mobile devices often have limited power and processing capabilities. As a consequence, the existing anonymous routing mechanisms which provide a high level of anonymity generally have poor scalability, i.e. the supported population of nodes, due to delays and overhead introduced by expensive cryptographic operations. To mitigate this, some approaches partially sacrifice anonymity to achieve better performance [1], [2], or disregard unlinkability [2].

In this paper we present a Trusted Anonymous Routing (TARo) protocol [18] that provides a high level of anonymity for network nodes and ensures unlinkability of data flows, while achieving a performance comparable to existing protocols. TARo is a distributed on-demand routing protocol which establishes multiple Virtual Circuits (VCs) between the sender and receiver.

Within the TARo proposal, we have the following contributions. We propose a solution for invisible implicit addressing [19] based on *keyed hash chain*. Shared keys between the sender and intermediate nodes are exchanged using the Diffie-Hellman mechanism [20] and these are later used to ensure unlinkability of data by per-hop data packet appearance alteration. Our second contribution is the proposed mechanism to facilitate selection of a trusted route by verifying connections between intermediate nodes through detecting untrusted connectivity announcements.

In the remainder of the paper, in Section 2 we first review the existing anonymous MANET routing algorithms and identify the anonymity and performance related issues. In Section 3 we describe the attack models, notations and cryptographic tools used in this work. Section 4 describes the proposed protocol, with performance evaluation presented in Section 5. We conclude and present future work in Section 6.

2. State of the Art

2.1. Vulnerabilities and Threats

MANETs suffer from all the vulnerabilities of wired networks, while the desirable features of wireless network, from network security point of view, expose the system to a wider range of threats. Open communication media enables connectivity with no infrastructure, however unauthorised network monitoring becomes much easier than in infrastructure based networks. Deployment of centralised Public Key Infrastructure (PKI) is impractical due to MANETs' distributed and infrastructure-less nature. Also, the resource constraints of mobile devices additionally limit their capability for using secure but computationally intensive cryptographic operations, e.g. public key encryption and decryption.

A large number of potential attacks exist against MANET routing. These attacks include identity spoofing, link spoofing, replay attack, man-in-the-middle attack, wormhole attack, black hole attacks, routing table overflow attack, Sybil attack, etc [21]. The purpose of these attacks is to interrupt routing decisions, and to take control of the communications in order to obtain sensitive information.

Generic network attacks against privacy can be achieved simply by monitoring network traffic. The packet headers reveal a lot of information, for example identities of the source, destination and forwarding nodes, hop distance between communicating nodes, physical location of devices, etc. We note that the use of long term identities will also enable tracing of node movement, thus compromising location privacy.

In the case that long term identities are not used, the goal of the attackers will be to attempt to reduce the anonymity set size of sender and receiver identities, ideally to identify them uniquely. More sophisticated traffic analysis techniques [22] listed below may be used to identify a route and hence to trace the source and destination of data packets.

- **Message coding analysis**: Messages that do not change their coding can be traced through the network by pattern matching. Attackers can be either internal and external, where internal attackers take part of the communication and possess necessary keys.
- **Message length analysis**: Examines the length of a message as it travels through the network.
- **Timing analysis**: Tries to observe the duration of a connection by correlating its establishment or release at the possible endpoints.
- **Profiling analysis**: Tracks user behaviour over long-term periods.

2.2. Existing Anonymous Routing Protocols

There have been a number of anonymous routing protocols proposed for MANET in the past years [1] - [15]. In summary, the the complete routing functions are generally performed in the following four phases in these protocols:

Anonymous neighbour authentication: Nodes establish trust relationships and broadcast/shared keys with one-hop neighbours in order to speed up the route discovery process.

Anonymous route discovery: A route is typically discovered using two messages: broadcast *route request* (RREQ) message and unicast/multicast *route reply* (RREP) message.

Anonymous data transmission: Delivers data (DATA) packets to destination without exposing the identity of nodes while ensuring unlinkability.

Route maintenance: Maintains the routes, relying on the assumption that link failures can be detected by observing lower layer parameters, or by network layer keep alive messages. Typically an error (ERROR) message is sent back to the source if a link breakage is detected.

All the reviewed protocols aim to achieve either node anonymity or unlinkability. Common mechanisms used to preserve node anonymity include onion routing/layered encryption ([3], [4] and [5]), pseudonyms ([4], [6] and [11]) and invisible implicit addressing ([5], [7]). Invisible implicit addressing preserves receiver anonymity by encrypting a message with receiver's public key or shared key, this encrypted message is called the *trapdoor*. Instead of sending the message directly to the receiver, it is broadcast to all nodes in the network, which consequently try to decrypt the trapdoor. However only the intended receiver will have the corresponding key to open it. Some protocols, like SDAR [5], MASK [1] and Discount-ANODR [2], partially violate the anonymity requirement as they use real identities of participating nodes in order to achieve improved performance. E.g. in [5] nodes use real identities but they are encrypted and known only to sender and receiver, which guarantees the anonymity of intermediate nodes to observers but not to the sender and receiver. In [1] and [2], real identity of the receiver is used in route discovery phase to avoid costly invisible implicit addressing.

To ensure unlinkability and prevent passive attackers from observing the data flow using traffic analysis introduced in Section 2.1, protocols utilise techniques like per-hop packet appearance alteration ([1], [4], [8], [10], [15]) to thwart message coding analysis, use of fixed packet size achieved by padding ([7], [8], [12]) to protect against message length analysis, random delay [1] for timing attack, random forwarding [13],[8], dummy packet injection [1] and traffic mixing [14] to thwart the profiling analysis.

Delay and overhead are directly related to the scale of the network, as the success rate of route discovery is reduced significantly as the hop distance increases, and large overhead exhausts the wireless resources with increasing network size. In source routing protocols, such as SDDRA [3], SDAR [5] and MASR [15], sender maintained up-to-date information of the network topology and constructs a data message that specifies the complete route. Non-source routing protocols like ANODR [6], MASK [1], ODAR [9] and A3RP [11], are more efficient than source routing protocols, as only one session identity (ID) is included in the data packet. With the exception of Discount-ANODR and MASK, all reviewed approaches require all network nodes to perform expensive cryptographic operation in the forward path (broadcasting RREQ message), which results in wasting both computation power and bandwidth, as only a few nodes will be selected as forwarding nodes. Destination discovery in many of the existing approaches is based on invisible implicit addressing [19]. The main disadvantage of this mechanism is that all nodes receiving the RREQ message must try to decrypt the global trapdoor to find out whether it is the intended receiver, resulting in considerable overhead. ARM [8], AnonDSR [4] and MASR [15] improve this scheme by using an index for shared key management. In MASR and AnonDSR, the key index is static, which may be traced in later route requests. ARM uses a dynamic index as the index changes on a per-request basis, however, the synchronisation of one-time pseudonyms may become an issue in practice.

3. Assumptions and Tools

3.1. Adversaries and Attack Models

We assume the adversaries are able to perform both active and passive attacks to compromise the anonymity of the network on two levels: (1) to try to reveal identities of sender, receiver and en-route nodes; (2) to try to link packets from the same communication flow. The attackers may also try to disrupt the routing process and manipulate data flows. We assume both *external* and *internal* adversaries exist [23] in the network. An *external adversary* is a wireless node that can eavesdrop, record, alter and inject packets to carry out attacks like identity spoofing, link spoofing, replay attack, man-in-the-middle attack, etc. An *internal adversary* can be a compromised en-route node (or *en-route insider*) that possesses the necessary cryptographic secret to reveal the content of a packet and to generate legitimate messages.

We also assume that the adversaries have unbounded eavesdropping capability but bounded computing and node intrusion capability, as per [16]. We note that our protocol protects anonymity in the network layer and that attacks in the physical or the application layer are beyond the scope of this paper.

3.2. Notations and Cryptographic Tools

Notations used in this paper are defined in Table 1.

3.2.1. Diffie-Hellman Key Exchange

Diffie-Hellman key exchange protocol allows two users to exchange a secret key over an insecure medium without any prior shared secrets. This mechanism is used in TARo: sender broadcasts a Diffie-Hellman public key in a RREQ message, subsequently en-route nodes reply to the sender with their public keys in the RREP messages. Sender is thus able to derive a shared key with each of the en-route nodes.

Notation	Parameter
$\mathbf{N_x}$	Node $\mathbf{X}$, where $\mathbf{N_s}, \mathbf{N_d}$ represent source and destination
F_{type}	message flag, $type = RREQ, RREP, DATA$ or $RERR$
$[.]_K$	Symmetric encryption using key K
$H(.)$	One-way hash function.
$[A\|\|B]$	Concatenation of content A and B
K_{sd}	Source-destination shared key
K_{sd}^i	i^{th} element of the source-destination key chain
PS_x	Pseudonym of node x. $PS_x = H(DH_x^p)$
DH_A^s, DH_A^p	Diffie-Hellman secret and public key generated by node A

Table 1: Notation Table

3.2.2. Keyed One-way Hash Chain

One-way hash chain was originally proposed in [24] to generate a sequence of random values, where a pair of consecutive elements in the chain are one-way linkable, i.e., $K_i = H(K_{i-1})$. The keyed one-way hash chain is a modified version of hash chain where a shared key is appended to the end of previous element to generate next element, i.e., $K_i = H(K_{i-1}||SharedKey)$ [25]. The keyed one-way hash chain generation is demonstrated in Figure 1. The $SharedKey$ is the shared secret of the nodes who are authorized to access the key chain. To prevent key collisions when same $SharedKey$ is chosen by different parties, key chain initial value K_0 should be unique system wide. In our application, we use the concatenation of two nodes' unique identifiers. Unlike the original one-way hash chain, the one-way linkable property is protected by the shared key. This additional property of keyed hash chain structure is utilized in TARo anonymous destination discovery, to prevent unauthorized nodes to derive the relationship between keys from the same key chain. Note the elements in the keyed hash chain can be generated on-demand and hence key storage is not required.

3.3. Network Assumptions

We assume that source and destination pre-share a secret in a secure way, e.g. through secure dedicated channels. For our protocol to work efficiently, we assume there are a limited number of associated destinations for each node. A more efficient and dynamic approach is to deploy a distributed

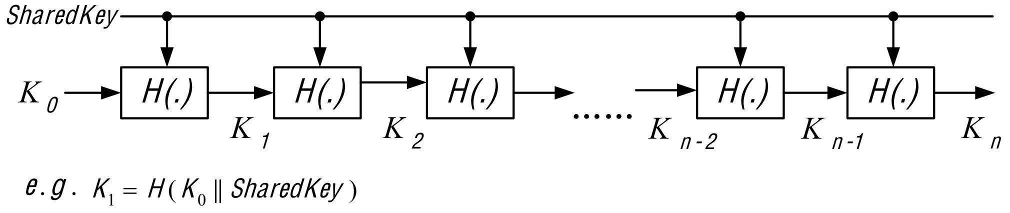


Figure 1: Keyed One-way Hash Chain Generation

anonymous key distribution system within the network, to provide source-destination share key service. The key distibution method is not in the scope of this paper. Additionally, the following parameters are known by all nodes in the network: Diffie-Hellman generator g and large prime p, message type flags: $F_{RREQ}, F_{RREQ}, F_{DATA}$ and F_{RERR}.

4. Trusted Anonymous Routing Protocol Design

TARo is a fully distributed routing protocol, which does not require a Trusted Third Party (TTP) to provide authentication services during routing, or the nodes to have prior knowledge of the entire network. The routing process is on-demand: route discovery is only initiated when a data packet arrives from upper layers and needs to be delivered to the destination node. The routing protocol sets up multiple anonymous and secure VCs to the destination.

To prevent the tracking of node's activities by attackers, permanent or long term identifiers for nodes are not exposed in the routing process, instead, a new short-lived pseudonym is created for every session. During route discovery, an onion routing mechanism is used to hide pseudonyms and keys of nodes in the path. Thus, an internal attacker with the correct keys may only reveal the previous hop and next hop information of a packet. The data message is both end-to-end and hop-by-hop encrypted. End-to-end encryption protects the security of the data packets while hop-by-hop encryption is aimed to preserve unlinkability of data flow by changing packet appearance at each hop.

Figure 2 shows a summary of the mechanisms used within the protocol to provide anonymity and security. The keyed hash chain is used for anonymous

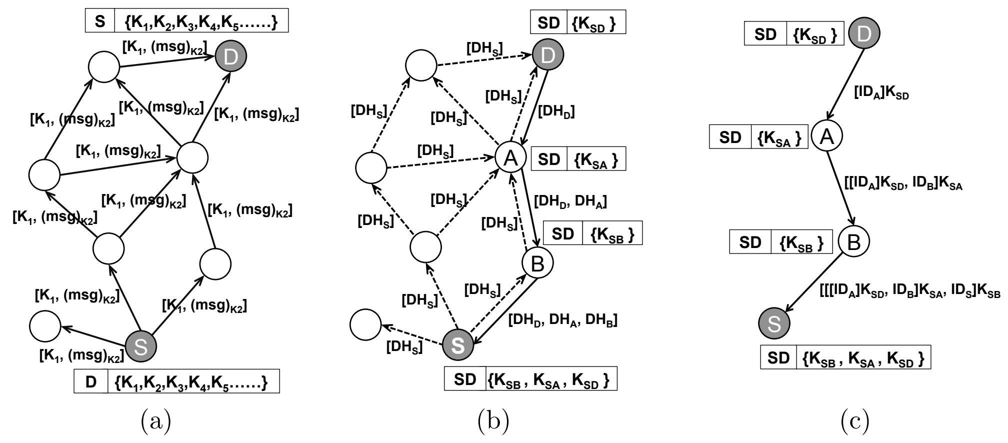


Figure 2: Protocol Overview. (a) Source discovers destination using keys from keyed hash chain; (b) Additional use of broadcast message for Diffie-Hellman key exchange. Unicast is used to forward Diffie-Hellman public keys of en-route nodes back to the source; (c) Verifying the individual connectivity of the path using encrypted topology information

route discovery. The sender node **S** uses the appropriate keys in the broadcast (forward) messages to discover the destination **D**, as per Figure 2a. We use the one to many Diffie-Hellman key exchange to establish shared keys between the source and en-route nodes as per Figure 2b. The node connections are verified using encrypted topology information exchanged between en-route nodes as per Figure 4. The exchanged shared keys from route discovery are later used for layered encryption in anonymous data transmission phase along the reverse path of Figure 2c.

4.1. Detailed Description of the Protocol

The protocol consists of three phases: anonymous route discovery, anonymous data transmission and route maintenance. The protocol flow-chart in Figure 3 shows the detailed operations performed in the source, intermediate and destination nodes during the route discovery and data transmission phases. From the operational point of view, routing is performed by manipulating information in the following four tables in each node.

1. **Destination Table** (Table 2): holds destination identifiers and corresponding key materials, i.e. keyed hash chain.

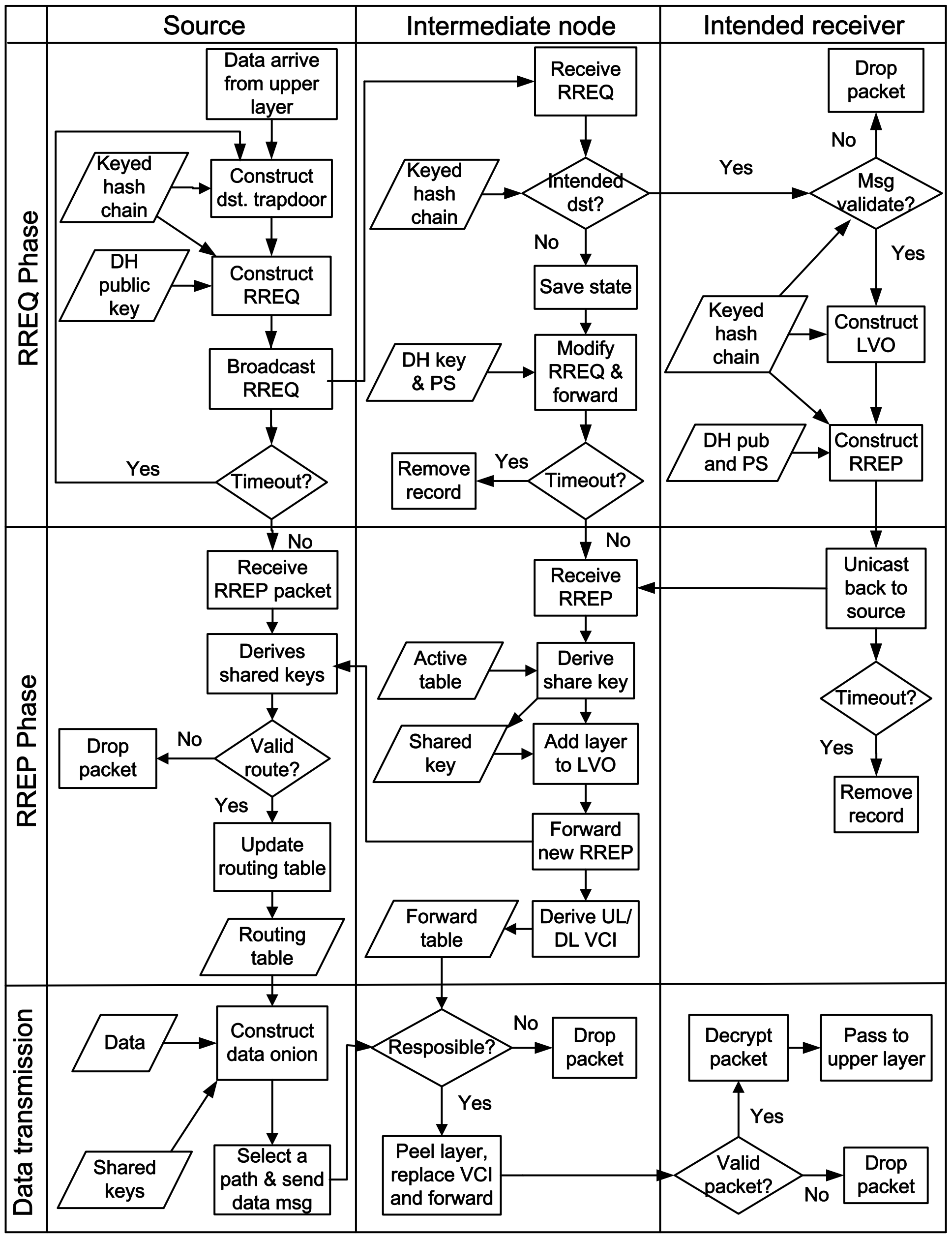


Figure 3: TARo Operation Flowchart

2. **Active Table** (Table 3): maintains current state of active routing sessions and related parameters during the route discovery phase.
3. **Forwarding Table** (Table 4): indicates how to forward, encrypt and decrypt data packets
4. **Routing Table** (Table 5): holds the next hop for each destination and layered encryption keys.

4.1.1. Anonymous Route Discovery

Each node maintains a destination table (Table 2) which contains a list of destinations with corresponding pre-shared secrets and key chains. Route discovery is triggered when a node wishes to communicate with another node in its destination table. In TARo, RREQ and RREP messages are used to: discover paths to the destination, establish shared keys between sender and en-route nodes, and set up virtual circuits for data transmission.

Node ID	*shared Key*	*Key Chain*
...	...	...

Table 2: Destination Table Attributes

Route Request

Sender constructs a RREQ message in the following format, which is then broadcast to the network.

$$(F_{RREQ}, K_{sd}^{i}, [PS_s||padding]_{K_{sd}^{i+1}}, DH_s^p, PS_i, ttl)$$

The first issue is how to find the destination anonymously and efficiently using RREQ message. In our approach, destination discovery is achieved using key pairs K_{sd}^{i} and K_{sd}^{i+1} generated by keyed one-way hash chain in both source and destination. The hash value of source and destination ID, $H(ID_s||ID_d)$, is used as initialization key K_0 (as in Figure 1) to avoid key collision. In the route request phase, K_{sd}^{i} is included in plaintext as a key index, and the consecutive key K_{sd}^{i+1} is used to encrypt a trapdoor message. Upon receiving the RREQ message, each node performs a key search in their destination table. If the key K_{sd}^{i} is found, that means the node is the intended receiver, the next key K_{sd}^{i+1} is then used to open the trapdoor and verify the message. The intermediate nodes do not need to perform any cryptographic operations.

In order to prevent replay attacks, both source and destination move to the next key pair after a single use. We note that the key index needs to be synchronized between the sender and receiver to combat packet losses. i.e. if a RREQ message is lost during transmission, source can initiate new RREQ messages using new keys from the key chain. Destination is able to verify the later messages by checking more keys along the key chain. If each node examines n consecutive keys for each destination, the system hence can tolerate up to n RREQ messages lost. The trade-off is in slowing down of the search process and increased storage memory. A new RREQ message is sent after no response for a *timeout* period. If n packets are sent and no response is received, the destination is defined as *unreachable* because source and destination are unable to synchronize further key index. To cater for different wireless environments, the value n in our scheme is adjustable in respect to the destination list size and wireless channel conditions.

The fourth field in the RREQ message DH_s^p is a fresh Diffie-Hellman public key generated by the source. The key is later used to derive shared keys with en-route nodes for the current session. PS_s and PS_i are the pseudonyms of the source and current forwarding node. Pseudonym PS_i is calculated from Diffie-Hellman public key DH_i^p with the relationship: $PS_i = H(DH_i^p)$. The concept is similar to the Cryptographically Generated Addresses (CGA) [26] that naturally binds the pseudonym to the private key. Note the source uses a random value instead of PS_s so that its neighbours cannot recognize it as the source.

The relationship between DH_i^p and PS_s in the trapdoor message allows the intended receiver to verify the integrity of the RREQ message. As PS_i can be derived from DH_i^p, random padding is added to prevent known-plaintext-attack in encrypted message, i.e. $[PS_s||padding]_{K_{sd}^{i+1}}$.

Upon receiving a RREQ message, node $\mathbf{N_i}$ proceeds with following actions:

1. $\mathbf{N_i}$ drops the packet if time to live field $ttl < 1$.
2. $\mathbf{N_i}$ matches the K_{sd}^i to session identifier (SID) in Table 3. If the SID is found, $\mathbf{N_i}$ adds the PS_i to the relay node list. The message is then dropped silently.
3. If the SID is not found, $\mathbf{N_i}$ checks whether it is the intended receiver by searching the K_{sd}^i from the destination key chains. If not found, $\mathbf{N_i}$ replaces PS_i with its own pseudonym, deducts *ttl* and forwards the modified message. Then $\mathbf{N_i}$ adds a record to the active session table:

K_{sd}^{i}, DH_s^p , DH key $pair$, PS_i, PS_{i-1}.

4. If $\mathbf{N_i}$ does not receive the corresponding RREP message after a timeout period, it removes the record from the active session table.
5. If K_{sd}^{i} is found, $\mathbf{N_i}$ first tries to decrypt the trapdoor using K_{sd}^{i+1}. The message is verified by comparing $H(DH_s^p)$ and PS_s. If the message is validated, the node enters the route reply phase.

SID	$SourceDH$	DH^P/DH^S	PS	$relay\ node\ list$	$Vtime$
...	...	...	...	...	...

Table 3: Active Session Table Attributes

Route Reply

After receiving a valid RREQ message, the destination node $\mathbf{N_d}$ constructs a RREP in the format of:

$$(F_{RREP}, K_{sd}^{i}, [PS_d||padding]_{K_{sd}^{i+1}}, PS_i, LVO_{d,i})$$

Link Verification Onion (LVO) is a data structure that contains the Diffie-Hellman public key and pseudonyms of the previous hop of each en-route node. Destination generates the core of the LVO as follows:

$$LVO_{d,n} = (DH_d^p||[PS_n||PS_d||padding]_{K_{sd}})$$

Where $\mathbf{N_n}$ is the node that forwarded the RREQ message to $\mathbf{N_d}$. K_{sd} is the shared key derived from source public key DH_s^p and destination private key DH_d^s. The purpose of including PS_n in the LVO is to ensure only $\mathbf{N_n}$ can forward the message. Each intermediate node encrypts last hop pseudonym and the current LVO, and appends the public key to construct a new LVO. Figure 4 shows an example of how LVOs are constructed and propagated. An intermediate node $\mathbf{N_i}$ constructs the $LVO_{d,i-1}$ in the following structure:

$$LVO_{d,i-1} = (DH_i^p||[PS_{i-1}||LVO_{d,i}]_{K_{si}})$$

PS_i is the pseudonym of the node which handles and forwards the RREP message. Upon receiving a RREP message, node $\mathbf{N_i}$ proceeds with following actions:

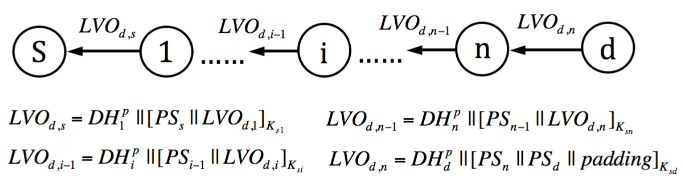

$LVO_{d,s}$
$LVO_{d,i-1}$
$LVO_{d,n-1}$
$LVO_{d,n}$
S
1
i
n
d
$LVO_{d,s} = DH_1^p \| [PS_s \| LVO_{d,1}]_{K_{s1}}$
$LVO_{d,n-1} = DH_n^p \| [PS_{n-1} \| LVO_{d,n}]_{K_{sn}}$
$LVO_{d,i-1} = DH_i^p \| [PS_{i-1} \| LVO_{d,i}]_{K_{si}}$
$LVO_{d,n} = DH_d^p \| [PS_n \| PS_d \| padding]_{K_{sd}}$

Figure 4: LVO Construction Example

1. If PS_i was used for the session, $\mathbf{N_i}$ computes the shared key and generates a new LVO as described above. $\mathbf{N_i}$ then forwards the message after replacing PS_i with the last hop pseudonym. $\mathbf{N_i}$ sends one RREP message to each node in the relay node list for multiple route discovery. Other nodes will discard the message.
2. $\mathbf{N_i}$ then computes uplink and downlink Virtual Circuit Identifiers (VCIs): $VCI_{uplink} = H(LVO_{d,i})$ and $VCI_{downlink} = H(LVO_{d,i-1})$. VCIs and the shared key are stored in Table 4. The route discovery phase is now completed and $SID = K_{sd}^i$ related record can be removed.

Downlink VCI	*Uplink VCI*	*Shared Key*	*Vtime*
...	...	...	...

Table 4: Forwarding Table Attributes

To verify the route, $\mathbf{N_s}$ derives shared keys sequentially and examines whether the hash of Diffie-Hellman public key $H(DH_i^p)$ matches the encrypted pseudonym PS_i. The route is finally verified if PS_d decrypted from $[PS_d||padding]_{K_{sd}^{i+1}}$ is the same as the PS_d from the core of LVO.

The valid route is identified by the VCI, which is computed from LVO: $VCI_{sd}^i = H(LVO_{d,s})$. The VCI and the list of corresponding shared keys are then added to the routing table (Table 5). The route discovery phase is thus completed.

4.1.2. Anonymous Data Transmission

After the route discovery phase which established multiple VCs to the destination, source node $\mathbf{N_s}$ is able to select a route to forward the data by placing corresponding VCI in the packet header. The main purpose of multipath routing is to randomize the data flow to enhance the anonymity of data

destination ID	*VCI*	*shared key list*	*Vtime*
...	...	...	...

Table 5: Routing Table Attributes

transmission against the traffic analysis attack. However, path selection can also be decided by other criteria such as quality of service (QoS), priority of the data, load balancing, etc.

To transmit data, the source builds a cryptographic onion for each data packet. Data is encrypted with the shared key of each node along the selected route in a sequence. e.g., for a forward path consisting of nodes $< \mathbf{N_s}, \mathbf{N_a}, \mathbf{N_b}, \mathbf{N_c}, \mathbf{N_d} >$, node $\mathbf{N_s}$ builds a data onion: $[[[[data]_{K_{sd}}]_{K_{sc}}]_{K_{sb}}]_{K_{sa}}$. The format of a data packet is:

$$(F_{DATA}, VCI_i, data\_onion)$$

When a node receives a data packet, it first checks whether VCI_i is in the forwarding table. The forwarding node peels one layer off the data onion by decrypting it with corresponding shared key, then replaces the downlink VCI with uplink VCI to forward the message. Other nodes ignore the packet.

4.1.3. Route Maintenance

If a link failure is detected, the event is reported to the source by sending an error (RERR) message in the format of:

$$(F_{RERR}, VCI_i, PS_i, [PS_i || PS_{i+1}]_{K_{si}})$$

The source validates the error message by opening the encrypted part with the corresponding shared key. All routes via the reported link are removed once the message is verified.

5. Protocol Evaluation

5.1. Anonymity, Trust and Security Analysis

To benchmark the performance of TARo in regards to the level of anonymity and unlinkability, we carry out a quantitative comparison with four other typical anonymous routing protocols, ANODR (TBO), Discount-ANODR,

Anonymity and Unlinkability	ANODR	Discount-ANODR	MASK	AnonDSR	TARo
Sender anonymity	yes	yes	yes	yes	yes
Receiver anonymity	yes	no	no	yes	yes
Forwarder anonymity	yes	yes	yes	yes	yes
Unlinkability: message length analysis	no	no	yes	no	yes
Unlinkability: external coding analysis	no	no	yes	no	yes
Unlinkability: internal coding analysis	no	no	no	no	yes
Unlinkability: timing analysis	no	yes	yes	no	yes
Unlinkability: profiling analysis	no	yes	yes	no	yes
Location privacy	yes	no	no	yes	yes
End-to-end data encryption	no	no	yes	yes	yes

Table 6: Qualitative Comparison of Anonymity and Unlinkability

MASK and AnonDSR in Table 6. It can be observed that the compared protocols do not consider all aspects listed in the table, or compromise anonymity for better performance.

On the other hand, TARo provides a high level of sender, receiver and intermediate node anonymity; together with random delay and traffic mixing techniques, the protocol has good resilience against traffic analysis attacks and techniques outlined in Section 2.1. The real identities of all participants are not revealed in both route discovery and data transmission in our protocol, hence the location privacy is also preserved.

The sender-receiver anonymity is maximised using keyed one-way hash chain for destination discovery as the keys K_{sd}^i in RREQ message are dynamic (one key per RREQ message) and not linkable. Public keys in the route discovery phase are self-generated at each node on per-session basis, so that adversaries cannot link them to real identities over time. Random padding or fixed control message size and random *ttl* techniques can be applied to prevent network observers to learn the hop distance from the control message by message coding and message length analysis.

In the anonymous data transmission phase, fixed size data packets are layered encrypted. Network nodes are not able to recognise the traffic flow using coding analysis techniques, as the bit pattern of a packet is changed hop-by-hop and directed by uncorrelated VCIs. The onion encryption structure protects the data flow from internal coding attack: two non-consecutive en-route internal attackers are not able to recognise a packet by observing the decrypted content.

Our protocol uses multi-path routing, which diverts the data flow and makes the profiling attack more difficult. Furthermore, without long term identities or techniques to link between pseudonyms, attackers cannot associate behaviors to any long-term identifier.

In our proposal, selecting a trusted route is not based on the previous behavior of nodes on the path but on proving connectivity between nodes. The basic idea of link verification is that a node must provide evidence from its neighbour proving that it links to this neighbour [27]. In the LVO, each en-route node encrypts the PS of the previous node, which confirms the connection between two nodes. By validating the connectivity along the LVO, the routability of the route is verified.

The defense mechanisms in TARo against some of most common passive and active attacks in MANETs [21] are discussed below. In addition, the impact of flood request Denail-of-Service (DoS) attack is also discussed.

Replay attack: An adversary can record and replay the RREQ and RREP messages, however, the replayed messages will be ignored by the network nodes as the destination discovery key is used only once.

Man-in-the-middle attack: The Diffie-Hellmen key exchange is vulnerable to man-in-the-middle attack: if an adversary replaces the DH public key with it's own in a RREQ message, the adversary can derive all shared keys and take full control of the routing. In our protocol, RREQ message trapdoor contains a public key related pseudonym so that the destination is able to verify the integrity of the message.

Link spoofing: In order to compromise the routing decision of the source, an adversary or a compromised node can falsely claim that it can route to other nodes. This kind of attack is prevented by LVO in the RREP message as it requires the nodes to confirm their connectivity relationship.

Identity spoofing: Our protocol does not use real identity for routing and data transmission, however adversary can masquerade as an en-route node, and attach it's own DH public key in the RREP phase in order to obtain a shared key. Our protocol thwarts this type of attack as the pseudonyms are linked to public key, and the corresponding private key is only known to the node that first announces the pseudonym.

Eavesdropping: The RREP and DATA messages are encrypted in onion like structure. An adversary can insert itself in a route, however, without **all** keys of the entire route, it is impossible to reveal the real content of the message.

RREQ Flooding DoS attack: Broadcast systems are susceptible to DoS attacks as an attacker is able to flood the network from a single point. An example of such attack on an anonymous MANET routing protocol is RREQ flooding: an attacker floods the network by broadcasting a large number of fake RREQ messages. As a result, the network is unable to transport normal traffic, as nodes consume an excessive amount of computation and bandwidth resources to handle the fake RREQ messages. We found that our proposed protocol is more tolerant to such an attack compared to other anonymous routing protocols. As mentioned in Section 2.2, most reviewed protocols use encrypted trapdoor in the RREQ message and require each node in the network to either perform public key operation or to test a large list of symmetric keys for every RREQ message, to find out whether it is the intended receiver. Our proposed destination discovery mechanism does not require any only required cryptographic computation but key searching, with a computational cost similar to the route decision process

5.2. Overhead and Delay Analysis

TARo is designed to minimise the overall network routing overhead in a wireless environment, with limited bandwidth and capacity. Many anonymous routing protocols perform key exchange in the route request phase: therefore, they require network nodes to append cryptographic means to the broadcasted RREQ message. Such mechanism will add a heavy burden to network capacity as the messages may grow large even after a few hops. In our approach, the broadcast RREQ message remains a constant size and the key exchange is placed in the unicast reverse path (route reply), where only the en-route nodes will have to forward the increasing (in size) RREP message. For the same reason, computational overheads are also significantly reduced, as cryptographic operations are only performed in the limited number of en-route nodes.

Unlike source routing, our approach does not introduce overhead to the data messages as the VCI only needs to be locally unique and is short enough to fit into the address field of the IP packet, e.g. 32 bits for IPv4 packets.

Network delay is also increased by cryptographic operations. Among those, public key cryptography is most costly and symmetric key and hash operations are more efficient, as per Table 9 and [28]. The design of the destination discovery mechanism in our proposal enables invisible implicit addressing without any cryptographic operations for en-route nodes and only hash and symmetric key crypto operations in the destination node. Although Diffie-Hellman key exchange mechanism is based on a special case of RSA, Diffie-Hellman key agreement performs faster than RSA public key decryption [28] and key generation can be done offline. Data forwarding also uses efficient symmetric key operation, which will not consume much computational resources for commonly available equipment like what was used in our experiments (512MHz CPU).

The overhead components for RREQ and RREP messages are shown in equations (1) and (2). Note RREQ message overhead is independent from the path length variable n as it has a constant packet size. The delays for RREQ, RREP and RREP verification by the source node can be estimated using equations (3), (4) and (5). The overall route discovery delay hence equals to the sum of the following: $T_{RREQ}+T_{RREP}+T_{VERIFY}$. The notations used in the equations are explained in Table 7.

$$L_{RREQ} = L_{flag} + L_{key} + 2L_{ps} + L_{padding} + L_{DH\_pub} + L_{ttl} \tag{1}$$

$$L_{RREP} = L_{flag} + L_{key} + 2L_{ps} + L_{padding} + n(L_{DH\_pub} + L_{ps}) \tag{2}$$

$$T_{RREQ} = T_{decrypt} + T_{hash} + n(T_{key\_search} + T_{proc}) \tag{3}$$

$$T_{RREP} = T_{enc} + (n+1)(T_{DH\_agree} + T_{enc} + T_{proc}) \tag{4}$$

$$T_{VERIFY} = n(T_{DH\_agree} + T_{decrypt}) + T_{hash} + T_{proc} \tag{5}$$

Also, additional overhead and delay may be introduced if random delay and traffic mixing options are used, which may be required to avoid the detection and linking of traffic when the network is not fully loaded.

Notation	**Parameter**
n	average path lenght
L_{flag}	size of message type header
L_{key}	size of one-way hash chain key
L_{ps}	size pseudonym
$L_{padding}$	size of padding in the trapdoor message
L_{DH_pub}	size of Diffie-Hellman public key
L_{ttl}	size of *ttl* field
T_{dec}	average symmetric key decryption time
T_{enc}	average symmetric key encryption time
T_{proc}	average process time at each hop
T_{hash}	average hash operation time
T_{key_search}	average destination key search time
T_{DH_agree}	average Diffie-Hellman key agreement time

Table 7: Notations used in the overhead and delay calculation

5.3. Performance Evaluation Using Simulation

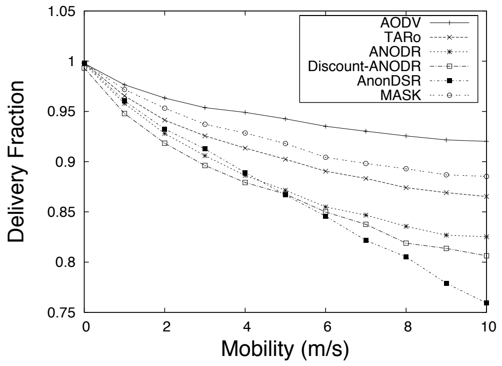
We additionally evaluate the performance of our routing protocol through simulations. Our proposal is again compared with the four target anonymous routing protocols, based on the commonly used metrics [29] over different mobility environments: (1) *Packet Delivery Fraction* - The fraction of data packets that are successfully delivered to the destination; (2) *Average Data Packet Latency* - average delay of a data packet from source to destination, including the queuing, transmission and packet handling delays; (3) *Normalized Control Bytes* - the total number of routing control packets transmitted for each delivered data packet. The performance of unsecured AODV, which is the major on-demand routing protocol for ad hoc networks, is also included in each metric as the upper bound performance boundary.

To quantitatively evaluate the performance of our scheme, we develop a Java based discrete event network simulator which considers the effect of both processing time and overheads. We adopt the network model, mobility model and network traffic model from [29]. The simulation parameters are summarised in Table 8.

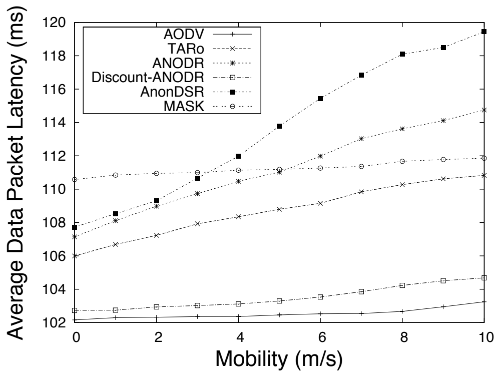
To have a fair comparision, in the simulation we unify the cryptographic mechanism, key and field size. We assume RSA (1024 bit key) is used for

Number of nodes	150	Size of field	$2400 * 600m^2$
Radio range	250m	Channel capacity	2Mbps
Mobility model	RWP	Node speed	0 - 10 m/s
Pause time	30 seconds	Data type	CBR
Packet size	512 byte	Data rate	4 packets/s
CBR sessions	5 pairs	Simulation time	15 minute

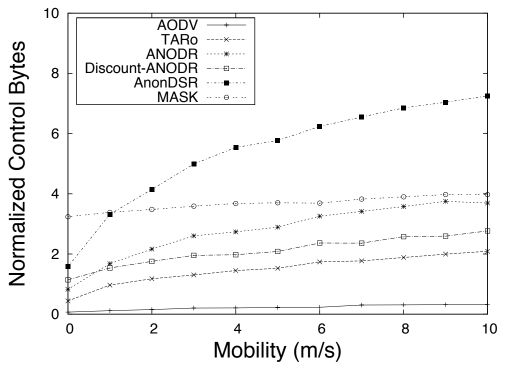
Table 8: Simulation Parameters



(a) Delivery Fraction vs. Mobility



(b) Average Data Packet Latency vs. Mobility



(c) Normalized Control Bytes vs. Mobility

Figure 5: Simulation Results and Comparison

public key system; MD5 (128 bit output) as a hash function; Diffie-Hellman (1024 bit key) for key exchange; AES (128 bit key block) for symmetric key encryption. The processing time for these cryptographic operations has been measured using OpenSSL 0.9.8g on an embedded computer (512MHz ARM processor and 256 MB memory) and is shown in Table 9. The offline processing time, such as required for key generation, key distrubution and neighbour authentication, is not considered in the data packet delay as we assume these operations are completed in the bootstrap phase before data transmission. Please note that our simulation only evaluates the impact of processing delay and cryptographic overhead for the route discovery and data transmission phases, more advanced features such as multi-path, random delay and traffic mixing in the simulated protocols are not considered. Other assumptions for various protocols are preserved, as per [29].

Operation	process time	Operation	process time
RSA 1024 encryption	1.45 ms	DH 1024 key gen	7.8 ms
RSA 1024 decryption	31.47 ms	DH 1024 key agree	14.9 ms
MD5 (1024 bit data)	0.503 ms	AES (1024 bit data)	0.191 ms

Table 9: Processing Time for Various Crypto Systems (ARM 512 MHz CPU, 256MB memory, OpenSSL 0.9.8g)

Figure 5a shows packet delivery fractions of five anonymous routing protocols and AODV. It can be observed that MASK has the best performance, this is due to the fact that it offloads the key exchange operation to neighbor authentication phase and sacrifices the receiver anonymity for efficient route discovery (see Table 6). TARo is next in performance and close to MASK. Discount-ANODR uses a bias coin flipping mechanism [2] which reduces the routing success rate and causes unstable results. The excessive delay during the route discovery phase makes it difficult for ANODR and AnonDSR to establish and maintain a route, therefore the delivery fractions are lower for these protocols.

Figure 5b illustrates the average data packet latency for the target protocols. Discount-ANODR achieves a low latency close to AODV, this is because Discount-ANODR does not provide data encryption and only one symmetric key decryption is required during packet forwarding. TARo has reasonably low latency while providing hop-by-hop data alteration and encryption and while preserving full anonymity. MASK has a steady data packet latency, introduced by both data packet encryption and decryption at forwarding nodes (as discussed above, the route discovery for MASK is very efficient). In contrast, the route discovery delay in ANODR and AnonDSR increases significantly as the mobility increases, which is reflected in the increased average data packet latency.

Benefiting from the reverse path key exchange mechanism and VC data forwarding, TARo has the smallest value of normalized control bytes among all the compared anonymous routing protocols. While ANODR and AnonDSR create a large amount of control overhead in the route request phase globally within the network, Discount-ANODR utilises most of it's control bytes in the data packet header for message forwarding. MASK again shows a stable cost of routing control overheads, as these are mostly created during the regular neighbor authentication and key exchanges.

6. Conclusions and Future Work

We have presented an anonymous routing protocol for MANETs and shown that the proposal provides both anonymity of sender, receiver and intermediate nodes and data unlinkability in regards to internal and external adversaries. The protocol is also resilient to a wide range of attacks, such as eavesdropping, identity and link spoofing, replay attack and man-in-the-middle attack. The evaluation of the proposed protocol, which was performed using analysis and simulation, shows that our protocol provides the smallest control message overhead and compares well to the existing protocols in regards to the stability of routes and latency. In future work, we plan a full implementation and experimental evaluation of the proposed protocol.

[1] W. L. Yanchao Zhang, Wei Liu, Y. Fang, MASK: Anonymous On-Demand Routing in Mobile Ad Hoc Networks, in: Transactions on Wireless Communications, Vol. 21, IEEE, 2006, pp. 2376–2385.

[2] L. Yang, M. Jakobsson, S. Wetzel, Discount Anonymous On Demand Routing for Mobile Ad Hoc Networks, in: SECURECOMM, Vol. 6, 2006.

[3] K. El-Khatib, L. Korba, R. Song, G. Yee, Secure Dynamic Distributed Routing Algorithm for Ad Hoc Wireless Networks, in: ICPP Workshops, 2003, pp. 359–366.

[4] R. Song, L. Korba, G. Yee, AnonDSR: Efficient Anonymous Dynamic Source Routing for Mobile Ad-hoc Networks, in: SASN '05, 2005, pp. 33–42.

[5] A. Boukerche, K. El-Khatib, L. Xu, L. Korba, SDAR: A Secure Distributed Anonymous Routing Protocol for Wireless and Mobile Ad Hoc Networks, Annual IEEE Conference onLocal Computer Networks 0 (2004) 618–624.

[6] J. Kong, X. Hong, ANODR: anonymous on demand routing with untraceable routes for mobile ad-hoc networks, in: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing, MobiHoc '03, ACM, 2003, pp. 291–302.

[7] B. Zhu, Z. Wan, M. S. Kankanhalli, F. Bao, R. H. Deng, Anonymous Secure Routing in Mobile Ad-Hoc Networks, in: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks, LCN '04, IEEE Computer Society, 2004, pp. 102–108.

[8] S. Seys, B. Preneel, ARM: Anonymous Routing Protocol for Mobile ad hoc Networks, Int. J. Wire. Mob. Comput. 3 (2009) 145–155.

[9] D. Sy, R. Chen, L. Bao, ODAR: On-Demand Anonymous Routing in Ad Hoc Networks, in: In Proceedings of The Third IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS, 2006, pp. 267–276.

[10] A. Boukerche, Y. Ren, ARMA: An Efficient Secure Ad Hoc Routing Protocol, in: GLOBECOM, 2007, pp. 1268–1272.

[11] J. H. Paik, B. H. Kim, D. H. Lee, A3RP: Anonymous and Authenticated Ad Hoc Routing Protocol, in: Proceedings of the 2008 International Conference on Information Security and Assurance (isa 2008), IEEE Computer Society, 2008, pp. 67–72.

[12] V. Fusenig, D. Spiewak, T. Engel, Acimn Protocol: a Protocol for Anonymous Communication in Multi-hop Wireless Networks, in: Proceedings of AISC, Australian Computer Society, Inc., 2008, pp. 107–114.

[13] J. Luo, X. Wang, M. Yang, A Resilient P2P Anonymous Routing Approach Employing Collaboration Scheme, J. UCS 15 (9) (2009) 1797–1811.

[14] J. Ghaderi, R. Srikant, Towards a Theory of Anonymous Networking, in: Proceedings of the 29th conference on Information communications, INFOCOM'10, IEEE Press, 2010, pp. 686–694.

[15] J. L. Jun Pan, MASR: An Efficient Strong Anonymous Routing Protocol for Mobile Ad Hoc Networks, Vol. 22, 2009, pp. 644–654.

[16] J. Kong, Mobility Changes Anonymity: Mobile Ad Hoc Networks Need Efficient Anonymous Routing, in: Proceedings of the 10th IEEE Symposium on Computers and Communications, IEEE Computer Society, 2005, pp. 57–62.

[17] A. Pfitzmann, M. Hansen, Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management - A Consolidated Proposal for Terminology, Tech. rep. (February 2008).

[18] J. T. Chen, R. Boreli, V. Sivaraman, TARo: Trusted Anonymous Routing for MANETs, in: Sixth IEEE/IFIP International Symposium on Trusted Computing and Communications (TrustCom-10), IEEE Computer Society, 2010, pp. 756–762.

[19] A. Pfitzmann, M. Waidner, Networks without User Observability-Design Options, EUROCRYPT '85 (1986) 245–253.

[20] W. Diffie, M. E. Hellman, New directions in cryptography, IEEE Transactions on Information Theory 22 (6) (1976) 644–654.

[21] H. Deng, W. Li, D. P. Agrawal, Routing security in wireless ad hoc networks, IEEE Communications Magazine 40 (10).

[22] M. Rennhard, B. Plattner, S. Rafaeli, L. Mathy, D. Hutchison, An Architecture for an Anonymity Network, Enabling Technologies, IEEE International Workshops on (2001) 165.

[23] D. Zhou, Security issues in ad hoc networks, CRC Press, Inc., 2003, pp. 569–582.

[24] L. Lamport, Password authentication with insecure communication, Commun. ACM 24 (1981) 770–772.

[25] Y. chun Hu, M. Jakobsson, A. Perrig, Efficient Constructions for One-way Hash Chains, in: In Applied Cryptography and Network Security (ACNS), 2003, pp. 423–441.

[26] T. Aura, Cryptographically Generated Addresses (CGA), Tech. Rep. 3972, RFC Editor (Mar. 2005).

[27] T. Chen, O. Mehani, R. Boreli, Trusted Routing for VANET, in: IEEE ITST'09, IEEE Computer Society, 2009.

[28] W. Dai, Speed Comparison of Popular Crypto Algorithms, crypto++ Library 5.6.0 Benchmarks (2009).

[29] A. Boukerche, Y. Ren, Z. Zhang, Performance Evaluation of an Anonymous Routing Protocol using Mobile Agents for Wireless Ad hoc Networks, in: Proceedings of the 32nd IEEE Conference on Local Computer Networks, IEEE Computer Society, 2007, pp. 893–900.