

# TARo: Trusted Anonymous Routing for MANETs

Jiefeng (Terence) Chen

National ICT Australia

Locked Bag 9013, Alexandria  
NSW 1435, Australia, and  
University of New South  
Wales, Australia,  
terence.chen@nicta.com.au

Roksana Boreli

National ICT Australia

Locked Bag 9013, Alexandria  
NSW 1435, Australia, and  
University of New South  
Wales, Australia,  
roksana.boreli@nicta.com.au

Vijay Sivaraman

School of Electrical Engineering

and Telecommunications

University of New South Wales  
Sydney, NSW 2052, Australia

vijay@unsw.edu.au

**Abstract**—The currently proposed anonymous routing mechanisms for Mobile Ad hoc Networks enable network entities to anonymously and securely communicate with each other. However, protocols that provide a high level of anonymity generally have poor scalability due to delays and overhead introduced by cryptographic operations, while other approaches sacrifice anonymity to achieve better performance. In this paper, we propose a novel 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 a novel 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.

**Index Terms**—MANET; onion routing, anonymity, security, trust

## I. INTRODUCTION

In Mobile Ad hoc Networks (MANETs), mobile nodes communicate using peers, without the aid of any pre-existing infrastructure. Due to the openness and cooperative nature of such networks, MANETs are vulnerable to a wide range of threats [1] to user's identity and data, therefore anonymity and security are essential for such networks.

Pfitzmann and Hansen define *anonymity* in [2] as "the state of being not identifiable within a set of subjects". In MANET data communication, anonymity means that the identities of source, destination and the route of a data message cannot be linked to any node within the network. A related requirement is *unlinkability* [2], i.e. it is necessary to ensure that data packets from a single data flow cannot be linked in order to trace the origin and the destination of this flow.

MANETs often have limited power and processing capabilities. 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 sacrifice anonymity to achieve better performance [3], [4], or disregard unlinkability [4].

In this paper we present a Trusted Anonymous Routing (TARo) protocol 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 TARo, we propose a novel solution for invisible implicit addressing [16] based on *keyed hash chain*. Shared keys between the sender and intermediate nodes are exchanged using the Diffie-Hellman mechanism [17] and these are later used to ensure unlinkability of data by per-hop data packet appearance alteration. TARo is also able to detect untrusted connectivity announcements by using the proposed Link Verification Onion (LVO) mechanism.

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

## II. ANONYMOUS ROUTING FOR MANET

There have been a number of anonymous routing protocols proposed for MANET in the past years [3] - [15]. They most commonly include the following four phases.

**Anonymous neighbour authentication:** Nodes establish trust relationships and broadcast/shared keys with one-hop neighbors 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 encryption ([5], [8] and [6]), pseudonyms ([8], [9] and [13]) and invisible implicit

addressing ([6], [7]). Some protocols, like SDAR [6], MASK [3] and Discount-ANODR [4], partially violate the anonymity requirement as they use real identities of participating nodes in order to achieve improved performance. E.g. in [6] 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 [3] and [4], 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, protocols utilise techniques like per-hop packet appearance alteration ([3], [8], [10], [12], [15]), use of fixed packet size achieved by padding ([7], [10], [14]), random delay [3], random forwarding [18] and traffic mixing [19].

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 [5], SDAR [6] and MASR [15], sender knows identities of en-route nodes to the destination and constructs a data message that specifies the complete route. Non-source routing protocols like ANODR [9], MASK [3], ODAR [11] and A3RP [13], 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 [16]. 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 [10], AnonDSR [8] 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.

### III. ASSUMPTIONS AND TOOLS

#### A. 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 [20] 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 [1]. 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.

#### B. Notations and Cryptographic Tools

Notations used in this paper are defined in Table I.

Notation	Explanation
$\mathbf{N}_x$	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 $x$ . $PS_x = H(DH_A^P)$
$DH_A^s, DH_A^P$	Diffie-Hellman secret and public key generated by node A

TABLE I: Notation Table

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.

2) *Keyed One-way Hash Chain*: One-way hash chain was originally proposed in [21] 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.,  $K_i = H(K_{i-1} \parallel \text{sharedkey})$  [22]. The keyed one-way hash chain generation is demonstrated in Figure 1. The elements in the keyed hash chain can be generated on-demand and hence key storage is not required.

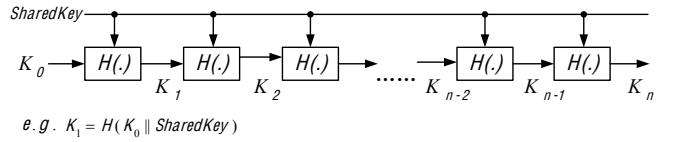


Fig. 1: Keyed One-way Hash Chain Generation

#### C. Network Assumptions

We assume that source and destination 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. Additionally,

the following parameters are known by all nodes in the network: Diffie-Hellman generator  $g$  and large prime  $p$ , message type flags:  $F_{RREQ}$ ,  $F_{RREP}$ ,  $F_{DATA}$  and  $F_{ERR}$ .

#### IV. TRUSTED ANONYMOUS ROUTING PROTOCOL DESIGN

In this section, we describe our proposed trusted anonymous routing protocol. The protocol consists of three phases: anonymous route discovery, anonymous data transmission and route maintenance. We do not use anonymous neighbor authentication due to the high overhead it creates and instead TARo performs authentication and key exchange on-demand. Routing related information is stored, in each node, within the following tables:

- 1) **Destination Table:** holds destinations and key materials.
- 2) **Active Table:** maintains current active routing sessions and related parameters during the route discovery phase.
- 3) **Forwarding Table:** indicates how to decrypt and forward packets.
- 4) **Routing Table:** holds the next next hop for each destination and related parameters.

Figure 2 presents an overview of the protocol operation.

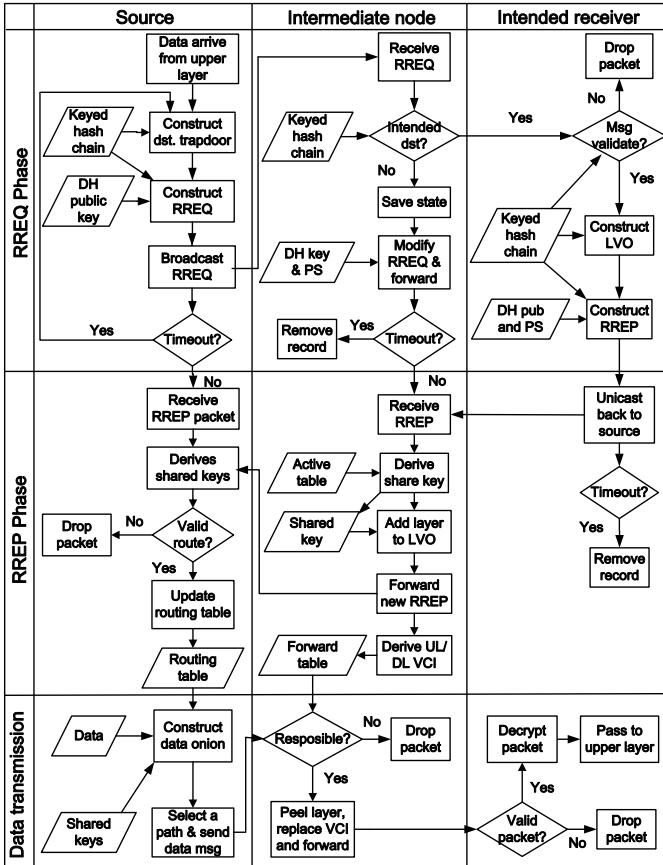


Fig. 2: TARo Operation Flow Chart

##### A. Anonymous Route Discovery

Each node maintains a destination table (Table II) 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 II: Destination Table Attributes

1) **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. 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 forth 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) [23] 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 allows the intended receiver to verify the integrity of the message. As  $PS_i$  can be derived from  $DH_i^p$ , random padding is used to prevent known-plaintext-attack [24].

Upon receiving a RREQ message, node  $N_i$  proceeds with following actions:

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

2) *Route Reply*: After receiving a valid RREQ message, the destination node  $N_d$  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  $N_n$  is the node that forwarded the RREQ message to  $N_d$ .  $K_{sd}$  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  $N_n$  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 3 shows an example of how LVOs are constructed and propagated. An intermediate node  $N_i$  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  $N_i$  proceeds with following actions:

- 1) If  $PS_i$  was used for the session,  $N_i$  computes the shared key and generates a new LVO as described above.  $N_i$

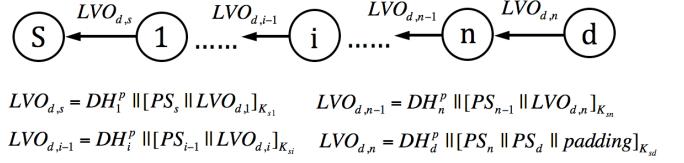


Fig. 3: LVO Construction Example

then forwards the message after replacing  $PS_i$  with the last hop pseudonym.  $N_i$  sends one RREP message to each node in the relay node list for multiple route discovery. Other nodes will discard the message.

- 2)  $N_i$  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 IV. 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 IV: Forwarding Table Attributes

To verify the route,  $N_s$  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 V). The route discovery phase is thus completed.

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

TABLE V: Routing Table Attributes

#### B. Anonymous Data Transmission

After the route discovery phase which established multiple VCs to the destination, source node  $N_s$  is able to select a route to forward the data by placing corresponding VCI in the packet header. The main purpose of multi-path routing is to randomize the data flow to enhance the anonymity of data 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  $< N_s, N_a, N_b, N_c, N_d >$ , node  $N_s$  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.

### C. 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.

## V. PROTOCOL EVALUATION

### A. Anonymity, Trust and Security Analysis

The protocol we are proposing 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.

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 nodes to learn the hop distance by message coding and message volume analysis.

In the anonymous data transmission phase, each data packet is encrypted in an onion data structure [25]. Network nodes are not able to recognise the traffic flow, as a packet is changed hop-by-hop and directed by uncorrelated VCIs. Two non-consecutive en-route insiders 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 traffic analysis based detection more difficult.

As some protocols compromise the anonymity for improved performance, it is necessary to benchmark the performance of our protocol in regards to the level of anonymity and unlinkability. A qualitative comparison with four other typical anonymous routing protocols, ANODR (TBO), Discount-ANODR, MASK and AnonDSR is provided in Table VI. It can be observed that our proposed protocol provides anonymity for all nodes and unlinkability protection against both external and internal attacks.

In our proposal, selecting a trusted route is not based on the previous behavior of nodes on the path but on the enforcement of 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 [26]. In the LVO, each en-route node encrypts the PS of the previous node, which confirms the connection between two nodes. By

Anonymity and Unlinkability	ANODR	Discount-ANODR	MASK	AnonDSR	TARo
Sender anonymity	yes	yes	yes	yes	yes
Receiver anonymity	yes	no	no	yes	yes
Forwarding node anonymity	yes	yes	yes	yes	yes
Data end-to-end encryption	no	no	yes	yes	yes
Unlinkability: external coding attack	no	no	yes	no	yes
Unlinkability: internal coding attack	no	no	no	no	yes
Unlinkability: timing attack	no	yes	yes	no	yes

TABLE VI: Qualitative Comparison of Anonymity and Unlinkability

validating the connectivity along the LVO, the routability of the route is verified.

The proposed routing protocol is secure against following most common passive and active attacks in MANETs [1].

**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.

### B. 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 VIII and [27]. 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 [27] 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).

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.

### C. 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 [28] 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 base 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 [28]. The simulation parameters are summarised in Table VII.

To have a fair comparison, in the simulation we unify the cryptographic mechanism, key and field size. We assume RSA (1024 bit key) is used for public key system; MD5 (128 bit output) as a hash function; Diffie-Hellman (1024 bit key)

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 VII: Simulation Parameters

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 VIII. The offline processing time, such as required for key generation, key distribution 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 [28].

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

Figure 4a 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 VI). TARo is next in performance and close to MASK. Discount-ANODR uses a bias coin flipping mechanism [4] 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 4b 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,

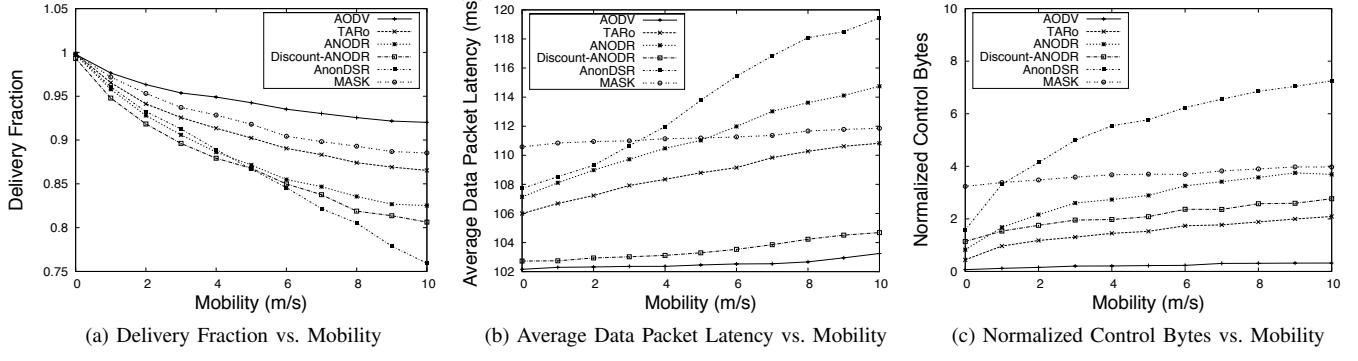


Fig. 4: Simulation Results and Comparison

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 its 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.

## VI. CONCLUSIONS AND FUTURE WORK

We have presented a novel 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. Protocol evaluation, done both analytically and using 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.

## VII. ACKNOWLEDGEMENT

This research work has been supported by funding from National ICT Australia (NICTA).

## REFERENCES

- [1] J. Kong, X. Hong, M. Y. Sanadidi, and M. Gerla, "Mobility Changes Anonymity: Mobile Ad Hoc Networks Need Efficient Anonymous Routing," in *ISCC*, 2005, pp. 57–62.
- [2] A. Pfitzmann and M. Hansen, "Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management - A Consolidated Proposal for Terminology," Tech. Rep., February 2008.
- [3] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "MASK: Anonymous On-demand Routing in Mobile Ad Hoc Networks," *IEEE Transactions on Wireless Communications*, no. 9, 2006.
- [4] L. Yang, M. Jakobsson, and S. Wetzel, "Discount Anonymous On Demand Routing for Mobile Ad Hoc Networks," *SECURECOMM*, vol. 6, 2006.
- [5] K. El-Khatib, L. Korba, R. Song, and G. Yee, "Secure Dynamic Distributed Routing Algorithm for Ad Hoc Wireless Networks," in *ICPP Workshops*, 2003, pp. 359–366.
- [6] A. Boukerche, K. El-Khatib, L. Xu, and L. Korba, "SDAR: A Secure Distributed Anonymous Routing Protocol for Wireless and Mobile Ad Hoc Networks," in *IEEE LCN '04*, 2004.
- [7] B. Zhu, Z. Wan, M. S. Kankanhalli, F. Bao, and R. H. Deng, "Anonymous Secure Routing in Mobile Ad-Hoc Networks," in *LCN '04*, 2004.
- [8] R. Song, L. Korba, and G. Yee, "AnonDSR: Efficient Anonymous Dynamic Source Routing for Mobile Ad-hoc Networks," in *SASN '05*, 2005, pp. 33–42.
- [9] J. Kong and X. Hong, "ANODR: anonymous on demand routing with untraceable routes for mobile ad-hoc networks," in *MobiHoc '03*, 2003.
- [10] S. Seys and B. Preneel, "ARM: Anonymous Routing Protocol for Mobile Ad hoc Networks," in *AINA '06*, 2006.
- [11] D. Sy, R. Chen, and L. Bao, "ODAR: On-Demand Anonymous Routing in Ad Hoc Networks," in *MASS*, 2006.
- [12] A. Boukerche and Y. Ren, "ARMA: An Efficient Secure Ad Hoc Routing Protocol," in *GLOBECOM*, 2007, pp. 1268–1272.
- [13] J. H. Paik, B. H. Kim, and D. H. Lee, "A3RP : Anonymous and Authenticated Ad Hoc Routing Protocol," in *International Conference on Information Security and Assurance*, 2008, 2008.
- [14] V. Fusenig, D. Spiewak, and T. Engel, "Acimn Protocol: A Protocol for Anonymous Communication in Multi-Hop Wireless Networks," in *AISC '08*, 2008.
- [15] J. L. Jun Pan, "MASR: An Efficient Strong Anonymous Routing Protocol for Mobile Ad Hoc Networks," 2009.
- [16] A. Pfitzmann and M. Waidner, "Networks without user observability—design options, booktitle = EUROCRYPT '85," pp. 245–253, 1986.
- [17] W. Diffie and M. E. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, 1976.
- [18] J. Luo, X. Wang, and M. Yang, "A Resilient P2P Anonymous Routing Approach Employing Collaboration Scheme," *J. UCS*, 2009.
- [19] J. Ghaderi and R. Srikant, "Towards a Theory of Anonymous Networking," *CoRR*, 2009.
- [20] J. A. Clark, J. Murdoch, J. A. Mcdermid, S. Sen, H. R. Chivers, O. Worthington, and P. Rohatgi, "Threat Modelling for Mobile Ad Hoc and Sensor Networks," 2007.
- [21] L. Lamport, "Password authentication with insecure communication," *Commun. ACM*, November 1981.
- [22] Y. chun Hu, M. Jakobsson, and A. Perrig, "Efficient Constructions for One-way Hash Chains," in *ACNS'03*, 2003.
- [23] T. Aura, "Cryptographically Generated Addresses (CGA)," in *ISC*, 2003.
- [24] Wikipedia.org, "Known-plaintext attack," 2010, [http://en.wikipedia.org/wiki/Known-plaintext\\_attack](http://en.wikipedia.org/wiki/Known-plaintext_attack).
- [25] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, "Proxies For Anonymous Routing," in *ACSAC '96*, 1996.
- [26] T. Chen, O. Mehani, and R. Boreli, "Trusted Routing for VANET," in *IEEE ITST'09*. IEEE Computer Society, 2009.
- [27] W. Dai, "Speed Comparison of Popular Crypto Algorithms," 2009, crypto++ Library 5.6.0 Benchmarks.
- [28] J. Liu, J. Kong, X. Hong, and M. Gerla, "Performance Evaluation of Anonymous Routing Protocols in Mobile Ad-hoc Networks," in *WCNC, Las Vegas*, 2005.