ANODR: ANonymous On Demand Routing with Untraceable Routes for Mobile Ad-hoc Networks

Jiejun Kong, Xiaoyan Hong
Computer Science Department
University of California, Los Angeles, CA 90095
{jkong,hxy}@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.

Categories and Subject Descriptors

C.2.2 [**Computer-Commmunication Networks**]: Network Protocols—*Routing protocols*

General Terms

Security, Design, Measurement, Experimentation, Performance

Keywords

Anonymity, Untraceability, Pseudonymity, Broadcast, Trapdoor, On-demand Routing, Mobile Ad-hoc Network

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 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

MobiHoc'03, June 1–3, 2003, Annapolis, Maryland, USA

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 [31, 32] 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.

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 4. Our implementation and performance evaluation are shown in Section 5. Finally Section 6 concludes this paper.

2. UNDERLYING MODELS AND TOOLS

2.1 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 [26] 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 seek to realize relationship anonymity among the corresponding set of route pseudonyms. The route pseudonymity approach differentiates this 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 a mobile node's identity and its location. 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.

2.2 Notations

In the paper we will use the notations shown in Table 1.

2.3 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

Table 1: 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

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 [25]) 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 [9]). 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.

2.4 Network model

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. 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 [3], ISDN-MIXes [27], Stop-and-Go-MIXes [13], Onion Routing [29], and many others, have been based on Chaum's anonymous email solution: a network of *MIX*es [6]. 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

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

Such a cryptographic data structure is named as "*onion*" in Internet Onion Routing networks [29]. 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 [19]. 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 rationales

Broadcasting with trapdoor assignment As shown in previous research [31, 32], 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 5), 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 [5].

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.

3.2 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

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

where (i) $seqnum$ is a globally unique sequence number[1]. (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.

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 1, 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

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

   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 this section).

   As depicted in Figure 1, 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.

   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.

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

[1] 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 [20].

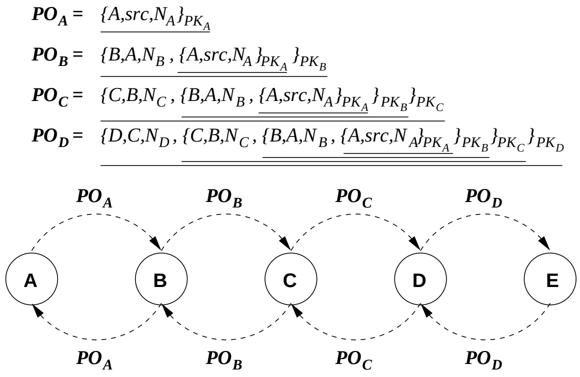
$PO_A = \{A,src,N_A\}_{PK_A}$
$PO_B = \{B,A,N_B, \{A,src,N_A\}_{PK_A}\}_{PK_B}$
$PO_C = \{C,B,N_C, \{B,A,N_B, \{A,src,N_A\}_{PK_A}\}_{PK_B}\}_{PK_C}$
$PO_D = \{D,C,N_D, \{C,B,N_C, \{B,A,N_B, \{A,src,N_A\}_{PK_A}\}_{PK_B}\}_{PK_C}\}_{PK_D}$
PO_A
PO_B
PO_C
PO_D
A
B
C
D
E
PO_A
PO_B
PO_C
PO_D

Figure 1: ANODR-PO: Anonymous route discovery using public key cryptography (A single path showed from source A to destination E)

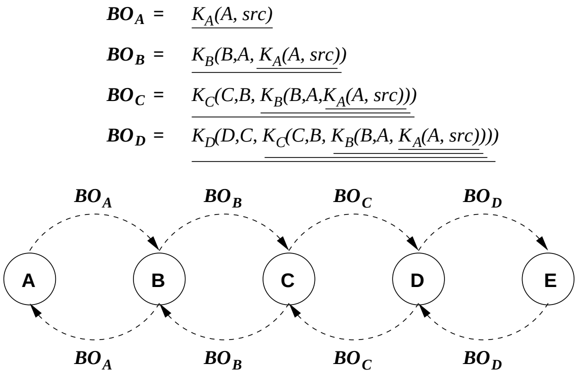
$BO_A = K_A(A, src)$
$BO_B = K_B(B,A, K_A(A, src))$
$BO_C = K_C(C,B, K_B(B,A,K_A(A, src)))$
$BO_D = K_D(D,C, K_C(C,B, K_B(B,A, K_A(A, src))))$
BO_A
BO_B
BO_C
BO_D
A
B
C
D
E
BO_A
BO_B
BO_C
BO_D

Figure 2: ANODR-BO: Anonymous route discovery using Boomerang Onion (A single path showed from source A to destination E)

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.

The efficient anonymous route discovery protocol is depicted in Figure 2. 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 3 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).

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.

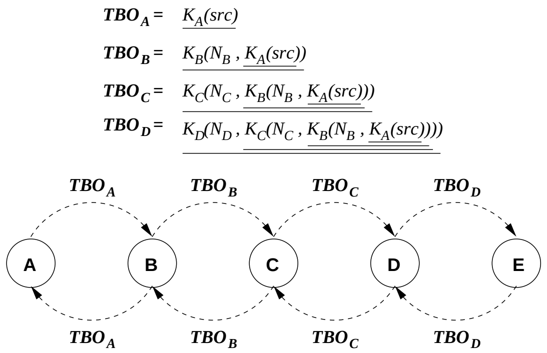
$TBO_A = K_A(src)$
$TBO_B = K_B(N_B, K_A(src))$
$TBO_C = K_C(N_C, K_B(N_B, K_A(src)))$
$TBO_D = K_D(N_D, K_C(N_C, K_B(N_B, K_A(src))))$
TBO_A
TBO_B
TBO_C
TBO_D
A
B
C
D
E
TBO_A
TBO_B
TBO_C
TBO_D

Figure 3: ANODR-TBO: Anonymous route discovery using Trapdoor Boomerang Onion (A single path showed from source A to destination E)

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.

3.3 Discussions

Unlinkable 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

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

and RREP packet format is changed to be

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

where K_{seed} is a nonce (same as N in the § 3.2 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 [17]) 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. 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[33], Blum, and Micali[4] 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 [7][8]. 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[25] 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.

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 [15] 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" [19]. This probability is much higher than p_c as we currently choose l to be 128-bit.

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 [10], 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

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

where K_T is destination's TESLA secret and K_c becomes a commitment key. Consequently RREP format is instantiated as

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

where K_c' is the anonymous proof presented by the destination. Any forwarding node can verify the anonymous proof of trapdoor opening by checking $K_c(dest) \overset{?}{=} K_c'(dest)$.

Here ANODR employs "*trapdoor commitment*", a cryptographic concept explores the collision-resistant property of one-way functions. That is, given an output of one-way function $K_c(dest)$, it is computationally hard to find the input, or another input collision that can produce the same output. TESLA[25] is an exemplary trapdoor commitment protocol. In TESLA, the output of a one-way function is published as a commitment before the the corresponding input is revealed. When both the input and output are available, any verifier can efficiently validate the commitment by applying the one-way function on them. In ANODR, $K_c(dest)$ embedded in RREQ packet is a public commitment made for the destination by the source. Later the destination node can present the input K_c as the proof of furnishing the commitment.

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[23][16][18][21]. 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.

4. UNTRACEABILITY ANALYSIS AND COMPARISONS TO RELATED WORK

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 [11, 22, 24]. As depicted in Figure 4, 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 [1]. 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.

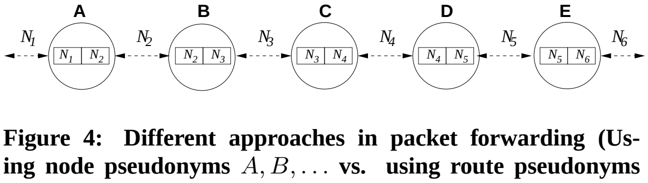

Figure 4: Different approaches in packet forwarding (Using node pseudonyms $A, B, \ldots$ vs. using route pseudonyms $N_1, N_2, \ldots$)

4.1 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 4, 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≤i≤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[2], 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.

4.2 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 [28]. 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 reorder 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 [27][13][3]. 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 upperbound[3] 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.

4.3 Comparison with DSR and AODV

DSR [11] 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 [24] 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 DSR and AODV, 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. do "matching-payload-attack" and check the chain of senders and receivers). 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 neces-

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

[3] 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.

sary 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.

4.4 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. [2] 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 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:

- 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.
- 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.
- When node intrusion is possible, it is a non-trivial issue to minimize the subsequent damages. Even after a hop-based link encryption scheme is realized to protect all routing information, a DSR route is traceable by a single intruder en route, while an AODV route is traceable by collaborative intruders that locate at every other forwarding node.

5. IMPLEMENTATION AND EVALUATION

5.1 Cryptographic implementation

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 each onion to be at least 400-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 [12]. In practice, the number 400 can be replaced by a number based on the estimation of the hop count of the network's diameter.

The processing overhead used in our simulation is based on actual measurement on a low-end device. Table 2 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 2: 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

5.2 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, we compare ANODR to DSR. 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. We use *traceable ratio* R (Section 4) to quantify the effect of node intrusions. The traceable ratio for a DSR route is 0 when none of

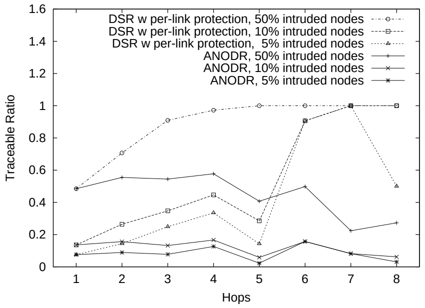

Figure 5: Comparison of traceable ratios

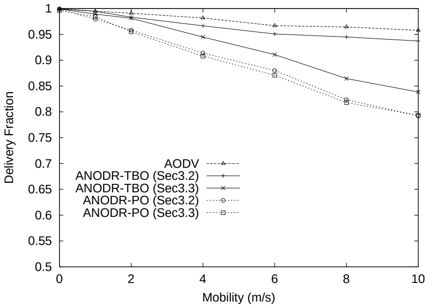

Figure 6: Data Packet Delivery Fraction

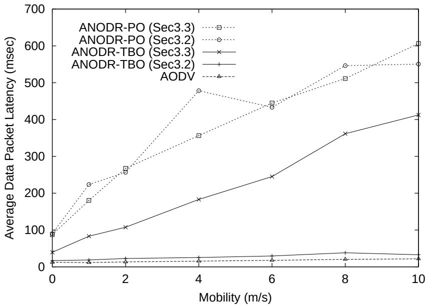

Figure 7: End-to-end Data Packet Latency

the nodes en route is intruded, or is 1 otherwise.

Then we evaluate the performance of ANODR-TBO proposed in both Section 3.2 and 3.3 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 3.2)" and "ANODR-TBO (Sec 3.3)", 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 3.3)", 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 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. (v) *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.

5.3 Simulation Model

The routing protocols are implemented within QualNet^{TM} [30], 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×300m with 50 nodes initially uniformly distributed. The transmission range is 250m. *Random Waypoint* mobility model [11] 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.

5.4 Simulation Results

5.4.1 Traceability Analysis

In the simulation a percentage of network members are marked as intruded. Figure 5 depicts the traceable ratio over different path lengths of routes for ANODR and DSR. 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. The 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

The figure shows that starting from paths of only one-hop, where the two protocols 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

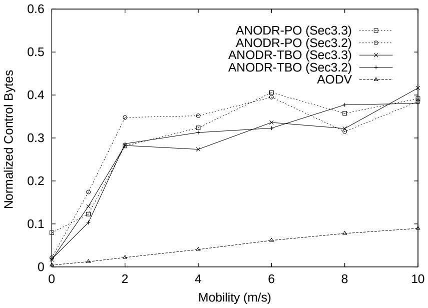

Figure 8: Normalized Control Bytes

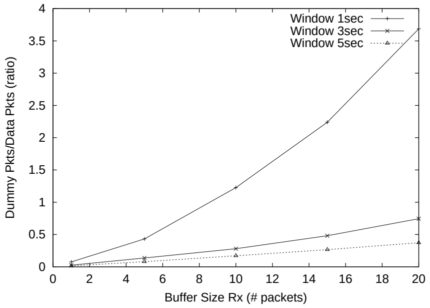

Figure 9: Overhead with Mixing

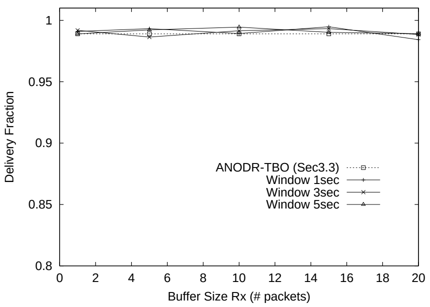

Figure 10: Data Delivery with Mixing

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 5 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.

5.4.2 Routing Performance

Figure 6 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 optimization operations, which is expected (similar deficiency due to lack of optimizations is reported in [10]). Further, the result that "ANODR-TBO (Sec 3.2)" 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 3.2)" is too small to make a difference in route discovery. The latter reason also explains why the performance of "ANODR-TBO (Sec 3.3)" and both ANODR-POs degrade faster than "ANODR-TBO (Sec 3.2)" – 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 3.3)", performance only degrades to 10 percent less than optimized AODV.

Figure 7 shows the average end-to-end data packet latency when mobility increases. "ANODR-TBO (Sec 3.2)" and AODV exhibits very close end-to-end packet latency as they require almost the same processing time. "ANODR-TBO (Sec 3.3)" has longer latency than "ANODR-TBO (Sec 3.2)" 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 delay trend of "ANODR-TBO (Sec 3.3)" and ANODR-POs 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.

Figure 8 gives the number of control bytes being sent in order to deliver a single data byte. The figure shows 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.

5.4.3 Mixing Performance

Figure 9 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 10 shows the packet delivery faction under the same mixing conditions as used in Figure 9. As a comparison, "ANODR-TBO (Sec 3.3)", 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 3.3)" 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.

6. 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.

Acknowledgments Our greatest thanks go to Professor Mario Gerla (gerla@cs.ucla.edu) for finding and actualizing the research topic. More details of this design is available in our technical report [14]. We must also express our gratitude to anonymous untraceable reviewers for their very helpful comments on the new anonymous untraceable routing scheme.

7. REFERENCES

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

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

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.

[8] J. Hástad, 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.

[9] 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.

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

[11] 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.

[12] 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.

[13] 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.

[14] J. Kong, X. Hong, and M. Gerla. An Anonymous On Demand Routing with Untraceable Routes for Mobile Ad-hoc Networks. Technical Report TR-030020, Dept. of Computer Science, UCLA, 2003.

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

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

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

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

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

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

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

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

[23] 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.

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

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

[26] 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.

[27] 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.

[28] 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.

[29] 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.

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

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

[32] 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.

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