An Anonymous On Demand Routing Protocol with Untraceable Routes for Mobile Ad-hoc Networks

Jiejun Kong, Xiaoyan Hong, Mario Gerla
Computer Science Department
University of California, Los Angeles, CA 90095
{jkong,hxy,gerla}@cs.ucla.edu

Abstract

In hostile environments, the enemy can launch traffic analysis against interceptable routing information embedded in routing messages and data packets. Allowing adversaries to trace network routes and infer the motion pattern of nodes at the end of those routes may pose a serious threat to covert operations. We propose ANODR, an anonymous on-demand routing protocol for mobile ad hoc networks deployed in hostile environments. We address two closely-related problems: For *route anonymity*, ANODR prevents strong adversaries from tracing a packet flow back to its source or destination; for *location privacy*, ANODR ensures that adversaries cannot discover the real identities of local transmitters. The design of ANODR is based on "*broadcast with trapdoor information*", a novel network security concept which includes features of two existing network and security mechanisms, namely "broadcast" and "trapdoor information". We use simulations and implementation to validate the effectiveness of our design.

1. INTRODUCTION

In hostile environments, allowing adversaries to trace network routes and nodes at the end of those routes may pose serious threats to the success of covert missions. Consider for example a battlefield scenario with ad hoc, multi-hop wireless communications support. Suppose a covert mission is launched, which includes swarms of reconnaissance, surveillance, and attack task forces. The ad hoc network must provide routes between command post and swarms (for delivery of reliable commands/controls from commander to swarms and for situation data/video reporting from swarms to the commander) as well as routes between swarms (data fusion, failure recovery, threat evasion etc). Providing anonymity and location privacy supports for the task forces is critical, else the entire mission may be compromised. This poses challenging constraints on routing and data forwarding. In fact, the adversary could deploy reconnaissance and surveillance forces in the battlefield and maintains communications among them. They could form their own network to infer the location, movement, number of participants, and even the goals of our covert missions.

On-demand routing schemes are more "covert" in nature in that they do not advertise in advance—they just set up routes as needed. Nevertheless, the enemy may gain a lot of information about the mission by analyzing on-demand routing information and observing packet flows once the connection is established. Since a necessary byproduct of any mission, whether covert or not, is communications across swarms and to/from command post, these flows and the routes temporarily set up at intermediate nodes must be protected from inference and intrusion.

The purpose of this paper is to develop "untraceable" routes or packet flows in an *on-demand* routing environment. This goal is very different from other related routing security problems such as resistance to route disruption or prevention of "denial-of-service" attacks. In fact, in our case the enemy will avoid such aggressive schemes, in the attempt to be as "invisible" as possible, until it traces, locates, and then physically destroys the assets. We address the untraceable routing problem by a route pseudonymity approach. In our design, the anonymous route discovery process establishes an on-demand route between a source and its destination. Each hop en route is associated with a random *route pseudonym*. Since data forwarding in the network is based on route pseudonyms with negligible overhead, local senders and receivers need not reveal their identities in wireless transmission. In other words, the route pseudonymity approach allows us to "unlink" (i.e., thwart inference between) network member's location and identity. For each route, we also ensure unlinkability among its route pseudonyms. As a result, in each locality eavesdroppers or any bystander other than the forwarding node can only detect the transmission of wireless packets stamped with random route pseudonyms. It is hard for them to trace how many nodes in the locality, who is the

TABLE I

TABLE OF VARIABLES AND NOTATION

PK_A	Node A's public key	K_A	An encryption key only known by node A
SK_A	Node A's private key corresponding to PK_A	K_{AB}	An encryption key shared by node A and B
$\{M\}_{PK_A}$	Encrypting/verifying message M using public key PK_A	N_A, N_A^i	Nonce or nonces chosen by node A
$[M]_{SK_A}$	Decrypting/signing message M using private key SK_A	RREQ	Route Discovery Request Packet
$K(M)$	Encrypting/decrypting message M using symmetric key K	RREP	Route Discovery Reply Packet
src	a special tag denoting the source	RERR	Route Maintenance Error Packet
$dest$	a special tag denoting the destination	,	concatenation of appropriately formatted bit strings

transmitter or receiver, where a packet flow comes from and where it goes to (i.e., what are the previous hops and the next hops en route), let alone the source sender and the destination receiver of the flow. We further tackle the problem of node intrusion within the same framework. In our design a strong adversary with node intrusion capability must carry out a complete "vertex cover" process to trace each on-demand ad hoc route.

The design of route pseudonymity is based on a network security concept called "*broadcast with trapdoor information*", which is newly proposed in this work. Multicast/broadcast is a network-based mechanism that has been explored in previous research [50], [51] to provide recipient anonymity support. Trapdoor information is a security concept that has been widely used in encryption and authentication schemes. ANODR is realized upon a hybrid form of these two concepts.

The contribution of this work is to present a *untraceable and intrusion tolerant routing protocol* for mobile ad hoc networks.

- *Untraceability*: ANODR dissociates ad hoc routing from the design of network member's identity/pseudonym. The enemy can neither link network members' identities with their locations, nor follow a packet flow to its source and destination. Though the adversaries may detect the existence of local wireless transmissions, it is hard for them to infer a covert mission's number of participants, as well as the transmission pattern and motion pattern of these participants.
- *Intrusion tolerance*: ANODR ensures there is no single point of compromise in ad hoc routing. Node intrusion does not compromise location privacy of other legitimate members, and an on-demand ANODR route is traceable only if all forwarding nodes en route are intruded.
- *Efficiency*: Unlike network nodes in the wired infrastructure, nodes in mobile ad hoc networks are characterized by limited energy, computation, and communication resources. Moreover, the communications have often very tight latency requirements. ANODR addresses these concerns and is suitable for mobile ad hoc networks where many network members are running on low-end devices. In particular, during the on-demand RREQ flooding phase, there is no computational overhead caused by expensive public key operations.

The rest of the paper is organized as follows. Section 2 describes the underlying models and useful tools to realize our scheme. The design framework and related discussions are illustrated in details in Section 3. Then we present untraceability analysis in Section 5. Our implementation and performance evaluation are shown in Section 6. In Section 7 we compare our work with related anonymity research. Finally Section 8 concludes this paper.

2. UNDERLYING MODELS AND TOOLS

2.1. Notations

In the paper we will use the notations shown in Table I.

2.2. Adversary and attack model

Passive eavesdroppers may be omnipresent in a hostile environment where ANODR is deployed. For example, nowadays technology has implemented wireless interface on low-cost sensor nodes (e.g., Motorola ColdFire, Berkeley Mote) that can be planted in ad hoc networks to monitor ongoing activities. However, an adversary with unbounded computing and active interference capability is capable of overwhelming any practical security protocol. Thus we design our schemes to be secure against a powerful adversary with *unbounded* eavesdropping capability but *bounded* computing and node intrusion capability.

- *Link intrusions*: An adversary at this level is an *external adversary* that poses threat to wireless link only. The adversary knows and actualizes all network protocols and functions. It can eavesdrop, record, inject, re-order, and re-send (altered) wireless packets. (i) The adversary can access its computational resources via a fast network with negligible delay (e.g., using directional antenna or ultra-wideband communication). This implies that collaborative adversaries can also contact each other in short latency. (ii) However, their computational resources may be abundant, but not unbounded. Network members can employ public key cryptosystems (e.g., RSA, El Gamal) and symmetric key cryptosystems (e.g., 3DES, AES) to protect critical messages. They can also employ efficient message authentication protocols (e.g., TESLA [41]) to get rid of unauthenticated and out-of-date packets injected by the adversary.
- *Node intrusions*: An adversary at this level is an *internal adversary* that also poses threat to network members. (i) After the adversary compromises a victim node, it can see the victim's currently stored records including the private route caches. (ii) The adversary may move from one node to another over time (i.e., *mobile adversary* proposed in previous research [18]). However, its capability to intrude legitimate members is not unbounded. During a time window T_{win} it cannot successfully compromise more than K members. (iii) Intrusion detection is not perfect. A *passive internal adversary* exhibiting no malicious behavior will stay in the system and intercept all routing messages. This means encrypting routing messages cannot stop a passive internal adversary.

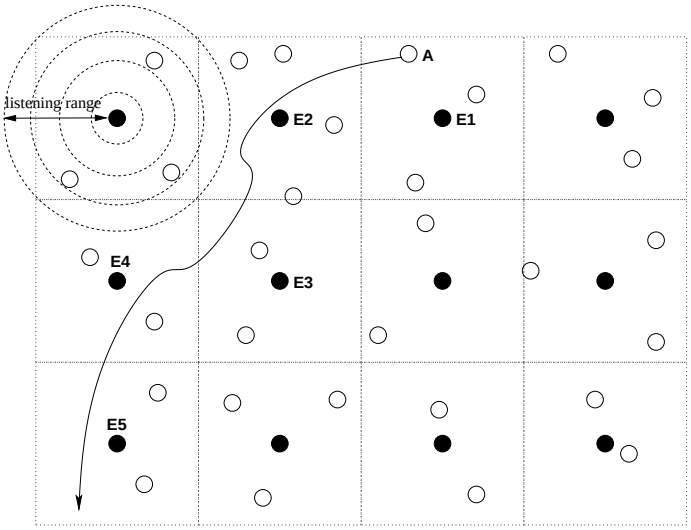

Fig. 1. **Passive traffic analysis: a referential scenario** (Collaborative adversarial nodes are depicted as solid black nodes)

Figure 1 shows the referential case for collaborative adversaries to trace the motion pattern of a mobile node. A collection of adversarial nodes can be (pre-)deployed to cover a region or even the entire ad hoc network. As depicted in Figure 1, the adversaries can divide the network into cells based on radio receiving range. One or more adversarial nodes can effectively monitor each cell. Any open wireless transmission within one-hop transmission range is thus collected and fed back to adversary's computing center for further analysis. The examples below demonstrate various passive attacks that can be launched by the adversaries.

Example 1: (Motion pattern inference attack) As implied by the name, the goal of this passive attack is to infer (possibly imprecise) motion pattern of mobile nodes. For example, the adversaries $\{E1, E2, E3, E4, E5\}$ can collaborate with each other using efficient communication means (e.g., directional antenna and ultra-wideband transmission). By monitoring wireless packets in and out a specific mobile node (say A), they can combine the intercepted data and trace the motion pattern of node A.

Example 2: (Location privacy attack) Given a specific cell L, the adversary may gather and quantify (approximated) information about active mobile nodes, for example, (a) the set of active nodes in L; (b) related quantification such as the size of the set; (c) traffic analysis against L, e.g., how many and what kind of connections in-and-out

the cell.

Example 3: (Route tracing attack) Route tracing attacks seek to monitor ad hoc route information against a specific node V. This includes (a) tracing the ad hoc routes in-and-out node V, (b) traffic analysis against node V, e.g., how many and what kind of connections in and out node V. The latter one belongs to traditional identity anonymity problem. For example, given the node V, the adversaries want to know (b.1) with whom node V has communicated, and (b.2) the sessions/transactions node V has initiated as sender and responded as recipient.

Existing end-to-end security protocols are not valid answers to the passive attacks. In end-to-end security, two communicating nodes can ensure content privacy by encrypting their application data payload. The cryptographic design cannot stop attacks based on traffic analysis. In addition, as lower-layer routing information, such as MAC header and IP header, is not encrypted and protected, passive adversaries can ignore the cryptographic design and efficiently trace mobile nodes.

Encrypting lower-layer routing information seems to be a good solution. Basagni et al. [3] propose a solution to encrypt routing messages using a network-wide symmetric key shared by all legitimate members. However, the encryption based solution has several drawbacks:

- In a wireless network using broadcast channel, encrypted routing headers may render the cost of data packet forwarding to be potentially very high, as each node has to decrypt and check any data packet before it can forward the packet. If such a scheme is used, one encryption/transmission always causes multiple receptions/decryptions at all local neighbors (though only one of them is the intended forwarder).
- Encryption cannot completely stop traffic analysis. External adversaries can use timing analysis to launch route tracing attack as usual. In *timing analysis*, the adversary can monitor network-wide transmission events with their timing information recorded. The adversary can use temporal dependency between transmissions to trace a victim message's forwarding path. For example, two packets transmitted in and out of a forwarding node at time t and $t + \epsilon$ are likely from the same packet flow.
- Problems caused by passive internal adversary are not solved. The internal adversaries know the secret keys that they possess. Data encryption is a good solution to protect secret plaintexts, but it does not necessarily offer protection when the related secret keys are revealed.

2.3. Nomenclature

Throughout the paper we will address the anonymous routing problem based on the nomenclature introduced by earlier related work. In particular, we refer to Pfitzmann and Köhntopp [42] who define the concept of *pseudonymity* and the concept of *anonymity* in terms of *unlinkability* or *unobservability*.

In a computer network, entities are identified by unique IDs. Network transmissions are treated as the *items of interest* (IOIs). *Pseudonym* is an identifier of subjects to be protected. It could be associated with a sender, a recipient, or any protégé demanding protection. The concept of *pseudonymity* is defined as the use of pseudonyms as IDs. The concept of *anonymity* is defined in terms of either *unlinkability* or *unobservability*. The difference between unlinkability and unobservability is whether security protection covers IOIs or not:

- *Unlinkability*: Anonymity in terms of unlinkability is defined as unlinkability of an IOI and a pseudonym. An anonymous IOI is not linkable to any pseudonym, and an anonymous pseudonym is not linkable to any IOI. More specifically, *sender anonymity* means that a particular transmission is not linkable to any sender's pseudonym, and any transmission is not linkable to a particular sender's pseudonym. *Recipient anonymity* is similarly defined.
  A property weaker than these two cases is *relationship anonymity* where two or more pseudonyms are unlinkable. In particular for senders and recipients, it is not possible to trace who communicates with whom, though it may be possible to trace who is the sender, or who is the recipient. In other words, sender's pseudonym and recipient's pseudonym (or recipients' pseudonyms in case of multicast) are unlinkable.
- *Unobservability*: Unobservability also protects IOIs from being exposed. That is, the message transmission is not discernible from random noise. More specifically, *sender unobservability* means that a could-be sender's transmission is not noticeable. *Recipient unobservability* means that a could-be recipient's transmission is not

noticeable. *Relationship observability* means that it is not noticeable whether anything is sent from a set of could-be senders to a set of could-be recipients.

Throughout this paper, IOI means wireless transmission in mobile ad hoc networks. We use the term "anonymity" as a synonym of "anonymity in terms of unlinkability". In other words, we do not address how to make wireless transmissions indistinguishable from random noises, thus unobservability is not studied in this work. Instead, we address two closely-related unlinkability problems for mobile ad hoc networks.

We study *route anonymity* problem to implement a untraceable routing scheme, where each route consists of a set of hops and each hop is identified by a route pseudonym. For each multi-hop route, we implement relationship anonymity for the corresponding set of route pseudonyms. As a result, all packets of a connection are efficiently forwarded as usual, while the adversaries cannot associate this packet flow to a set of forwarding nodes, or reconstruct any route from a starting point to other points en route. The route pseudonymity approach clearly differentiates our work from earlier studies addressing identity pseudonymity (e.g., person pseudonymity, role pseudonymity, and transaction pseudonymity).

The route pseudonymity approach enables *location privacy* support that realizes unlinkability between location and mobile node's identity. This is achieved by anonymous wireless communications that hide the sender and receiver. This part covers the traditional meaning of sender anonymity, recipient anonymity, and relationship anonymity in a wireless neighborhood.

2.4. Network model

Due to the limited radio propagation range of wireless devices, routes in a mobile ad hoc network are often "multi-hop." Nodes in the network move arbitrarily, thus network topology changes frequently and unpredictably. Moreover, communication, computation, and energy resources on many network nodes are limited.

We assume wireless links are symmetric; that is, if a node X is in transmission range of some node Y, then Y is in transmission range of X. On a wireless link a node's medium access control (MAC) interface is capable of broadcasting data packets locally. [1]. Within its transmission range, a network node can send a unicast packet to a specific node, or a broadcast packet to all local nodes. A node may hide its identity pseudonym using an anonymous broadcast address. In 802.11, a distinguished predefined multicast address of all 1's can be used as source MAC address or destination MAC address to realize anonymity for local senders and receivers. In addition, by anonymous acknowledgment and re-transmission, a local sender and a local receiver can implement locally reliable unicast. If the count of re-transmission exceeds a predefined threshold, the sender considers the connection on the hop is lost.

2.5. Underlying cryptographic tools

MIX-Net A number of protocols for anonymity, Web-MIXes [4], ISDN-MIXes [43], Stop-and-Go-MIXes [24], Onion Routing [46], and many others, have been based on Chaum's anonymous email solution: a network of *MIX*es [7]. The MIX-Net design assumes that a sender can instantly send secret messages to any receiver that can be decrypted by the receiver only, for example, using the receiver's public key in encryption. Suppose a message m needs to be sent from source S to destination D via one MIX M, the input of the MIX-Net should be prepared as

$$\{D, N_S^1, \underline{\{m, N_S^0\}_{PK_D}}\}_{PK_M},$$

so that only M can decrypt the input, throws away the random nonce (proposed in Chaum's original work to stop ciphertext match attack as the network has limited number of nodes and corresponding public keys), knows D is the downstream forwarder, and forwards the protected message to D.

If the message needs to go through a sequence of MIXes $\{M_{n+1}, M_n, \cdots, M_2, M_1\}$, then the MIX-Net's input becomes

$$\{M_n, N_S^n, \{\cdots \underline{\{M_1, N_S^2, \{D, N_S^1, \underline{\{m, N_S^0\}_{PK_D}}\}_{PK_{M_1}}\}_{PK_{M_2}}}\} \cdots\}_{PK_{M_{n+1}}}$$

[1] Here we discuss an 802.11-like wireless MAC scheme. Though non-broadcasting wireless MAC schemes (such as directional antenna technology) are under development, broadcast based MAC continues to be an affordable solution that can be used by all network nodes.

Such a cryptographic data structure is named as "*onion*" in Internet Onion Routing networks [46]. Each MIX en route peels off one layer of the onion, knows the downstream forwarder, then forwards the remaining onion to it. Each forwarding MIX only knows the immediate downstream forwarder, and the immediate upstream forwarder as well (if data forwarding is observable).

Chaum also addressed defense against *timing analysis*, which relies on network delays to expose certain information about routing. A technique called *mixing* can thwart this attack. Such mixing techniques include sending messages in reordered batches, sending dummy messages, and introducing random delays. An idealized MIX-Net protocol should ensure that timing analysis will be effectively stopped.

Trapdoor information Trapdoor is a common concept in cryptographic functions [31]. A *function* $f : X \rightarrow Y$ maps its *domain* X to its *codomain* Y. Each element $x \in X$ is mapped into its *image* $y = f(x) \in Y$. Each element $y \in Y$ may have indefinite number of *preimage* x where $f(x) = y$. A function $f : X \rightarrow Y$ is a *one-way function* if it is "easy" to obtain image for every element $x \in X$, but "computationally infeasible" to find preimages given any element $y \in Y$. A function f is a *trapdoor one-way function* if f is a *one-way function* and it becomes feasible to find preimages for $y \in Y$ given some *trapdoor information*. Without the secret trapdoor keys, it is hard to inverse the cryptographic functions to obtain protected plaintexts or signatures. With the secret trapdoor keys, the cryptographic functions are invertible in polynomial time.

Cryptographic operations incur processing overheads. ANODR minimizes such overheads, and only uses cryptographic trapdoor one-way functions during anonymous route discovery phase. The cryptographic functions are needed to establish route pseudonyms, which in turn efficiently realize local trapdoors without cryptographic operation/overhead.

3. ANODR System Design

3.1. Design challenges

The design of ANODR faces many challenges, largely due to security attacks that can be launched in the practical adversary model and complex network dynamics of mobile ad hoc networks.

Defense against traffic analysis Traffic analysis is a network based attack against distributed systems. By this attack, an external adversary needs not compromise a legitimate node or break relevant cryptographic designs, yet it can trace a packet flow using timing and other critical information.

Defense against passive internal adversary Safety of ground forces is always the greatest concern of any electronic warfare design. Our likely enemies probably are not able to deploy a battle-ready mobile ad hoc network comparable to our armed forces. The electronic warfare in the two gulf wars demonstrates that the enemy's radars and other communication systems are facing immediate elimination once they are detected and targeted. However, during a combat the enemies may be able to capture several legitimate ground nodes. To baffle our intrusion detection systems and followed countermeasures, it is appealing for them to use the captured nodes to launch passive attacks without introducing obvious anomalies into network routing. In particular, they can launch traceability attacks to trace, locate, and then physically destroy our assets without violating ad hoc routing protocols. Such passive internal adversaries will avoid aggressive schemes that can be easily detected by intrusion detection systems, and attempt to be as "invisible" as possible.

Fully distributed security design In an ad hoc network, mobile nodes are autonomous units that are capable of roaming independently. Decision-making in the network is usually not centralized, otherwise the central point becomes the single point of failure and single point of compromise. Consequently, a smart adversary will seek to compromise the central points at first. A fully distributed design is less vulnerable to such attacks, and is more compatible with system dynamics of ad hoc networks.

The impact of expensive processings on ad hoc routing protocols Unnecessary cryptographic operations in wireless broadcast channel may incur excessive overheads. For example, in on-demand routing protocols, the RREQ flooding is a network-wide process. When expensive public key processings meet such expensive on-demand flooding process, routing performance is expected to deteriorate significantly (as demonstrated in Section 6). It is imperative for our design to avoid such ominous combinations.

3.2. Design rationales

Broadcasting with trapdoor assignment As shown in previous research [50], [51], multicasts and broadcasts without specifically identifying the receiver(s) are effective means to achieve recipient anonymity. In this work we extensively explore the mechanism of broadcasting with trapdoor assignment, that is, by embedding a trapdoor information known only to the receiver(s), data can be anonymously delivered to the receiver(s) but not other members in the same receiving group.

Intrusion tolerant location privacy and untraceability design Due to the limited radio propagation range of wireless devices, routes in ad hoc networks are often "multi-hop." Major goals of our design are to ensure location privacy for each forwarding node and to prevent the enemy from effectively tracing a multi-hop route from a starting point to other points en route (especially to the source and to the destination).

However, in hostile environments, intrusion is likely inevitable over a long time window. A distributed protocol vulnerable to single point of compromise is not a proper solution. A qualified solution should maximize its tolerance to multiple compromises, especially against passive internal adversaries who would exhibit no malicious behavior and stay in the system. The number of such passive internal adversaries can be added up over a long time interval. Message encryption is a good solution, but it does not necessarily offer protection if the protected message can be decrypted (due to node intrusion). Instead, we employ a pseudonymity approach where each hop of an ad hoc route is assigned a random pseudonym to be used in data forwarding. With respect to attacks against route pseudonyms, two pseudonyms en route are unlinkable when no node is intruded, and K pseudonyms en route cannot be linked together when less than $K-1$ nodes are intruded.

Dissociating untraceable ad hoc routing from identity pseudonymity and content privacy In our design, untraceable routing in ad hoc networks is orthogonal to identity pseudonymity and content privacy. The route pseudonymity approach allows mobile nodes to transmit their packets anonymously without identifying the sender and the receiver. Network members may also employ end-to-end security protocols (e.g., SSL/TLS, host-to-host IPsec) to ensure privacy of their application payloads. Such protocols provide security services at or above the network layer, and are not the subjects studied in this work.

Avoiding expensive cryptographic operations Cryptographic operations incur processing overheads. Compared to symmetric key operations, public key processing on resource-limited nodes are relatively much more expensive. According to our measurements on low-end mobile devices (Section 6), symmetric key encryption scheme AES/Rijndael can achieve 2.9×10^7bps encryption bit-rate on an iPAQ 3670 pocket PC. Other comparable encryption schemes have similar performance on the same platform. However, common public key cryptosystems require 30–100 milliseconds of computation per encryption or per signature verification, 80–900 milliseconds of computation per decryption or per signature generation. These measurements are consistent with previous results generated by other research groups on similar platforms [6].

Therefore, ANODR avoids using public key cryptosystems if symmetric key cryptosystems can provide the needed support. It also avoids using symmetric key cryptosystems if not indispensable.

3.3. Design components

ANODR divides the routing process into two parts: *anonymous route discovery* and *anonymous route maintenance*. Besides, in *anonymous data forwarding* data packets are routed anonymously from senders to receivers as usual. The details of these parts are described below:

Anonymous route discovery Anonymous route discovery is a critical procedure that establishes random route pseudonyms for an on-demand route. A communication source initiates the route discovery procedure by assembling an RREQ packet and locally broadcasting it. The RREQ packet is of the format

$$\langle RREQ, seqnum, tr_{dest}, onion\rangle,$$

$RREQ$	$seqnum$	tr_{dest}	$onion$
8-bit	160-bit	to be addressed in § 4	≈400-bit

where (i) $seqnum$ is a globally unique sequence number[2]. (ii) tr_{dest} is a cryptographic trapdoor that can only be opened by the destination. Depending on the network's cryptographic assumptions, how to realize the global trapdoor is an implementation-defined cryptographic issue and will be discussed later in the section. (iii) $onion$ is a cryptographic onion that is critical for route pseudonym establishment.

Using cryptographic onion in RREQ network-wide flooding raises design validity concerns as well as performance concerns. We will present three variants to illustrate our design. The first one is a naive porting of MIX-Net to mobile ad hoc networks. The last one features best anonymity guarantee and best performance.

$$\begin{aligned}
\mathbf{PO_A} &= \underline{\{A,src,N_A\}_{PK_A}} \\
\mathbf{PO_B} &= \underline{\{B,A,N_B\,,\underline{\{A,src,N_A\}_{PK_A}}\}_{PK_B}} \\
\mathbf{PO_C} &= \underline{\{C,B,N_C\,,\underline{\{B,A,N_B\,,\underline{\{A,src,N_A\}_{PK_A}}\}_{PK_B}}\}_{PK_C}} \\
\mathbf{PO_D} &= \underline{\{D,C,N_D\,,\underline{\{C,B,N_C\,,\underline{\{B,A,N_B\,,\underline{\{A,src,N_A\}_{PK_A}}\}_{PK_B}}\}_{PK_C}}\}_{PK_D}}
\end{aligned}$$

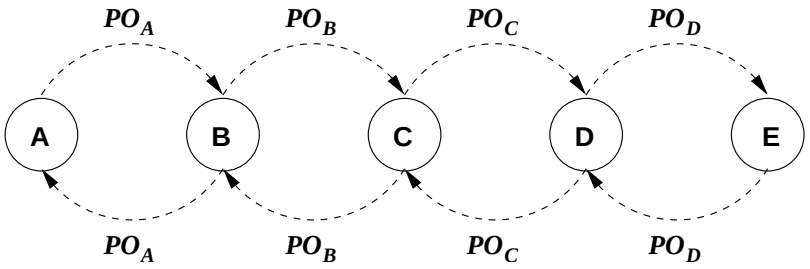

Fig. 2. **ANODR-PO: Anonymous route discovery using public key cryptography** (A single path showed from source A to destination E)

Like MIX-Net, the cryptographic onion used in the first scheme is formed as a public key protected onion (PO). The corresponding ANODR-PO protocol is described below:

1) *RREQ phase*: RREQ packets with previously seen sequence numbers are discarded. Otherwise, as depicted in Figure 2, each RREQ forwarding node X prepends the incoming hop to the PO structure, encrypts the result with its own public key PK_X, then broadcasts the RREQ locally.
2) *RREP phase*: When the destination receives an RREQ packet, the embedded PO structure is a valid onion to establish an anonymous route towards the source. The destination opens the trapdoor and assembles an RREP packet of the format

$$\langle RREP, N, pr_{dest}, onion \rangle$$

$RREP$	N	pr_{dest}	$onion$
8-bit	128-bit	to be addressed in § 4	≈400-bit

where $onion$ is the same cryptographic onion in the received RREQ packet, pr_{dest} is the proof of global trapdoor opening, and N is a locally unique random route pseudonym. The RREP packet is then transmitted by local broadast. Unlike RREQ phase when the ad hoc route is determined, the RREP phase is less time-critical and is implemented by reliable transmissions (The details about proof of global trapdoor opening, anonymous reliable transmission, and uniqueness of local pseudonyms are discussed later in Section 4).
As depicted in Figure 2, any receiving node X decrypts the onion using its own private key SK_X. If its own identity pseudonym X does not match the first field of the decrypted result, it then discards the packet. Otherwise, the node is on the anonymous route. It selects a locally unique nonce N', stores the correspondence between $N \rightleftharpoons N'$ in its forwarding table, peels off one layer of the onion, replaces N with N', then locally broadcasts the modified RREP packet. The same actions will be repeated until the source receives the onion it originally sent out.

[2]There are many methods to implement the globally unique sequence number, for example, applying collision-resistant one way hash functions on node's unique identity pseudonyms can generate statistically unique values [34].

Upon receiving different RREQ packets, the destination can initiate the same RREP procedure to realize multiple anonymous paths between itself and the source. We leave the decision to be made by implementation defined policies.

During the RREP phase, the protocol uses local broadcasts with trapdoor information to improve receivers' anonymity. After the RREP phase, every node on the route only needs its own forwarding table to anonymously route data packets. Node's identity pseudonyms, such as network addresses and cryptographic pseudonyms, are separately maintained without affecting data forwarding.

Firstly, this ANODR-PO scheme has a significant drawback. As RREQ is a network-wide flooding process, large processing overhead will exhaust computation resources at the entire network level. Hence we need to devise an efficient scheme featuring extremely low processing delay during RREQ flooding.

As RREQ and corresponding RREP packets are forwarding through the network like a boomerang, high-speed symmetric key encryption can play an important role in anonymous route discovery. In other words, secret information can be protected by symmetric key encryption in RREQ phase, and can be lately decrypted at the same node in RREP phase. This will minimize the processing latency in route discovery, so that our scheme will have maximal chance to choose the identical path as regular on-demand route discovery protocols.

$$\boldsymbol{BO_A} = \underline{K_A(A, src)}$$

$$\boldsymbol{BO_B} = \underline{K_B(B,A, \underline{K_A(A, src)})}$$

$$\boldsymbol{BO_C} = \underline{K_C(C,B, \underline{K_B(B,A,\underline{K_A(A, src)})})}$$

$$\boldsymbol{BO_D} = \underline{K_D(D,C, \underline{K_C(C,B, \underline{K_B(B,A, \underline{K_A(A, src)})})})}$$

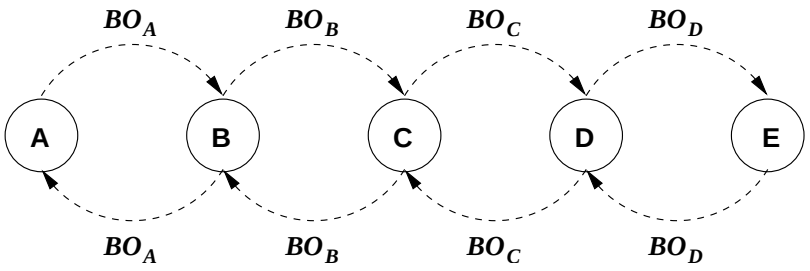

Fig. 3. **ANODR-BO: Anonymous route discovery using Boomerang Onion** (A single path showed from source A to destination E)

The efficient anonymous route discovery protocol is depicted in Figure 3. Instead of relying on public key encrypted onions, the new scheme ANODR-BO uses symmetric key based *Boomerang Onions* (BO).

1) When intermediate forwarding node X sees an RREQ packet, it prepends the incoming hop to the boomerang onion, encrypts the result with a random symmetric key K_X, then broadcasts the RREQ locally.
2) The boomerang onion will be bounced back by the destination. Like the public key version, when node X sees an RREP packet, it strips a layer of the boomerang onion and locally broadcasts the modified RREP packet. Finally the source will receive the boomerang onion it originally sent out.

Compared to ANODR-PO, ANODR-BO ensures that no public key operation is executed during RREQ flooding, hence the impact on processing latency is acceptable because many symmetric key encryption schemes have good performance even on low-end devices.

Secondly, ensuring identity anonymity for ad hoc network members is a critical design goal. We have so far assumed that RREQ and RREP packet senders reveal their identity pseudonyms in wireless transmission. Fortunately, the senders need not to reveal their identity pseudonyms if trapdoor information is appropriately embedded and transmitted. Figure 4 shows the case where anonymous route discovery depends completely on local broadcast with trapdoor information. The depicted ANODR-TBO only uses trapdoor boomerang onions (TBO).

$$\begin{aligned}
\textbf{TBO}_A &= K_A(src) \\
\textbf{TBO}_B &= K_B(N_B\,,\, K_A(src)) \\
\textbf{TBO}_C &= K_C(N_C\,,\, K_B(N_B\,,\, K_A(src))) \\
\textbf{TBO}_D &= K_D(N_D\,,\, K_C(N_C\,,\, K_B(N_B\,,\, K_A(src))))
\end{aligned}$$

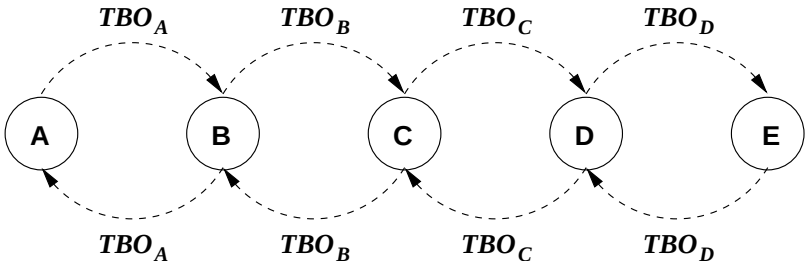

Fig. 4. **ANODR-TBO: Anonymous route discovery using Trapdoor Boomerang Onion** (A single path showed from source A to destination E)

1) When intermediate forwarding node X sees an RREQ packet, it embeds a random nonce N_X to the boomerang onion (this nonce is not a route pseudonym nonce), encrypts the result with a random symmetric key K_X, then broadcasts the RREQ locally. The trapdoor information consists of N_X and K_X, and is only known to X.
2) The boomerang onion will be bounced back by the destination. After each local RREP broadcast, only the next hop (i.e., the previous hop in RREQ phase) can correctly open the trapdoor it made in the RREQ phase, hence the result is equivalent to a wireless unicast. Then the node strips a layer of the boomerang onion and locally broadcasts the modified RREP packet.

Anonymous data forwarding For each end-to-end connection, the source wraps its data packets using the outgoing route pseudonym in its forwarding table. A data packet is then broadcast locally without identifying the sender and the local receiver. The sender does not bother to react to the packet it just sent out. All other local receiving nodes must look up the route pseudonym in their forwarding tables. The node discards the packet if no match is returned. Otherwise, it changes the route pseudonym to the matched outgoing pseudonym, then broadcasts the changed data packet locally. The procedure is then repeated until the data packet arrives at the destination.

Anonymous route maintenance Following the soft state design, the routing table entries are recycled upon timeout T_{win}. Moreover, when one or more hop is broken due to mobility or node failures, nodes cannot forward packet via the broken hops. We assume nodes can detect such anomalies when re-transmission count exceeds a predefined threshold. Upon anomaly detection, a node looks up the corresponding entry in its forwarding table, finds the other route pseudonym N' which is associated with the pseudonym N of the broken hop, and assembles a route error packet of the format $\langle RERR, N' \rangle$. The node then recycles the table entry and locally broadcasts the RERR packet. If multiple routes are using the broken hop, then each of them will be processed and multiple RERR packets are broadcast locally.

A receiving node of the RERR packet looks up N' in its forwarding table. If the lookup returns result, then the node is on the broken route. It should find the matched N'' and follow the same procedure to notify its neighbors.

4. DISCUSSIONS

Unlinkable onions, pseudonyms, and payloads Given an input onion I, the symmetric key encryption function used in onion production ensures that cryptanalysts cannot know the relation between the input onion I and output onion O with non-negligible probability. Only the forwarding producer knows that it produces O from I—and by cryptanalysis it is hard for any other node to discover the relation. Hence it is also hard for cryptanalysts to correlate the route pseudonyms established on top of the cryptographic onions.

However, an unbounded eavesdropper can trace on-demand routes by exploring other data fields in RREQ/RREP packets: (i) RREP packets with the same pr_{dest} field are likely on the same route. (2) RREQ packets with the same $\langle seqnum, tr_{dest}\rangle$ fields belong to the same route. The unbounded eavesdropper can record all onions during RREQ phase, then the RREP packets using the onions from previously matched RREQ packets belong to the same route.

To resist the unbounded eavesdropper, an asymmetric secret channel is needed from an RREP sender to its receiver. During the RREQ phase, a forwarding node must embed its one-time public key from a public/private key pair (pk_{one}, sk_{one}). RREQ packet format is changed to be

$$\langle RREQ, pk_{one}, seqnum, tr_{dest}, TBO\rangle,$$

$RREQ$	pk_{one}	$seqnum$	tr_{dest}	TBO
8-bit	≈160-bit	160-bit	to be discussed right below	≈400-bit

and RREP packet format is changed to be

$$\langle RREP, \{K_{seed}\}_{pk_{one}}, K_{seed}(pr_{dest}, TBO)\rangle,$$

$RREP$	$\{K_{seed}\}_{pk_{one}}$	$K_{seed}(pr_{dest}, TBO)$
8-bit	≈160-bit	(to be discussed right below) + (≈400-bit onion)

where K_{seed} is a nonce (same as N in the § 3.3 original design). During the RREP phase, the producer of an onion can secretly recover the needed information as usual. For the RREQ one-time key, storage overhead can be traded off for key generation overhead as the node may generate a number of such key pairs prior to joining in the ad hoc network. In addition, the key length should be minimized to reduce transmission overhead, but must be long enough to resist cryptanalysis. ANODR recommends elliptic curve based schemes, such as ECAES, with key length ranging from 112-bit to 160-bit (approximately equivalent to RSA using 512-bit to 1280-bit key length [28]) to resist a 1-day cryptanalysis with hardware cost ranging from $50,000,000 to $250,000,000.

The revised design ensures that there is no expensive public key computation incurred during RREQ flooding, so that the chance of finding identical on-demand routes as the original design is magnified. During the RREP phase, each forwarding node en route must do one encryption and one decryption using a well-known public key scheme. Fortunately, the tradeoff can realize more appealing features.

The first benefit is self-synchronized route pseudonym update. Consider a single hop on an anonymous route, the two nodes at both ends will share a route pseudonym in their forwarding tables. One is an outgoing entry, and the other is an incoming entry. As long as these two entries are appropriately synchronized, the pseudonym can be constantly changed to other random but locally unique values. If previous hops and next hops have the same behavior, the packet flow of the same connection will be marked by "one-time" route pseudonyms changed over time and over hops from the source to the destination.

Route pseudonym update explores the concept of *unpredictability in polynomial time*. This concept means that no Turing-complete algorithm is able to differentiate a cryptographically strong pseudorandom sequence from a truly random sequence in polynomial time. The pioneer work done by Yao[54], Blum, and Micali[5] illustrates the relation between one-way functions and pseudorandom number generators. They showed that cryptographically strong pseudorandom bit generators realized on top of one-way functions can pass next-bit-test. Thus any polynomial time statistical test cannot distinguish the next pseudorandomly generated bit from a truly random bit.

Slow but provably secure pseudorandom bit sequences can be constructed using hardcore predicates of a one-way function. In particular, as the hardcore predicate for any one-way function have been discovered, cryptographically strong pseudorandom generators are constructible from any one-way function [16][17]. However, due to performance concerns, many implementations use (relatively) fast one-way functions (e.g., MD5,SHA1,AES) to generate pseudorandom block sequences instead of bit sequences.

In ANODR, route pseudonym sequence is generated by feeding the shared secret seed K_{seed} into the fast one-way function f, then feeding the output back to the input repetitively. In other words, the i-th pseudonym is

$$n_i = \underbrace{f(f(\cdots f}_{i}(K_{seed})\cdots)) = f^i(K_{seed}).$$

The two ends of a hop should update the shared route pseudonym per forwarding packet for a reliable transmission. For a unreliable transmission, at least two candidate schemes are useful: (1) If tight time synchronization is feasible, that is, difference between the two system clocks is smaller than the delay to transmit the smallest packet on their

network interface, then both ends can agree to update the route pseudonym per short interval t_{int}; (2) The sender stamps a non-decreasing sequence number seq on each packet payload. The receiver computes $n_{seq} = f^{seq}(K_{seed})$ based on current pseudonym. The values for seq are not necessarily consecutive. If the difference between two consecutively received sequence numbers is reasonably small, experiments on TESLA protocol[41] have shown that the computational overhead is acceptable.

The second benefit is packet payload shuffle. In an ad hoc network the adversary can simply match data payloads to trace a specific packet (if his collaborators are on the forwarding path, or his mobility speed can catch up with the packet forwarding process). In 802.11, the shared secret can be used as WEP key to implement link payload encryption per hop and foils the attack which is not against route pseudonyms but data payloads. The purpose of such link payload shuffle is to foil "matching-payload-attack" rather than to ensure content privacy. On some 802.11 hardware, e.g., those based on PRISM chipset, the WEP payload shuffle can be accomplished by the hardware and does not consume CPU cycles.

Setting and opening global trapdoor As we stated previously, design of global trapdoor is an implementation-defined issue that heavily depends on other cryptographic assumptions of the network. For example, as assumed in Ariadne [20], if the source shares the destination's TESLA secret key K_T, then the global trapdoor tr_{dest} is the anonymous assignment $K_T(dest, K_c)$ where $dest$ is the special destination tag and K_c is a nonce. The probability of revealing $dest$ from $K_T(dest, K_c)$ is negligible without knowing the key K_T. Trying to open the global trapdoor incurs another decryption overhead at each node, but the RREQ communication latency from the source to the destination does not increase as each forwarding node can try to open the trapdoor after forwarding the RREQ packet.

Under the exemplary assumptions, RREQ format is instantiated as

$$\langle RREQ, pk_{one}, seqnum, K_T(dest, K_c), K_c(dest), TBO\rangle,$$

$RREQ$	pk_{one}	$seqnum$	$K_T(dest, K_c)$	$K_c(dest)$	TBO
8-bit	≈160-bit	160-bit	256-bit	128-bit	≈400-bit

where K_T is destination's TESLA secret, $dest$ is an 128-bit special tag denoting destination, and K_c is an 128-bit commitment key. Consequently RREP format is instantiated as

$$\langle RREP, \{K_{seed}\}_{pk_{one}}, K_{seed}(K_c', TBO)\rangle,$$

$RREP$	$\{K_{seed}\}_{pk_{one}}$	$K_{seed}(K_c', TBO)$
8-bit	≈160-bit	(128-bit proof) + (≈400-bit onion)

where $seqnum$ is the one from corresponding RREQ packet, K_c' is the anonymous proof presented by the destination. Any forwarding node can verify the anonymous proof of trapdoor opening by checking $K_c(dest) \stackrel{?}{=} K_c'(dest)$.

Here ANODR explores the concept of "*trapdoor commitment*". One-way functions are collision resistant—given a message digest $K_c(dest)$, it is computationally hard to find the preimage of the digest, or another preimage collision that can produce the same digest. Thus the one who can present the preimage K_c is the one who was committed. TESLA protocol also explores the same concept and has been used in ad hoc networks to provide data origin authentication service [20].

Reliability of local broadcasts In RREP/RERR packet transmission and also in reliable data communication, local broadcasts must be reliably delivered to the intended receiver despite wireless interference. This can be achieved by anonymous acknowledgments. Once the receiver has opened the trapdoor and anonymously received the data, it should locally broadcast an anonymous ACK packet. In an anonymous ACK packet, the source or destination MAC address is the predefined all-1's broadcast address. The packet payload uniquely determine which packet is being acknowledged. In particular, route pseudonyms can be embedded in the ACK's payload to acknowledge an RREP/RERR packet or application data packet.

At the other end of the hop, the sender must try to re-transmit data packets until it receives the anonymous acknowledgment. Like 802.11's reliable unicasts, if retransmission count exceeds a predefined threshold, then the node considers the hop connection is broken. If this happens during application data forwarding, route maintenance

will be initiated to refresh forwarding table entries.

Route pseudonym collision In the ideal case, no route pseudonym collision is allowed within any forwarder's single hop neighborhood. Here we study how to enforce the constraint.

As the chance of collision p_c decreases exponentially when pseudonym length l increases linearly (currently we select the route pseudonym length $l = 128$ bits), random selection following uniform distribution inside the pseudonym space is computationally collision resistant. For arbitrarily k randomly selected local pseudonyms, the chance of collision p_c is only

$$p_c = 1 - \frac{\prod_{i=0}^{k-1}(2^l - i)}{(2^l)^k}$$

When pseudonym collisions happen, packets will be duplicated and erroneously forwarded to other destinations. Currently we address this problem by adding keyed end-to-end packet checksum. HMAC [26] functions are keyed collision resistant hash functions widely used in message digesting. Like SSL/TLS, for each connection an initial handshake establishes a shared secret key between the message sender and receiver. Without the secret key, it is computationally infeasible to generate a correct packet checksum. As different connections have different keys, an incorrectly forwarded packet will finally be discarded at the destination.

Our study shows that the packet checksum method may not be necessary if we increase the route pseudonym bit-length l. Any checksum, including the one used in TCP or UDP, is only computationally sound rather than perfect. In other words, there is negligible but greater than zero probability that a checksum-protected packet is indeed corrupted but undetectable. For example, by using 128-bit MD5 as the function to create cryptographic checksum, the probability of such detection failure is about 1 per $2^{128/2} = 2^{64}$ packets due to "birthday paradox" [31]. This probability is much higher than p_c as we currently choose l to be 128-bit.

Routing optimizations One limitation of ANODR is the sensitivity to terminal node mobility. As nodes move, the path is broken and must be reestablished. The well-known AODV and DSR "repair" strategies (which typically benefits from routes cached during unrelated path establishments) cannot be applied here since only anonymous paths specifically set up for the current connection can be used, or the optimization technique by the design conflicts with the anonymity goals.

To enhance performance in a mobile environment, and in particular to mitigate the disruption caused by path breakage, we encourage actual implementations to use multiple paths discussed in the anonymous route discovery part. Several multi-path routing techniques have been described and evaluated in the ad hoc routing literature[38][27][29][36]. Several paths can thus be computed and are used in a round robin schedule. If the application runs on TCP, a TCP protocol resilient to out-of-sequence must be used. Sequential path computation has the advantage of allowing online maintenance—if a path fails, a new path is computed while the remaining paths are still in use.

5. Untraceability Analysis and Comparisons

In order to unlink a network member's identity and its standing location, ANODR employs a very different approach from common on-demand routing protocols [22], [37], [39]. As depicted in Figure 5, common on-demand routing protocols use node's identity pseudonyms to furnish packet forwarding, while ANODR uses an on-demand route discovery process to randomly name each transmission hop and to record the mapping between consecutive hops in each forwarding node. ANODR's anonymous routes bear resemblance to virtual circuits used in Internet QoS [2]. However, the design goal of ANODR is completely different from virtual circuits: When node intrusion occurs in hostile environments, the damage is localized in ANODR, but not in other on-demand protocols.

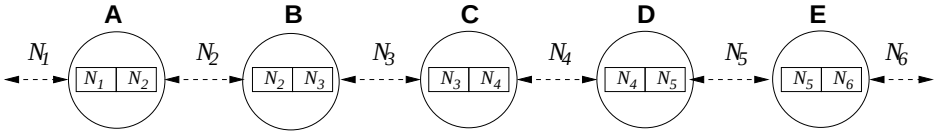

Fig. 5. **Different approaches in packet forwarding** (Using node pseudonyms $A, B, \ldots$ vs. using route pseudonyms $N_1, N_2, \ldots$)

5.1. Intruders and route traceable ratio

If a node X is compromised, the adversaries can link two random pseudonyms together for each route going through the node X. For each route, if F forwarding nodes are compromised and they are consecutive en route, then a route segment of $F+1$ hops are linked together. If the compromised nodes are not consecutive en route, then the adversary can form multiple route segments, but it is hard to link together the multiple compromised segments. For example, if A is the source and E is the destination in Figure 5, and A, B, D, E are intruded, then adversaries can form traceable segments $\overline{ABC}$ and $\overline{CDE}$, but they have to intrude C to discover that $\overline{ABC}$ and $\overline{CDE}$ belong to the same route.

Let's quantify the damage caused by node intrusion. Suppose the route totally has L hops, K compromised route segments, and the hop count of i-th compromised segment is F_i,$1 \leq \mathrm{i} \leq \mathrm{K}$, we define the *traceable ratio* R of the route as

$$R = \frac{\sum_{i=1}^{K}(F_i \cdot W_i)}{L} = \frac{\sum_{i=1}^{K}(F_i \cdot \frac{F_i}{L})}{L}$$

where W_i is a weight factor. Without loss of generality[3], we select $W_i = \frac{F_i}{L}$ so that the traceable ratio of a route is 100% when all forwarding nodes en route are intruded, or 0 when no forwarding node en route is intruded. In addition, the longer a compromised segment is, the larger the traceable ratio R is as the adversary can trace a longer distance towards its target. Using the same example from the previous paragraph, $L = 4$. The traceable ratio $R = \frac{2 \cdot \frac{2}{4} + 2 \cdot \frac{2}{4}}{4} = \frac{1}{2}$ when A, B, D, E are intruded, or $R = \frac{3 \cdot \frac{3}{4} + 1 \cdot \frac{1}{4}}{4} = \frac{5}{8}$ when A, B, C, E are intruded.

5.2. Eavesdroppers and traffic analysis

In Internet anonymous routing schemes, it is demanded to resist strong attacks such as flooding attack (aka. node flushing attack, $n-1$ attack) and timing analysis [45]. Both attacks require a network-wide monitoring mechanism to trace a set of indeterministic points that the victim message may be routed through. (i) In timing analysis, data transmission is assumed to be observable, and the adversary can monitor network-wide transmission events with timing information recorded. The adversary can use temporal dependency between transmissions to trace a victim message's forwarding path. Each node can use mixing technique to thwart timing analysis. That is, it uses a playout buffer to store and re-order received data packets, and to inject dummy packets into the buffer if necessary. Then it flushes the buffer at the end of a playout time window. (ii) In flooding attack, the adversary can send $n-1$ messages to trace a victim message even though each MIX node using a playout buffer of size n. The adversary can match its own attacking messages and differentiates the victim message.

In ANODR, flooding attack is effectively stopped by hop-based payload shuffle. To foil timing analysis, ANODR uses similar methods proposed in various MIX-Net designs [43][24][4]. Let's assume node X chooses t_X as its playout time window size and r_X as its playout buffer size. During t_X period, if X has received r data packets with distinct pseudonyms, then it generates $d = r_X - r$ random dummy packets ($d = 0$ if $r = 0$ or $r_X \leq r$). The random pseudonyms used in the dummy packets should be out of the synchronization with any pseudonym sequence in use. At the end of time window t_X, the node X randomly re-orders the r_X packets and sends them out in batch.

Unlike a wired link, wireless medium is shared by all local nodes. Thus r is the number of all packets received during t_X, including those packets not intended for the node. Nevertheless, the mixing process may potentially generate many dummy packets that consume significant communication and energy resources, thus it is allowed to trade untraceability with performance. The node X may shrink the size of its playout time window, or generates less dummy packets to decrease the overhead, but the price is that the protocol is more traceable.

Let's estimate the effectiveness of traceability attack on a multi-hop on-demand route. Assume in a locality an adversary records that r route pseudonyms have been used during a time interval T_{attack}, all of the r pseudonyms are unique if the one-way function returns collision-free results, thus the adversary has to guess the relation between two pseudonyms by testing all $\binom{r}{2}$ cases. The probability of a correct guess is $p_g = 1/\binom{r}{2}$. If the route being traced has h remaining hops towards the adversary's target, then the probability of a successful trace is less than a

[3]The weight W_i can be of form $(\frac{F_i}{L})^r$ where $r \geq 0$.

upperbound[4] p_g^h, which is a number rapidly approaching zero when h or r increases. The goal of sending dummy packets is to maintain a large enough r in the neighborhood where a real transmission occurs.

In addition to data packets, RREP and RERR packets are also threatened by timing analysis. Similarly, each node can send dummy RREP and RERR packets to confuse the eavesdroppers. A dummy RREP packet uses a random dummy pk_{one} in encryption so that nobody can decrypt it. A dummy RERR packet uses a random pseudonym that is out-of-synchronization of any pseudonym sequence in use on the node.

5.3. Comparison with DSR and AODV

DSR [22] is traceable by a single eavesdropper en route since it explicitly embeds routing information in packet headers. For any DSR route, the identities of all forwarding nodes and the relation among all forwarding hops are recoverable from a single intercepted packet. AODV [39] is more untraceable because routing information is stored in routing tables instead of packet headers. Nevertheless, it is traceable by collaborative eavesdroppers and does not provide location privacy support.

ANODR is much more robust against anonymity and traceability attacks than DSR and AODV:

- In standard DSR and AODV, all routing information are open to public. External adversaries can trivially compromise node's mobile privacy—an eavesdropper can successfully detect the identities of all local transmitters. The eavesdropper also knows that these identities are currently in the area bounded by its signal receiving range.
  In contrast, the locally unique route pseudonyms allow ANODR nodes to transmit their packets anonymously without identifying the sender and the receiver. Though the adversary can detect the existence of wireless transmissions, it is hard to discover the identities of local transmitters.
- As DSR embeds all forwarders' identities in its packet header, a DSR route is immediately visualized if one data packet is intercepted. An AODV route is traceable if multiple collaborative eavesdroppers en route combine their eavesdropped data and analyze the forwarding chain (e.g. checking the chain of senders and receivers in link layer packet headers or simply doing "matching-payload-attack"). In other words, if a region is covered by multiple collaborative eavesdroppers, then they can visualize all AODV paths intersected with the region. In our adversary model an omnipresent eavesdropper is assumed, thus all AODV routes can be visualized.
  In contrast, ANODR separates routing from node's identity pseudonyms. To visualize an on-demand route, it is necessary to link two route pseudonyms together. However, cryptographically strong pseudorandom sequence generation ensures that pseudonyms used on the same hop are unlinkable in polynomial time by any Turing-complete algorithm. To link two pseudonyms on consecutive hops, the adversary has to do timing analysis or to intrude the forwarding node. Mixing techniques can resist the former attack, and physical protections can resist the latter attack.
- Node intrusion is a common attack against mobile nodes deployed in hostile environments. In DSR, an intruder can visualize every cached on-demand route. In AODV, the intruder knows the compromised node is en route to each cached destination and how far the destination is. We consider these vulnerabilities are not apposite to untraceable routing schemes. In ANODR only the mapping between two random sets of route pseudonyms is exposed.

5.4. Comparison to encryption based proposals: Why not simply encrypting routing information?

It is feasible to provide untraceability support to DSR and AODV using methods other than ANODR. Basagni et al. [3] use a network-wide symmetric key to secure routing information. The proposed solution effectively stops eavesdroppers, but it has to address the problem of single node intrusion. The authors argue to protect the key using tamper resistance facilities which introduce physical cost and offer indefinite physical warranty. Another possible answer is to change the network-wide key to hop-based link encryption keys, then a node intrusion would only

[4]The upperbound p_g^h is not achievable if the adversary fails to physically move in the same direction of the packet flow. It is beyond the paper's scope to maximize the distance the adversary has to roam.

compromise the routes going through the node. However, it is an open question to establish a web of hop-based link encryption keys in an ad hoc network.

We do not use such encryption-based schemes to protect routing information due to following reasons:

1) For each data packet forwarding with encrypted route headers, one encryption/transmission causes multiple receptions/decryptions at all local neighbors in a wireless broadcast environment. The computational cost of data packet forwarding is potentially very high. In addition, adversaries may simply inject random messages to consume legitimate node's resource. Like regular data packet forwarding, one such attacking packet provokes multiple decryptions for all local victims. The situation favors the adversaries rather than the legitimate nodes.
2) As an encryption function is a one-way function with trapdoor keys, an encryption proposal also follows a trapdoor approach where only nodes knowing the corresponding decryption trapdoor keys can see the plaintext, hence such an encryption proposal is in general equivalent to an ANODR variant with encrypted route pseudonyms. In ANODR, route pseudonyms are low-cost non-cryptographic trapdoors. The pseudorandom pseudonym update is actually an efficient encryption operation. The encryption overhead on 128-bit data only applies to the two communicating nodes, while other wireless nodes pay little cost doing fast table lookup.
3) When node intrusion is possible, it is a non-trivial issue to minimize the subsequent damages. Even after an ideal hop-based link encryption scheme is realized in the future to protect all routing information, a DSR route is traceable by a single intruder en route, while AODV is more vulnerable than ANODR in terms of traceability attacks. For example, collaborative intruders that locate at every other forwarding node can recover an AODV route segment (e.g., A and C in Figure 5 can recover route segment ABC by matching A's downstream forwarder with C's upstream forwarder. This is not possible in ANODR). In fact, in ad hoc routing, nodes rely on their neighbors in data forwarding—this constraint makes many ad hoc routing protocols reveal ad hoc nodes' unique identity pseudonyms to their potential forwarders, thus each passive internal adversary can effectively collect node information within its radio transmission/receiving range. At the physical layer, each internal adversary can increase transmission/receiving power to enlarge its monitoring cell. At the data link layer and network layer, a set of passive internal adversaries can treat the network as a chessboard (namely "chessboard attack"), then adversaries positioning only in black grids (or only in white grids) can effectively visualize all AODV routes. In addition, if a few of these passive internal adversaries are capable of roaming, they can effectively visualize the entire network topology by exchanging hello information with legitimate nodes. ANODR is immuned from such passive attacks that exhibit no malicious behavior against routing protocols.

6. Implementation and Evaluation

6.1. Cryptographic implementation

The processing overhead used in our simulation is based on actual measurement on a low-end device. Table II shows the performance of different cryptosystems. For public key cryptosystems, the table shows processing latency per operation. For symmetric key cryptosystems (the five AES final candidates), the table shows encryption/decryption bit-rate.

TABLE II

Processing overhead of various cryptosystems (on iPAQ3670 pocket PC with Intel StrongARM 206MHz CPU)

Cryptosystem	decryption	encryption
ECAES (160-bit key)	42ms	160ms
RSA (1024-bit key)	900ms	30ms
El Gamal (1024-bit key)	80ms	100ms
AES/Rijndael (128-bit key & block)	29.2Mbps	29.1Mbps
RC6 (128-bit key & block)	53.8Mbps	49.2Mbps
Mars (128-bit key & block)	36.8Mbps	36.8Mbps
Serpent (128-bit key & block)	15.2Mbps	17.2Mbps
TwoFish (128-bit key & block)	30.9Mbps	30.8Mbps

In our cryptographic implementation, the length of $src, dest$ tags and route pseudonym (i.e., K_{seed}) nonces is 128-bit. And the length of other nonces is 40-bit. In RREQ packet, the sequence number $seqnum$ is formed by appending 32-bit timestamp to the source's identity pseudonym (e.g., 128-bit IPv6 address), then applying 160-bit SHA1 HMAC function to the concatenation. In RREQ and RREP packets, the onion is padded with random bits to hide its actual length. Currently we pad the initial onion to be around 400-bit (400 ± 100-bit) because each extra hop extends the actual length of an onion with a 40-bit nonce, and 10-hop is considered a reasonably big hop count in related research [23]. In practice, the number 400 can be replaced by a number based on the estimation of the hop count of the network's diameter.

Besides the 400-bit lower-bound, the PBOC (Probabilistic Boomerang Onion Compression) algorithm can be used to derive algorithms that control the upperbound of cryptographic onions, so that the bit length of cryptographic onions would not expand to arbitrary length.

Algorithm 1 PBOC: Probabilistic Boomerang Onion Compression: RREQ phase

Require: (i) An input onion I received by the current node X. (ii) A pre-calculated probability p based on node X's computing power. Currently we choose $p = \frac{F_X}{F_{max}}$ where $F_{max} = 3GHz$ is the fastest clock frequency available on current Intel Pentium 4 CPUs, and F_X is the clock frequency of node X's CPU.

Query the probabilistic module that returns 1 with probability p.
if the query returns 1 **then**
  Compress the input onion I. Let I be the compressed onion.
  Set a flag bit to 1.
else
  Set the flag bit to 0.
end if
Prepend the flag bit to I.
Produce the output onion O, i.e., prepend a nonce N_X to I, then encrypts the concatenation with a nonce key K_X.
Store the association $O, (N_X, K_X)$ into a retrievable stroage with O as the identifier.

Ensure: Node X produces an output onion O that can be identified, decrypted, and decompressed at RREP phase.

Algorithm 2 PBOC: Probabilistic Boomerang Onion Compression: RREP phase

Require: An input onion O received by the current node X.

Decrypt the onion O with the corresponding K_X associated with O.
if the decrypted nonce matches N_X **then**
  if the first bit following N_X is 0 **then**
    Let I be the remaining bits.
  else
    Decompress the remaining bits. Let I be the result of decompression.
  end if
else
  Not my onion, ignore the RREP packet.
end if

Ensure: Node X recovers the input onion I.

Unfortunately, we know that encryption results produced by "good" one-way functions are indistinguishable from truly random results by any Turing-complete algorithms in polynomial time. The entropy of such encryption results is equal to a truly random results, thus a compression on encrypted data does not work. Fortunately, the design shown in Algorithm 1 and 2 can derive a similar scheme (Algorithm 3 and 4) that trades off storage overhead with cryptographic onion's bit-length (i.e., the communication overhead needed to transmit the onion). Thus onions' bit length and needed communication overhead are tractable.

Algorithm 3 PBOS: Probabilistic Boomerang Onion Substitution: RREQ phase

Require: (i) An input onion I received by the current node X. (ii) A pre-calculated probability p based on node X's storage capacity. Currently we choose $p = \frac{S_X}{S_{max}}$ where S_{max} is a reasonably large upperbound of stroage capacity, and S_X is the storage capacity of node X.

Query the probabilistic module that returns 1 with probability p.
if the query returns 1 **then**
  Produce a random chunk of data with the same bit length of I.
  Set the random data as I, and set a flag bit to 1.
else
  Set the flag bit to 0.
end if
Prepend the flag bit to I.
Produce the output onion O, i.e., prepend a nonce N_X to I, then encrypts the concatenation with a nonce key K_X.
if the flag bit is 1 **then**
  Store the association $O, (N_X, K_X), I$ into a retrievable stroage with O as the identifier.
else
  Store the association $O, (N_X, K_X)$ into a retrievable stroage with O as the identifier.
end if

Ensure: Node X produces an output onion O that can be identified and decrypted at RREP phase. The corresponding I is retrievable.

Algorithm 4 PBOS: Probabilistic Boomerang Onion Substitution: RREP phase

Require: An input onion O received by the current node X.

Decrypt the onion O with the corresponding K_X associated with $seqnum$.
if the decrypted nonce matches N_X **then**
  if the first bit following N_X is 0 **then**
    Let I be the remaining bits.
  else
    Retrieve the stored I.
  end if
else
  Not my onion, ignore the RREP packet.
end if

Ensure: Node X recovers the input onion I.

If storage capacity is not a concern (since the onions are measured in hundreds of bits, i.e., tens of bytes, while storage capacity on even low-end devices is measured in megabytes), then we can let $p = 1$ and enforce PBOS algorithm on all ad hoc nodes all the time, hence the bit-length of cryptographic onions is always the sum of 40-bit nonce N_X and the bit length of the initial source onion. Now the adversary cannot tell the hop count from the onion length, thus we can decrease the initial onion length to be 128-bit (which can produce statistically unique random values as we proved previously). The communication overhead for transmitting a cryptographic onion is no longer ≈400-bit, but only 168-bit. In the followed simulation we use 400-bit as bit-length of cryptographic onions, so that the simulation results are applicable to both PBOS-enabled and PBOS-disabled designs.

6.2. Evaluation

We implement ANODR in simulation as a basic on-demand route discovery/maintenance scheme with flavors of both source routing and table driven. The source routing part is adopted to simulate the appending and pealing

off layers in RREQs and RREPs, a way that is similar to the creation and transmission of RREQs and RREPs in DSR. The table driven part is used to establish the per hop pseudonym switching during RREP propagation and data forwarding, a way that is similar to the routing table maintenance in AODV. Possible optimizations used for AODV and DSR are not used in our implementation, for example, no expanding ring search, no local route repair, no promiscuous listening, no salvaging, no gratuitous route repair, no aggressive caching and no switching entry reuse at intermediate nodes. In addition, ANODR also implements larger RREQ, RREP, and RERR packets with extra processing overhead for encryption and decryption at each packet stop.

We evaluate our proposed routing schemes in three aspects. First, we investigate untraceability of ANODR in terms of intrusion tolerance. As ANODR uses a way similar to source routing in establishing a route and table switching for next hop, we compare ANODR to DSR and AODV. For ANODR, a node intrusion unconditionally exposes everything cached on the node including the mapping between two sets of random route pseudonyms. For DSR, we assume it is protected by an ideal hop-based link encryption scheme. Nevertheless, the entire DSR route will be exposed as long as a packet passing through a compromised node. For AODV, even with the link encryption protection, a compromised node knows the previous hop and the next hop on a path. The difference between the information leaked by AODV and ANODR is that for two two-hop-apart compromised nodes, if AODV is running, they can connected the two path segmentations by exchanging the separately detected information, while this is impossible when ANODR is running. We use *traceable ratio* R (Section 5) to quantify the effect of node intrusions. The traceable ratio for a DSR route is 0 when none of the nodes en route is intruded, or is 1 otherwise. And traceable ration for AODV is calculated using the way mentioned above.

Then we evaluate the performance of ANODR-TBO proposed in both Section 3.3 and 4 in a mobile ad hoc network scenario. The former one provides location privacy support, but is vulnerable to route traceability attacks. They are denoted as "ANODR-TBO (Sec 3)" and "ANODR-TBO (Sec 4)", respectively. Computational delay using symmetric key cryptosystem AES/Rijndael (approximately 0.02ms for each onion construction) is added to each RREQ and RREP forwarding stop. For "ANODR-TBO (Sec 4)", additional key processing time for RREP packets ($42 + 160 = 202$ms) is added according to our measurement. For a comparison, ANODR-PO using the same ECAES public key cryptography and AODV with route optimization are also presented in simulation.

Finally, we evaluate the impact of mixing technique on ANODR performance. We study both mixing overhead and routing performance given many combinations of mixing playout window sizes and playout buffer sizes. In the experiment, the dummy packet size is a random value computed from the average size of data packets recently received.

Metrics we used for routing performance include: (i)*Packet delivery fraction* – the ratio between the number of data packets received and those originated by the sources. (ii) *Average end-to-end data packet latency* – the time from when the source generates the data packet to when the destination receives it. This includes: route acquisition latency, processing delays at various layers of each node, queueing at the interface queue, retransmission delays at the MAC, propagation and transfer times. (iii) *Average data path length* – the average hops that a data packet traveled. (iv) *Normalized control packet overhead* – the number of routing control **packets** transmitted by a node normalized by number of delivered data packets, averaging over all the nodes. Each hop-wise transmission of a routing packet is counted as one transmission. (v) *Normalized control byte overhead* – the total **bytes** of routing control packets transmitted by a node normalized by delivered data bytes, averaging over all the nodes. Each hop-wise transmission of a routing packet is counted as one transmission. This metric is useful in evaluating the extra padding overhead of ANODR. (vi) *Average route acquisition latency* – the average latency for discovering a route, i.e., the time elapsed between the first transmission of a route request and the first reception of the corresponding reply. (vii) *Dummy packet ratio* – the ratio between the number of dummy data packets and real data packets given a specific playout time window and buffer size.

6.3. Simulation Model

The routing protocols are implemented within QualNet^{TM} [49], a packet level simulator for wireless and wired networks, developed by Scalable Network Technologies Inc. The distributed coordination function (DCF) of IEEE 802.11 is used as the MAC layer in our experiments. It uses Request-To-Send (RTS) and Clear-To-Send (CTS)

control packets to provide virtual carrier sensing for unicast data packets to overcome the well-known hidden terminal problem. Each data transmission is followed by an ACK. Broadcast data packets are sent using CSMA/CA only. The radio uses the *two-ray ground reflection* propagation model and has characteristics similar to a commercial radio interface (e.g., Lucent's WaveLAN). The channel capacity is 2 Mbits/sec.

In order to hide the sender's and receiver's identity, ANODR's local broadcast with trapdoor uses broadcast address rather than source and destination's link layer addresses. This behavior makes ANODR's transmission look like 802.11 broadcast. However, ANODR's local broadcast with trapdoor is an equivalence of 802.11's unicast rather than broadcast, except that 802.11 uses traceable identity pseudonyms while ANODR uses untraceable trapdoors (with simple table lookup). In data forwarding we use 802.11 unicast plus 1μs table lookup delay to simulate ANODR's local broadcast with trapdoor. We believe it is practical to implement the same feature in commercial 802.11 device drivers.

The network field is 1500m$\times$300m with 50 nodes initially uniformly distributed. The transmission range is 250m. *Random Waypoint* mobility model [22] is used to simulate nodes' motion behavior. According to the model, a node travels to a random chosen location in a certain speed and stays for a whole before going to another random location. In our simulation, mobility speed varies from 0 to 10 m/sec, and the pause time is fixed to 30 seconds. CBR sessions are used to generate network data traffic. For each session, data packets of 512 bytes are generated in a rate of 4 packets per second. The source-destination pairs are chosen randomly from all the nodes. During 15 minutes simulation time, a constant, continuously renewed load of 5 short-lived pairs is maintained. The simulations are conducted in identical network scenarios (mobility, communication traffic) and routing configurations across all the schemes. Results are averaged over multiple runs with different seeds for the random number generator.

6.4. Simulation Results

1) Traceability Analysis: In the simulation a percentage of network members are marked as intruded. The simulation uses 100 random CBR pairs each generating only one packet and nodes move in 2 m/s. The following table gives the path length distribution over all the connections. Figures 6 and 7 depict the traceable ratio over different path lengths of routes for ANODR as compared to DSR and AODV. In the simulation, location privacy support is not considered. DSR and AODV are assumed to be protected by ideal futuristic hop-based link protections. In other words, each node shares a random key with another node, and no two different pairs of nodes use the same key. All DSR and AODV messages, including on demand routing packets and regular data packets, are assumed to be encrypted/decrypted by the link keys hop-by-hop.

The simulation results are averaged over 4 runs with different seeds.

hops	1	2	3	4	5	6	7	8
# of routes	45.25	19.5	20.25	6.75	4.25	3	0.5	0.5

Figure 6 shows that starting from paths of only one-hop, where ANODR and DSR expose the same amount of information (approximately same as the percentage of intruded nodes), the two protocols diverge into different trends. For DSR, traceable ratio increases when path length increases, due to the fact that longer paths are more likely to have intruded forwarding nodes. As a result, having as low as only 5 percent of intruded nodes, DSR's traceable ratio will be larger than 20 percent for paths longer than 2 hops. With 50 percent intruded nodes, DSR's traceable ratio quickly approaches 100 percent (reaches 90 percent at 3-hops long paths) when path length increases. In the graph, we see special cases in paths of 7 or more hops. This is because the chance of constructing long paths is rare in our simulation scenario. Even with multiple runs, the occurrence is too rare for meaningful statistics.

In contrast, ANODR is not sensitive to path length because the knowledge exposed to intruders is localized. Figure 6 shows that in general the traceable ratio of ANODR stays at the percentage of intruded nodes. When path grows longer, the traceable ratio will not exceed the percentage of intruded nodes. The result demonstrates ANODR's resistance to strong adversaries with node intrusion capability.

For AODV simulation runs, we implement "chessboard attack" so that collaborative intruders that locate at every other forwarding node can recover an AODV route segment (e.g., A and C in Figure 5 can recover route segment ABC by matching A's downstream forwarder with C's upstream forwarder). Figure 7 shows that for low ratio of intruded nodes, AODV and ANODR leak almost the same amount of routing information. This is because AODV is

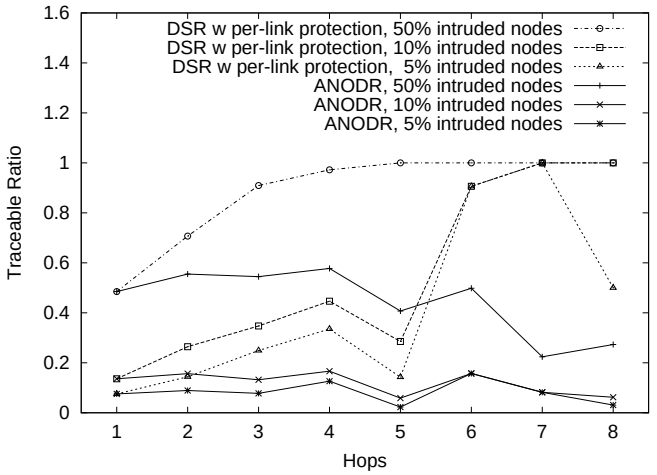

Fig. 6. **Comparison of Traceable Ratios**

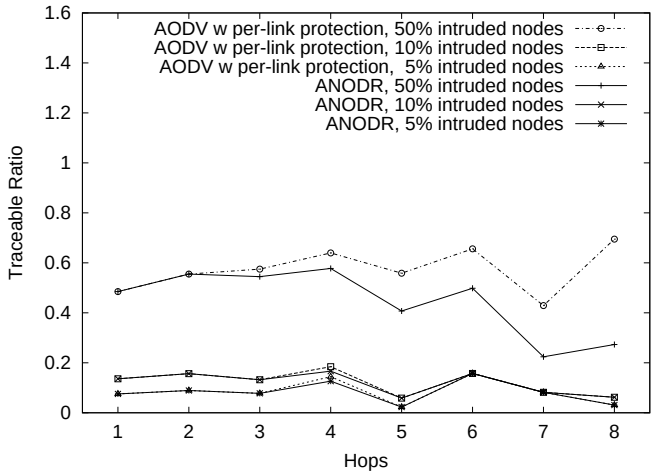

Fig. 7. **Comparison of Traceable Ratios**

similar to ANODR when internal adversary is not omnipresent (Note that AODV also provides no location privacy support). But the effectiveness of "chessboard attack" increases as number of intruded nodes increases. When there are 50 percent intruded nodes, AODV shows its weakness especially for long paths.

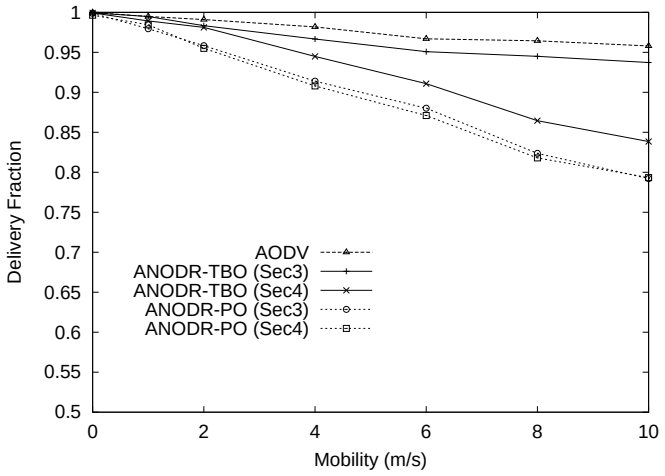

Fig. 8. **Data Packet Delivery Fraction**

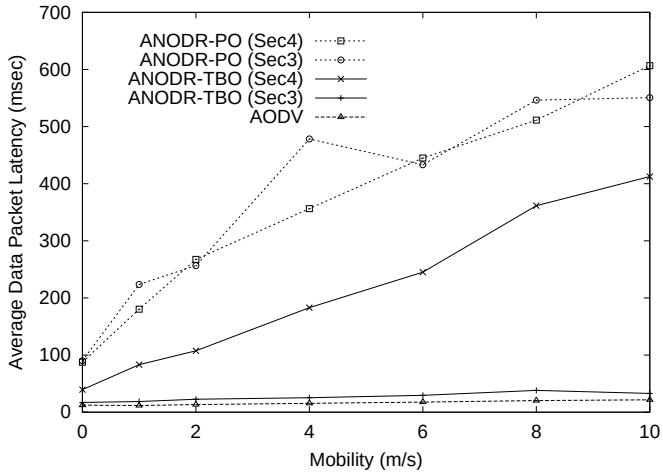

Fig. 9. **End-to-end Data Packet Latency**

2) Routing Performance: Figure 8 gives the packet delivery fraction as a function of increasing mobility. The figure shows that ANODR does not perform as good as optimized AODV. A common reason for the degradation of ANODR is the absence of route optimizations, which is expected (similar deficiency due to lack of optimizations is reported in [20]). Further, the result that "ANODR-TBO (Sec 3)" performs very close to AODV can be justified by the following two reasons: (i) The onion used in ANODR-TBO control packets and the route pseudonym field used in data packets are not big enough to incur noticeable impact to the packet delivery fraction. (ii) The 0.02ms cryptographic computation overhead for "ANODR-TBO (Sec 3)" is too small to make a difference in route discovery. The latter reason also explains why the performance of "ANODR-TBO (Sec 4)" and both ANODR-POs degrade faster than "ANODR-TBO (Sec 3)" – their long computation time prolongs the route acquisition delay, which reduces the accuracy of the newly discovered route, leading to more packet losses. Clearly, the figure shows the tradeoff concern between the performance and the degree of protection. Fortunately, even with a much stronger protection provided by "ANODR-TBO (Sec 4)", performance only degrades to 10 percent less than optimized AODV.

Figure 9 shows the average end-to-end data packet latency when mobility increases. "ANODR-TBO (Sec 3)" and

AODV exhibits very close end-to-end packet latency as they require almost the same processing time. “ANODR-TBO (Sec 4)” has longer latency than “ANODR-TBO (Sec 3)” due to additional public key processing delay during RREP phase. ANODR-POs also have extremely long end-to-end packet delay. This is largely due to its excessive public key processing at each intermediate node during both RREQ and RREP phases. The end-to-end delay increases when mobility increases, since the increasing mobility increases packet loss which triggers more route discovery, leading to increasing buffering time in waiting for a new route. The increasing trends of “ANODR-TBO (Sec 4)” and ANODR-POs are larger than the others, because more processing time is needed in the recovery phases.

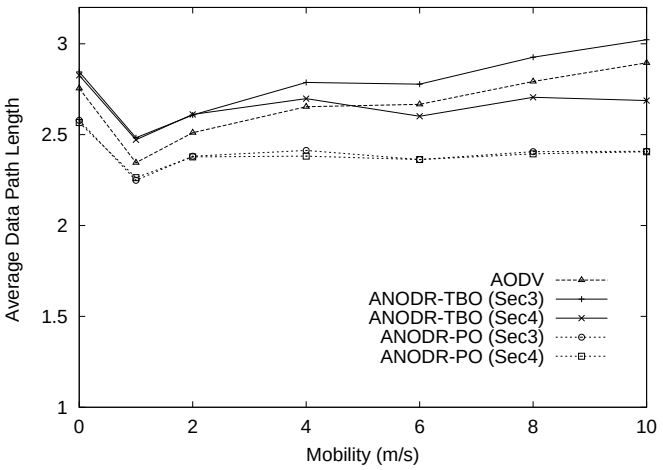

Fig. 10. **Data Packet Path Length**

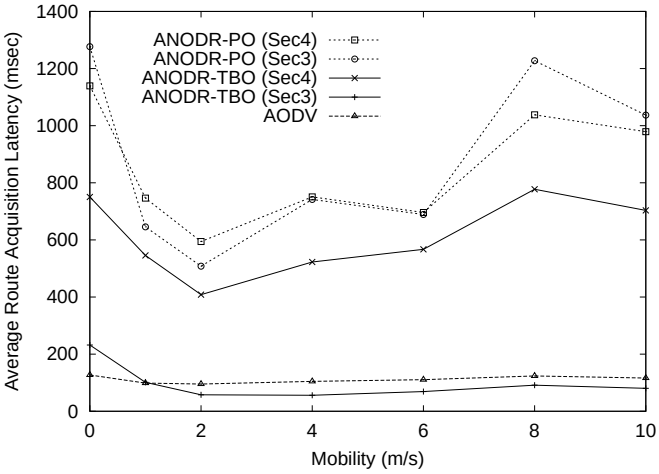

Fig. 11. **Average Route Acquisition Latency**

Figure 10 shows the average path lengths associated with data packets in terms of mobility. The figure shows that ANODR-TBO (Sec 3)” has a little longer path lengths than AODV because ANODR does not optimize an earlier established path with a later possible shorter path. The figure also shows that ANODR-TBO (Sec 4)” has less far away data delivered than the aforementioned two when mobility increases due to the easier breakage of longer paths in high mobility. ANODR-POs also have shorter path length, i.e., they deliver less packets for far away nodes than the other three schemes. This is due to the route accuracy decreases when distance increases, thus packet loss ratio increases. The figure also shows that for all the schemes mobility increases the path length, but the static case keeps long path length. For the latter phenomenon, an explanation is that almost 100 percentage packets are delivered, even to far away destinations.

Figure 11 shows average route acquisition latency when Mobility increases. The figure tells several things. The first observation is that ANODR-POs have much longer route acquisition latency than the others because they require expensive public key encryption and decryption processing at each RREQ and RREP forwarding node. Similarly, ”ANODR-TBO (Sec 4)” also incurs expensive computational cost at each RREP forwarder, thus it has considerable longer latency than ”ANODR-TBO (Sec 3)” and AODV. The figure also shows that route acquisition latency at zero mobility is longer than that at higher mobility. The reason is that when network is static, more far away destinations can be reached. Thus the time for establishing the long paths is longer than for shorter paths. In addition, as long as a path is establish, the path will be used for all the data delivery, thus the long acquisition latency has less influence for end-to-end data packet delivery.

Figure 12 shows the number of transmitted routing control packets for each successfully delivered data packet. The figure shows that when mobility increases, which incurs more link breaks, all the protocols generate more control packets due to the need for more route discoveries. However, ANODRs generate larger number of control packets than AODV at each mobility point. The reason is that without routing operation optimizations, the ANODR protocols rely more on route discoveries from the sources for repairing broken links. Especially, ANODR-POs’ long processing overhead, leading to more reports about route errors, thus triggers more route discoveries and generates more control packets.

Figure 13 gives the number of control bytes being sent in order to deliver a single data byte. The figure shows

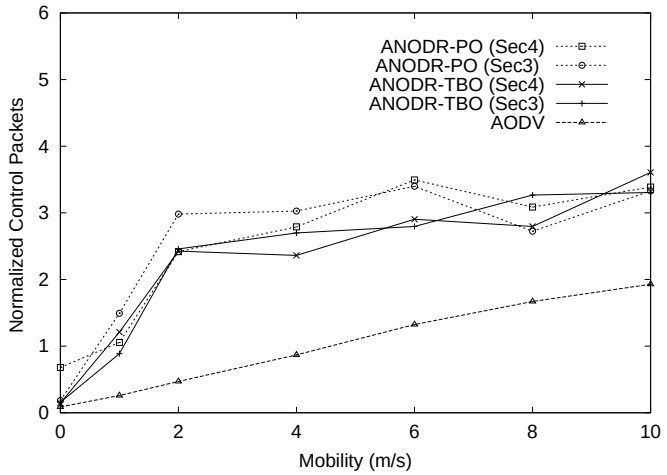

Fig. 12. **Normalized Control Packets**

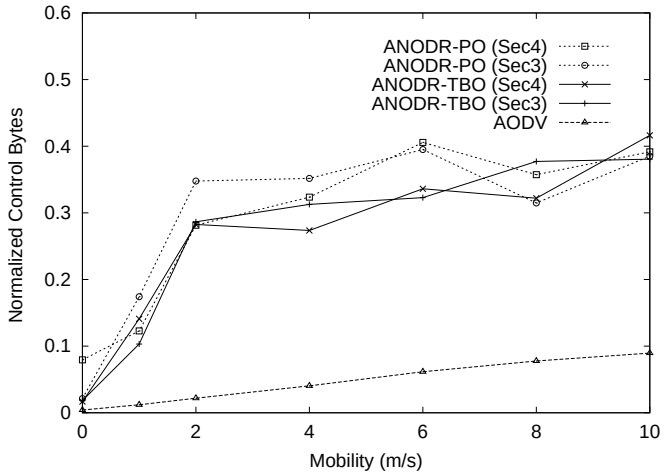

Fig. 13. **Normalized Control Bytes**

that all the ANODR variants send more control bytes than AODV. This result is expected, because they use larger packets due to global trapdoor and padded cryptographic onion. When mobility increases, the figure shows the normalized control overhead grows in all the schemes as more control packets are transmitted for path recovery. The lack of optimization in ANODR variants demonstrates here a faster increasing trend as more recovery are generated from sources so more control overhead is produced.

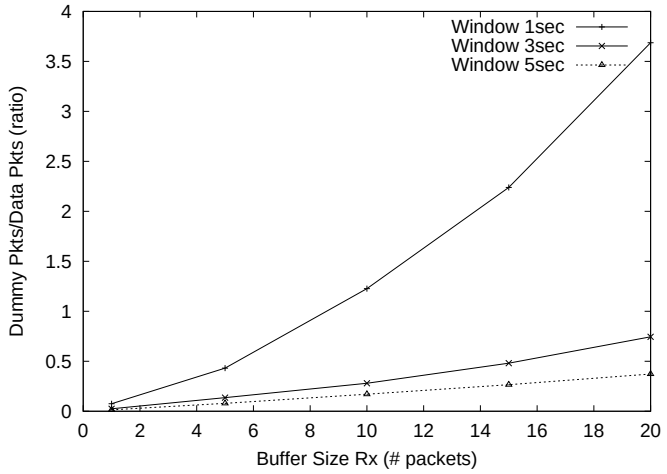

Fig. 14. **Overhead with Mixing**

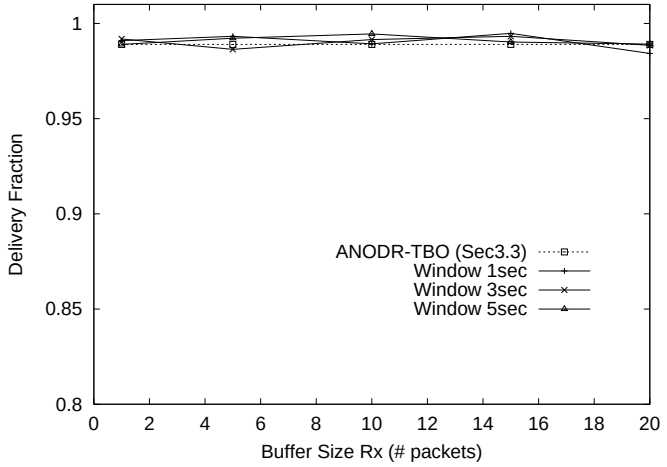

Fig. 15. **Data Delivery with Mixing**

3) Mixing Performance: Figure 14 shows the ratio of dummy packets transmitted over actual date packets transmitted. It suggests that for a fixed playout time window size t_X, the larger the playout buffer size r_X is, the more dummy packets need to be transmitted according to the formula $r_X - r$. The figure also shows that when the playout time window size t_X increases, less dummy packets are transmitted due to the increment of value r accumulated over the time window. In many cases, the dummy packet ratios are reasonably small (say, less than 100% such that averagely at least one of two transmitted data packets is real). This demonstrates that mixing technique is practical in mobile ad hoc networks if appropriate values of playout window size and buffer size are selected.

However, it is a non-trivial problem to choose the best values for playout window size t_X and buffer size r_X. Many ad hoc network dynamics, including distributed decision making, wireless bandwidth estimation, end-to-end application latency requirement, and pre-defined lower bound metrics for t_X and r_X, have significant impacts on the choice. It is appealing to employ an adaptive scheme to replace the fixed scenarios simulated in this work.

Figure 15 shows the packet delivery faction under the same mixing conditions as used in Figure 14. As a comparison, "ANODR-TBO (Sec 4)", which has been extensively studies in previous subsection, is presented here. The mobility parameter used in this experiment is equal to 1. The figure shows that "ANODR-TBO (Sec 4)" and its mixing variants perform closely. Some randomness occurs in the figure, but it does not suggest noticeable performance degradation. Thus the result suggests that the mixing packets generated under the current conditions do not affect the data packet delivery much.

7. Comparison with Related Work

In this section we compare our scheme to related route anonymity solutions, and offer an overview to related anonymity research and other security protocols proposed for ad hoc networks.

7.1. Onion Routing

The Onion Routing networks was proposed by Reed et al. [46] to protect Internet application services like Web browsing, E-mail, and electronic cash. An Onion Routing network is a group of onion proxies to forward data from one proxy to another. The Onion Routing network interfaces with an out-of-band connection through an entry funnel and an exit funnel. At the beginning of an anonymous connection, the proxy at the entry funnel sends an initialization onion to form an anonymous path in the Onion Routing network for all future messages of the same connection. Following a loose source routing paradigm, the onion travels from the proxy at the entry funnel to the designated proxy at the exit funnel. Each onion proxy en route is only allowed to know the immediate upstream proxy and downstream proxy. Each upstream onion proxy names the hop by a locally unique identifier, then the receiving proxy will change the identifier to a different value and stores the mapping locally in its forwarding table. The process is repeated until the exit funnel is reached.

ANODR bears resemblance to Onion Routing in addressing route pseudonymity. Nevertheless, ANODR is significantly different from Onion Routing in many aspects, especially with respect to single point of compromise and location privacy: (a) Onion Routing runs on top of a set of fixed number of onion proxies. A source proxy is always the single point of compromise as it knows the entire path for all routes originated from it. Such vulnerability is non-trivial in mobile ad hoc networks. ANODR employs a fully distributed approach where no single point knows more than the mapping between two sets of random route pseudonyms. (b) By using local trapdoors, in ANODR data forwarding is possible without knowing the identities of upstream and downstream forwarders. This feature is critical to provide location privacy in each locality.

7.2. Crowds

Crowds [47] is similar to Onion Routing where an initialization message forms a path of proxies through which the source sends its future messages. In a Crowds network, logically each proxy is one hop away from other proxies. Upon receiving the initialization message, each proxy decides, based on a *probability of forwarding* p_f, whether to extend the path through another proxy chosen at random with uniform probability or to become the last node on the path and communicate with the destination directly. Once a path is formed, data packets will flow from the source to the destination with route anonymity protection. The path must be reformed periodically, or when proxies on the path leave the session.

Like Onion Routing, each proxy is allowed to know its immediate upstream proxy and downstream proxy in an anonymous connection. However, each proxy also knows where the destination is, and the assumption that proxies are one logical hop away from each other makes it unattractive to mobile ad hoc networks.

7.3. Hordes

As implied in the name, Hordes [50], [51] uses multicast to hide a recipient among a horde of receivers. Assuming the underlying Internet IP multicast is efficient, the unicast operations in Crowds (or Onion Routing) can be replaced by IP multicast operations. Hordes is designed on an analytic model for identity anonymity [51], and the problem of route anonymity is left to be addressed by underlying proxy network, namely Crowds or Onion Routing network. Similar to Hordes, ANODR also explores multicast/broadcast to improve recipient anonymity. However, ANODR is an on-demand protocol, and it extensively explores trapdoor information in broadcast. These features are not discussed in Hordes' multicast mechanisms.

7.4. Untraceable last-hop wireless network

Study on untraceability in one-hop wireless cellular networks has attracted attention since mid 1990s. Samfat et al. [48] uses KryptoKnight [33], [32] to build a light-weight untraceability service for mobile users. Ateniese et al. [1] provide a comprehensive survey on related mechanisms. They devise schemes to hide user's identity from eavesdroppers, foreign domains, third party domains, and even the home domains. Some other features, such as relationship anonymity between two domains that the mobile user roams from and to, are also discussed. However, the schemes assume a network infrastructure and are not applicable to infrastructureless networks.

7.5. Anonymity without routing concerns

Privacy in mobile networks have different semantics from the traditional ones for business banking systems and the Internet. Cooper and Birman [14] identify three kinds of privacy for mobile computing: content, participant identity, and participant location. Content privacy is provided via traditional mechanisms using symmetric key and public key cryptosystems. To provide privacy for participant identities, the authors use multiple server replications and secret sharing to protect participants from disclosing their identities to a server or observer. MIX-Net is employed to provide location privacy, namely unlinkability between identities and corresponding locations.

Anonymity is not a new topic in terms of network participants' identities. Sender anonymity, recipient anonymity, relationship anonymity have been studied since early 1980s. David Chaum et al. [7], [8], [9], [10], [13], [12][44] [52] have done extensive work in developing techniques for secure, untraceable electronic transactions in banking environments or fixed networks. Chaum also proposed DC-Net [9], [11] to achieve sender anonymity against strong cryptographic attackers. Unlike MIX-Net, DC-Net provides no support other than identity anonymity. It is based on the Dining Cryptographers Problem proposed by Chaum [11]. In a DC-Net, each participant shares a secret coin flip with every other participant. The parity of the flips a participant has seen is then announced to the public. Since each flip is announced twice, the total parity should be even. To send a message, a participant incorrectly states the parity seen. This causes the total parity to be odd, which indicates a message transmission. No one except the sender knows who sent the message, unless all participants who flipped coins with the sender reveal their coin-flips among themselves. As the public key cryptography is not used in Chaum's original design, DC-Net incurs low computational cost, but at a high cost in communication overhead. However, any participant may launch a denial-of-service attack by lying. Strategies have been developed by Waidner [52] to detect such an attacker, but public key cryptography is employed again and the countermeasure incurs expensive computation overhead.

7.6. Other security supports in mobile ad hoc networks

Recently many solutions are proposed for ad hoc routing schemes to resist routing disruption attacks. Based on the TESLA protocol [40], [41], Hu et al. proposed Ariadne [20] and SEAD [19] as the secured version of DSR and DSDV, respectively. The TESLA protocol can efficiently differentiate authenticated packets from attacking packets with little computational overhead. Papadimitratos and Haas [35] proposed an end-to-end secure route discovery protocol for ad hoc routing. Unlike Ariadne, the protocol does not authenticate intermediate nodes which forward route requests. Dahill et al. [15] observed that light weight RSA cryptosystem (512-bit key) features good performance on mobile devices with enough computational resource. They presented ARAN, a secure AODV protocol based on the observation. Yang and Lu [53] proposed a secure AODV protocol based on "watchdog" mechanism [30] and probabilistic intrusion detection. Other related research includes ubiquitous authentication in ad hoc networks. Kong et al. [25] proposed an access control scheme for infrastructureless networks. Hubaux et al. [21] studied how to authenticate mobile nodes following PGP's web-of-trust paradigm. The design of ANODR is orthogonal to these security schemes. ANODR can be potentially integrated with these security schemes to devise untraceable and robust ad hoc routing protocols.

8. CONCLUSIONS AND FUTURE WORK

In this work we propose ANODR, an anonymous on-demand routing protocol for mobile ad hoc networks deployed in hostile environments. We have addressed two close-related unlinkability problems, namely *route*

anonymity and *location privacy*. Based on a route pseudonymity approach, ANODR prevents strong adversaries, such as node intruders and omnipresent eavesdroppers, from exposing local wireless transmitters' identities and tracing ad hoc network packet flows. Moreover, ANODR also demonstrates that untraceable data forwarding without encrypted routing header can be efficiently realized. The design of ANODR is based on "*broadcast with trapdoor information*", a novel network security concept with hybrid features merged from both network concept "broadcast" and security concept "trapdoor information". This network security concept can be applied to multicast communication as well. Currently we are working towards solutions to adaptively adjust ANODR's playout window size and buffer size, to improve ANODR's performance in high mobility scenarios, and to devise an anonymous untraceable multicast routing scheme for mobile ad hoc networks.

REFERENCES

[1] G. Ateniese, A. Herzberg, H. Krawczyk, and G. Tsudik. Untraceable Mobility or How to Travel *Incognito*. *Computer Networks*, 31(8):871–884, 1999.

[2] ATM Forum. Asynchronous Transfer Mode. `http://www.atmforum.org/`.

[3] S. Basagni, K. Herrin, E. Rosti, and D. Bruschi. Secure Pebblenets. In *MobiHoc*, pages 156–163, 2001.

[4] O. Berthold, H. Federrath, and M. Köhntopp. Project Anonymity and Unobservability in the Internet. In *Computers Freedom and Privacy Conference 2000 (CFP 2000), Workshop on Freedom and Privacy by Design*, 2000.

[5] M. Blum and S. Micali. How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits. In *Symposium on Foundations of Computer Science (FOCS)*, pages 112–117, 1982.

[6] M. Brown, D. Cheung, D. Hankerson, J. L. Hernandez, M. Kurkup, and A. Menezes. PGP in Constrained Wireless Devices. In *USENIX Security Symposium (Security '00)*, 2000.

[7] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[8] D. L. Chaum. Blind Signatures for Untraceable Payments. In D. L. Chaum, R. L. Rivest, and A. T. Sherman, editors, *CRYPTO'82, Lecture Notes in Computer Science*, pages 199–203, 1983.

[9] D. L. Chaum. Security Without Identification: Transaction Systems to Make Big Brother Obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[10] D. L. Chaum. Showing Credentials Without Identification: Signatures Transferred Between Unconditionally Unlinkable Pseudonyms. In F. Pichler, editor, *EUROCRYPT'85, Lecture Notes in Computer Science 219*, pages 241–244, 1986.

[11] D. L. Chaum. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. *Journal of Cryptology*, 1(1):65–75, 1988.

[12] D. L. Chaum. Showing Credentials without Identification: Transfeering Signatures between Unconditionally Unlinkable Pseudonyms. In J. Seberry and J. Pieprzyk, editors, *AUSCRYPT'90, Lecture Notes in Computer Science 453*, pages 246–264, 1991.

[13] D. L. Chaum, A. Fiat, and M. Naor. Untraceable Electronic Cash. In S. Goldwasser, editor, *CRYPTO'88, Lecture Notes in Computer Science 403*, pages 319–327, 1989.

[14] D. A. Cooper and K. P. Birman. Preserving Privacy in a Network of Mobile Computers. In *IEEE Symposium on Research in Security and Privacy*, pages 26–38, 1995.

[15] B. Dahill, B. N. Levine, E. Royer, and C. Shields. A Secure Routing Protocol for Ad Hoc Networks. In *10th International Conference on Network Protocols (ICNP'02)*, 2002.

[16] O. Goldreich and L. A. Levin. A Hard-Core Predicate for all One-Way Functions. In *Symposium on the Theory of Computation (STOC)*, pages 25–32, 1989.

[17] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A Pseudorandom Generator from any One-way Function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[18] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. Proactive Secret Sharing or: How to Cope with Perpetual Leakage. extended abstract, IBM T.J. Watson Research Center, November 1995.

[19] Y.-C. Hu, D. B. Johnson, and A. Perrig. SEAD: Secure Efficient Distance Vector Routing in Mobile Wireless Ad Hoc Networks. In *Fourth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'02)*, 2002.

[20] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: A Secure On-demand Routing Protocol for Ad Hoc Networks. In *MOBICOM*, 2002.

[21] J. Hubaux, L. Buttyan, and S. Capkun. The Quest for Security in Mobile Ad Hoc Networks. In *MobiHOC*, 2001.

[22] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In T. Imielinski and H. Korth, editors, *Mobile Computing*, volume 353, pages 153–181. Kluwer Academic Publishers, 1996.

[23] D. B. Johnson, D. A. Maltz, Y.-C. Hu, and J. G. Jetcheva. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks, February 2002.

[24] D. Kesdogan, J. Egner, and R. Buschkes. Stop-and-go MIXes Providing Probabilistic Security in an Open System. *Second International Workshop on Information Hiding, Lecture Notes in Computer Science 1525*, pages 83–98, 1998.

[25] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing Robust and Ubiquitous Security Support for Mobile Ad-hoc Networks. In *ICNP'01*, pages 251–260, 2001.

[26] H. Krawcyzk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. `http://www.ietf.org/rfc/rfc2104.txt`, 1997.

[27] S.-J. Lee and M. Gerla. Split Multipath Routing with Maximally Disjoint Paths in Ad Hoc Networks. In *ICC*, pages 3201–3205, 2001.

[28] A. K. Lenstra and E. R. Verheul. Selecting Cryptographic Key Sizes. In *Public Key Cryptography*, pages 446–465, 2000.

[29] M. K. Marina and S. R. Das. Ad Hoc On-demand Multipath Distance Vector Routing. In *ICNP*, pages 14–23, 2001.

[30] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. In *MOBICOM*, 2000.

[31] A. J. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[32] R. Molva, D. Samfat, and G. Tsudik. Authentication of Mobile Users. *IEEE Network*, 8(2):26–34, 1994.

[33] R. Molva, G. Tsudik, E. V. Herreweghen, and S. Zatti. KryptoKnight Authentication and Key Distribution System. In Y. Deswarte, G. Eizenberg, and J.-J. Quisquater, editors, *ESORICS'92, Lecture Notes in Computer Science 648*, pages 155–174, 1992.

[34] G. Montenegro and C. Castelluccia. Statistically Unique and Cryptographically Verifiable (SUCV) Identifiers and Addresses. In *Network and Distributed System Security Symposium (NDSS)*, 2002.

[35] P. Papadimitratos and Z. J. Haas. Secure Routing for Mobile Ad Hoc Networks. In *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)*, 2002.

[36] P. Papadimitratos, Z. J. Haas, and E. G. Sirer. Path Set Selection in Mobile Ad Hoc Networks. In *MOBIHOC*, pages 160–170, 2002.

[37] V. D. Park and M. S. Corson. A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks. In *INFOCOM*, pages 1405–1413, 1997.

[38] M. R. Pearlman, Z. J. Haas, P. Sholander, and S. S. Tabrizi. On the Impact of Alternate Path Routing for Load Balancing in Mobile Ad Hoc Networks. In *MOBIHOC*, pages 3–10, 2000.

[39] C. E. Perkins and E. M. Royer. Ad-Hoc On-Demand Distance Vector Routing. In *IEEE WMCSA'99*, pages 90–100, 1999.

[40] A. Perrig, R. Canetti, B. Briscoe, D. Tygar, and D. Song. TESLA: Multicast Source Authentication Transform. draft-irtf-smug-tesla-00.txt, June 2001.

[41] A. Perrig, R. Canetti, D. Tygar, and D. Song. The TESLA Broadcast Authentication Protocol. *RSA CryptoBytes*, 5(2):2–13, 2002.

[42] A. Pfitzmann and M. Köhntopp. Anonymity, Unobservability, and Pseudonymity - A Proposal for Terminology. In H. Federrath, editor, *DIAU'00, Lecture Notes in Computer Science 2009*, pages 1–9, 2000.

[43] A. Pfitzmann, B. Pfitzmann, and M. Waidner. ISDNMixes: Untraceable Communication with Very Small Bandwidth Overhead. In *GI/ITG Conference: Communication in Distributed Systems*, pages 451–463, 1991.

[44] A. Pfitzmann and M. Waidner. Networks Without User Observability: Design Options. In F. Pichler, editor, *EUROCRYPT'85, Lecture Notes in Computer Science 219*, pages 245–253, 1986.

[45] J.-F. Raymond. Traffic Analysis: Protocols, Attacks, Design Issues, and Open Problems. In H. Federrath, editor, *DIAU'00, Lecture Notes in Computer Science 2009*, pages 10–29, 2000.

[46] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Anonymous Connections and Onion Routing. *IEEE Journal on Selected Areas in Communications*, 16(4), 1998.

[47] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[48] D. Samfat, R. Molva, and N. Asokan. Untraceability in Mobile Networks. In *MOBICOM*, pages 26–36, 1995.

[49] Scalable Network Technologies (SNT). QualNet. `http://www.qualnet.com/`.

[50] C. Shields. Secure Hierarchical Multicast Routing and Multicast Internet Anonymity. PhD Thesis, Computer Engineering, University of California, Santa Cruz, June 1999.

[51] C. Shields and B. N. Levine. A protocol for anonymous communication over the Internet. In *ACM Conference on Computer and Communications Security (CCS 2000)*, pages 33–42, 2000.

[52] M. Waidner. Unconditional Sender and Recipient Untraceability in spite of Active Attacks. In J.-J. Quisquater and J. Vandewalle, editors, *EUROCRYPT'89, Lecture Notes in Computer Science 434*, pages 302–319, 1990.

[53] H. Yang and S. Lu. Self-Organized Network Layer Security in Mobile Ad Hoc Networks. In *First ACM Workshop on Wireless Security (WiSe)*, pages 11–20, 2002.

[54] A. C.-C. Yao. Theory and Applications of Trapdoor Functions (Extended Abstract). In *Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, 1982.