Towards Address Privacy in Mobile Ad Hoc Networks

Yanchao Zhang  
Dept. of Electrical and Computer Engineering  
New Jersey Institute of Technology  
yczhang@njit.edu

Kui Ren  
Dept. of Electrical and Computer Engineering  
Illinois Institute of Technology  
kren@ece.iit.edu

ABSTRACT

Security concerns are an impediment to deploying mobile ad hoc networks (MANETs) in hostile environments. This paper proposes and investigates solutions to a new security requirement called address privacy to prevent attackers from ascertaining network addresses of MANET principals. Lack of address privacy is devastating to critical MANET operations. For example, if knowing the network address of a target principal, attackers can easily locate the target by passively monitoring the open wireless channel and then launch a pinpoint attack. We present Swarms, the first solution to address privacy in MANETs. Swarms eliminates the conventionally explicit one-on-one mappings between MANET principals and network addresses and allows any two principals to communicate while blind to each other’s address. We quantitatively measure the address privacy offered by Swarms via an entropy-based information-theoretic metric.

Categories and Subject Descriptors

C.2.0 [Computer Communication Networks]: General—Security and protection; C.2.1 [Computer Communication Networks]: Network Architecture and Design—Wireless communication

General Terms
Design, Security, Performance

Keywords
Mobile ad hoc networks, security, privacy, routing

1. INTRODUCTION

A mobile ad hoc network (MANET) consists of mobile nodes communicating via multi-hop wireless links. Each node is affiliated with a principal (a person or a piece of equipment) that interacts with others over the MANET substrate. Security designs have been considered indispensable for both military and civilian MANETs. Conventional MANET security research focuses on securing node-to-node communications while largely overlooking the role of principals. Partially filling this gap, this paper studies address privacy of MANET principals to prevent attackers from ascertaining their network addresses.

Lack of address privacy is highly undesirable in both military and civilian MANETs. Note that the open nature of wireless channels couples the address privacy of MANET principals tightly with their location privacy. In particular, if knowing the address of a target principal, attackers can locate him by overhearing and analyzing radio messages, thus breaching his location privacy. This is unacceptable in both military and civilian settings. For example, attackers in a military MANET can locate and launch pinpoint attacks on VIP principals after obtaining their addresses. Attackers may also profile the movement of principals to infer secret tactical information such as a forthcoming action. Principals in a civilian MANET often have similar requirements for address and location privacy, as many do not want to expose their whereabouts. This situation highlights the necessity of address privacy in MANETs. As far as we know, address privacy in MANETs has not received any attention so far. Instead, it is a long-held implicit assumption that any two communicating principals in a MANET know each other’s address. Therefore, if either is compromised, attackers will immediately know the other’s address, which is a clear violation of address privacy.

Our contributions are mainly threefold. First, we identify address privacy as a security requirement for MANETs, which is likely to inspire new research ideas. Second, we present a solution to address privacy in MANETs, named Swarms. Third, we quantitatively analyze the address privacy offered by Swarm via an information-theoretic metric. Swarms allows any two principals to communicate without knowing each other’s address, thus improving the other’s address privacy when either is compromised. This is achieved by hiding each principal’s address within a set of addresses (called a swarm) and routing a packet via a sequence of swarms from the source to the destination.

Swarms is motivated by a common scene in spy movies. For example, suppose that Alice need deliver a message to Bob, but she only knows Bob’s pseudonym, say Alex, instead of his address; there is a middleman Tom knowing the address of Bob, but he associates it with Alex. In this scenario, Alice can address the message to Alex and then deliver it to Tom who subsequently forwards the message to Alex (who is actually Bob). The whereabouts of Bob can thus be protected from both Alice and Tom. Swarms extends this familiar
scene to enable both efficient communications and adequate address privacy via a de Bruijn graph [5]. Built atop the MANET substrate, Swarms can be easily integrated with any underlying routing protocol. This nice feature would greatly enhance the feasibility and applicability of Swarms.

The rest of this paper is organized as follows. Section 2 presents network and threat models. Section 3 details the Swarms protocol, followed by quantitative analysis in Section 4.2. This paper is then concluded in Section 5.

2. PRELIMINARIES

2.1 Network Model

We consider single-authority MANETs, of which typical examples are those deployed in military, counter-terrorist, and law enforcement actions. The network consists of $N$ principals whose IDs compose the set $U = \{u_i\}_{i=1}^{N}$. Hereafter we will refer to the principal with ID $u_i$ as principal $u_i$ for convenience. Each $u_i$ is equipped with a mobile node (radio device) that has a layer-2 MAC address and a layer-3 network address denoted by $IP_i$, which are also referred to as $u_i$’s MAC and network addresses. This means that we have to provide both MAC-address privacy and network-address privacy so that attackers cannot link an interested MAC or network address to a certain principal.

MAC addresses are only used in local communications, and their privacy can be easily achieved. For example, if each node has a unique fixed MAC address, the mapping between a MAC address and the corresponding principal should be kept confidential to the trusted authority (not in the MANET). Another way of dissociating a principal from his nodal MAC address is to let his node use a dynamic MAC address [4]. Both approaches can effectively prevent attackers from obtaining the MAC address of a target principal.

Assuming that MAC-address privacy can be preserved, we only focus on network-address privacy in this paper and may use the term “address privacy” for short. More specifically, we want to keep attackers from knowing the address-ID mappings $\{u_i \leftrightarrow IP_i\}_{i=1}^{N}$. We will refer to the node with address $IP_i$ as node $IP_i$ henceforth for simplicity. In addition, when we say that $u_i$ sends a message, it should be understood that the message is actually and appears to be sent from node $IP_i$ as viewed by others.

2.2 Threat Model

The goal of the adversary is to break network-address privacy, i.e., obtaining the one-on-one address-ID mappings. In particular, the adversary attempts to ascertain the corresponding address (or principal) of any given principal (or address). It is beyond the scope of this paper to mitigate other important attacks on MANETs such as physical-layer jamming, MAC-layer misbehavior, or routing disruption. The adversary has a number of agents in the target MANET from which to collect information for analyzing address-ID mappings. Some agents are external attackers that do not belong to the MANET and only passively eavesdrop on radio transmissions. Other agents are internal attackers that are legitimate MANET principals compromised and fully controlled by the adversary. We also assume that compromising a principal amounts to compromising his mobile node, so we will not differentiate compromised principals and nodes hereafter. To design a feasible solution, we follow the conventional assumption that non-compromised nodes are always the majority. The capabilities and strategies of the adversary will be further illustrated when appropriate.

3. SWARMS

In this section, we first introduce two basic solutions that motivate the design of Swarms. Then we illustrate Swarms and defer its security analysis to Section 4. In what follows, we assume that each principal $u_i$ has a public/private key pair $PU_i/PR_i$. Efficient public-key management in MANETs can be realized via many schemes such as [9]. Also let $E(PU_i, X)$ denote asymmetric encryption of plaintext $X$ using $PU_i$ and $D(PR_i, Y)$ denote asymmetric decryption of ciphertext $Y$ using $PR_i$.

3.1 Basic Solutions

Network addresses of MANET principals were seldom considered necessary to be kept secret, and there is often a known unique mapping between a network address and the corresponding principal. Consider an example in which principal $u_s$ need send messages to $u_d$. He has to know $u_d$’s address $IP_d$ before sending packets via the underlying routing protocol. There are various ways for $u_s$ to acquire $IP_d$. For example, he can know $u_d \leftrightarrow IP_d$ from the authority before network deployment. This conventional method is very straightforward and efficient, but it may cause severe security issues. For instance, if managing to compromise $u_s$ and thus know $u_s \leftrightarrow IP_s$, attackers can precisely locate $u_d$. This is extremely dangerous if $u_d$ happens to have a critical role (e.g., a commander). We can improve $u_d$’s address privacy through the following two solutions.

3.1.1 Privacy zone

In this solution, the address set $\{IP_i\}_{i=1}^{N}$ is partitioned into mutually disjoint subsets, each called a privacy zone. The address of each principal is thus unidentifiable within a privacy zone that is known to all his potential correspondents (i.e., principals allowed to directly send messages to principal $u_i$). This time $u_d$ only knows $ZONE_d$, and $IP_d \in ZONE_d$, but he cannot single $IP_d$ out.
To send a message DATA to \( u_d \), principal \( u_s \) transmits \( E(\text{PU}_s u_s, u_d, \text{DATA}) \) individually to each node in \( \text{ZONE}_d \), where \( \text{PU}_d \) is the public key of \( u_d \). Upon receipt of the packet, each node in \( \text{ZONE}_d \) tries to decrypt it using the private key of its affiliated principal, and only \( IP_d \) can succeed in finding \( u_d \) in the decryption result which is then passed to \( u_d \). Fig. 1(a) shows an example where \( \text{ZONE}_d = \{ IP_1, IP_2, IP_3 \} \) and each virtual link may correspond to single/multi-hop physical links in the MANET substrate.

Let us briefly analyze the security of this technique. \( u_s \) only knows that \( u_d \)’s address is in \( \text{ZONE}_d \), but he cannot associate \( IP_d \) with \( u_d \). As a result, even after compromising \( u_s \) and knowing \( \text{ZONE}_d \), the adversary cannot easily establish the mapping \( u_d \leftrightarrow IP_d \). Instead, he must attempt to compromise nodes in \( \text{ZONE}_d \) one by one to find principal \( u_d \) and may have to try on the average \( \left\lceil \frac{\text{ZONE}_d - 1}{2} \right\rceil \) times.

### 3.1.2 Packet rerouting

Motivated by the spy-movie scene in Section 1, we can also improve \( u_d \)’s address privacy by interposing \( k \) middlemen \( \{u_j\}_{j=1}^k \) between \( u_s \) and \( u_d \), through which packets from \( u_s \) to \( u_d \) will be rerouted. \( u_s \) also has a pseudonym \( \tilde{u}_s \). Now \( u_s \) knows \( \{\tilde{u}_s, \tilde{u}_d, IP_d\} \), \( u_j \) (\( 1 \leq j \leq k - 1 \)) knows \( \{\tilde{u}_s, IP_{j+1}\} \), and \( u_d \) knows \( \{\tilde{u}_d, IP_d\} \). To send a message \( \text{DATA} \) to \( u_d \), principal \( u_s \) transmits \( \text{E}(\text{PU}_s u_s, u_d, \text{DATA}) \) to the node \( IP_1 \). Upon receipt of it, node \( IP_1 \) forwards it to node \( IP_d \) based on the embedded \( \tilde{u}_d \). This process continues until node \( IP_d \) receives and terminates the forwarding of \( \text{E}(\text{PK}_d, u_s, u_d, \text{DATA}) \). Fig. 1(b) shows an example with two middlemen, where each virtual link again may correspond to single/multi-hop physical links.

We now briefly analyze the security of this technique. Assume that effective countermeasures (such as [8]) are in place to prevent the adversary from identifying that the forwarding of packet \( \text{E}(\text{PK}_d, u_s, u_d, \text{DATA}) \) is terminated at node \( IP_d \). All the middlemen only know that they are forwarding information for principal \( \tilde{u}_d \), and even \( u_i \) cannot link \( IP_d \) to \( u_d \) or \( u_s \). Suppose that \( u_d \) is compromised. The adversary only knows the address of the next middleman towards \( u_d \). To locate \( u_d \), the adversary would have to sequentially compromise nodes \( IP_{d-1}, IP_{d-2}, \ldots, IP_1 \). To put it differently, before compromising node \( IP_d \), the adversary cannot distinguish \( u_d \)’s address from the addresses of all the other non-compromised nodes.

### 3.1.3 Comparison

Now we briefly compare these two techniques. The privacy-zone technique involves \( (k+1) \) unicast transmissions via the MANET substrate, so does the packet-rerouting technique. For simplicity, we assume the same communication cost to unicast a packet between any two nodes, so both techniques have the same communication overhead under the chosen parameter \( k \). The end-to-end delay of the privacy-zone technique is, however, \( k \) times less than that of the packet-rerouting technique, as packets can be almost simultaneously sent to all the nodes in the privacy zone.

In contrast, the packet-rerouting technique outperforms the privacy-zone technique in protecting \( u_s \)’s address privacy. In particular, assume that the adversary only compromises \( u_s \). If the packet-rerouting technique is used, the adversary only knows that \( u_s \)’s address is one of the \( (N-1) \) non-compromised addresses; if the privacy-zone technique is used, the adversary knows that \( u_d \)’s address must be one of the \( k \) addresses in \( \text{ZONE}_d \).

The Swarms scheme we will illustrate next is built upon the privacy-zone and packet-rerouting techniques to strike a balance between end-to-end delays and address privacy. The essential idea is to hide each principal’s address within an address block, called a Swarms, and route a packet between any two principals via a sequence of intermediate swarms. In what follows, we will detail the Swarms design. Section 3.2 presents the formation of swarms, followed by the routing process among swarms in Section 3.3. Then we illustrate the complete process of routing a packet from a source principal to a destination principal in Section 3.4. Finally, we discuss how to improve the routing efficiency.

### 3.2 Swarms Formation

We assume that the MANET is bootstrapped by a trusted authority (TA) not in the resulting network. Prior to network deployment, the TA generates an arbitrary partition \( \{U_d\}_{d=0}^{2^{\beta} - 1} \) of the principals such that

\[
\begin{align*}
U_d \cap U_c &= \emptyset, \forall b, c \\
\bigcup_{d=0}^{2^\beta - 1} U_d &= \{u_i\}_{i=1}^N,
\end{align*}
\]

where \( \beta \) is a system parameter determining the maximum number of allowable swarms. \( U_d \) is called the \( d^{th} \) swarm with an address block \( T\mathcal{P}_d \) comprising the network addresses of the principals in \( U_d \), named \( T\mathcal{P}_d := \{IP_i\}_{i=1}^{N_d} \). Apparently, we also have

\[
\begin{align*}
T\mathcal{P}_b \cap T\mathcal{P}_c &= \emptyset, \forall b, c \\
\bigcup_{d=0}^{2^\beta - 1} T\mathcal{P}_d &= T\mathcal{P} = \{IP_i\}_{i=1}^N.
\end{align*}
\]

It is worth noting that swarms are virtual: principals of the same swarm may be physically apart.

Each principal knows which swarm he belongs to and the corresponding address block, but he is blind to other principals in the same swarm, called swarm peers. Consider principal \( u_d \) as an example who belongs to \( U_d \). He only knows his affiliation with \( U_d \), as well as \( T\mathcal{P}_{u_d} \), but without any information about \( U_{\{\setminus u_d\}} \) (he does not know the IDs of swarm peers). In this way, each address cannot be linked to the corresponding principal, so each principal can protect his address privacy from swarm peers.

### 3.3 Routing among Swarms

In Swarms, the address block \( T\mathcal{P}_d \) of each \( U_d \) is only known to members of selected swarms called landmarks of \( U_d \). More specifically, suppose that swarm \( U_d \) is a landmark of \( U_c \). Each principal \( u_c \in U_c \) knows a tuple \((a, \Psi_c^a)\), where \( \Psi_c^a \) is a random \( \lambda^{th} \)-subset of \( T\mathcal{P}_c \). The impact of the system-wide parameter \( \lambda \) on the address privacy will be discussed in Section 4.2.2. Packets for principals in \( U_d \) need be routed through a node in one of \( U_d \)’s landmarks. This mimics the spy-movie scenes described earlier.

The issues we need further address include (1) how to select landmarks for each swarm \( U_d \) and (2) how to efficiently route a packet from any source swarm to any destination swarm. Regarding the first issue, it is desirable for \( U_d \) to have as few landmarks as possible so as to minimize the exposure of its address block \( T\mathcal{P}_d \). The second issue requires a small number of intermediate swarms to avoid high communication overhead. Consider a simple case that swarm \( U_{(a-1)} \mod 2^\beta \) is the single landmark of \( U_d \) for \( 0 \leq a \leq 2^\beta - 1 \).
Then each swarm has the smallest possible number of landmarks, but the number of intermediate swarms between any two swarms is $O(2^\beta)$, the largest possible value. To balance these two factors, we view each swarm as a "virtual node" over the MANET substrate and its address block as its "network address." The landmarks of each swarm are like maintaining "routing information" about that swarm. Then it is easy to draw the analogy between routing among swarms and that in P2P networks [6]. This motivates us to apply many elegant results from P2P networks research. In a P2P network with $N^\delta$ nodes, it is well-known that any two nodes can communicate in $O(\log N^\delta)$ hops (the network diameter) with each node maintaining the IP addresses about $\kappa$ other nodes, where $\kappa$ (the node degree) depends on the specific P2P architecture [6]. P2P networks based on de Bruijn graphs [5] can achieve diameter $O(\log N^\delta)$ with a constant and smallest possible $\kappa = 2$ [6]. Therefore, we decide to perform swarm routing using de Bruijn graphs [5], as doing so allows each swarm to have a small number of landmarks and communicate with any other swarm via $O(\log N^\delta)$ intermediate swarms.

Recall that each swarm index is $[0, \ldots, 2^\beta)$ and can be converted into a $\beta$-bit binary index. The de Bruijn graph in Swarms is a directed graph with $2^\beta$ nodes, each corresponding to a swarm. There are 2 outgoing and 2 incoming edges at each swarm $U_a$. In particular, if $U_a$’s binary index is $a_1 \cdots a_\beta$, then $U_a$ has an outgoing edge to swarm with binary indices $a_2 \cdots a_\beta 0$ and $a_2 \cdots a_\beta 1$. In addition, swarms with binary indices $0a_1 \cdots a_{\beta-1}$ and $1a_1 \cdots a_{\beta-1}$ each have an outgoing edge to swarm $U_b$ and are its only 2 landmarks. Fig. 2 shows a de Bruijn graph of swarms for $\beta = 3$, where swarms’ binary indices are depicted in the eclipses. As we can see, $U_a$’s landmarks are $U_b$ and $U_c$, while $U_b$ only has one landmark $U_d$ (excluding itself). Without considering the MANET substrate, shortest-path routing between any two swarms follows the approach given in [5, 6].

As an example, a packet from swarm $U_b$ to $U_d$ in Fig. 2 follows the swarm path $000 \rightarrow 001 \rightarrow 011$, while a packet from $U_b$ to $U_d$ follows $011 \rightarrow 110 \rightarrow 100 \rightarrow 001$.

### 3.4 Packet Routing and Forwarding

Our previous description about routing among swarms ignores the underlying MANET routing operations. In this section, we illustrate the complex process of routing a packet from any source principal to any destination principal. Without loss of generality, we assume that principal $u_s$ in swarm $U_0$ intends to send some packets to principal $u_d$ in swarm $U_1$. In addition, the shortest path from $U_0$ to $U_d$ consists of $l$ ($0 \leq l \leq \beta$) hops on the de Bruijn graph, denoted by $U_0 \rightarrow U_1 \rightarrow \cdots \rightarrow U_{a_d}$. Source $u_s$ does not know the exact address $IP_{a_d}$ of $u_d$, but he knows that $u_d$ is in the $\alpha^l_{a_d}$ swarm. The Swarms scheme can be built upon any MANET routing protocol, be it proactive (like OLSR [2]) or reactive (like AODV [7]). We will explain the slightly different operations needed to be taken when either approach is used. We consider the following two cases.

#### 3.4.1 Case 1: $l = 0$

This means that $u_s$ and $u_d$ are in the same swarm. Since $u_s$ knows $IP_{a_d}$ and $IP_d \in IP_{a_d}$, he multicasts to nodes in $IP_{a_d} \setminus \{IP_d\}$ the following information:

**Payload** := $(u_s, E(\{IP_d, u_s, u_d, K_{s,d}\}, E(K_{s,d}, DATA)))$,

where $K_{s,d}$ is a session key picked by $u_s$ and DATA is the information for $u_d$. We want to emphasize here that $u_d$ does not know the corresponding principals $u_{a_d} \setminus \{u_s\}$.

After finding $a_1$ (i.e., $a_1$) in the received Payload, each principal in $U_{a_1} \setminus \{u_s\}$ knows that he is a potential receiver and then attempts to decrypt $E(\{IP_d, u_s, u_d, K_{s,d}\})$ using his private key. Only $u_d$ can get a meaningful decryption result in which his ID $IP_d$ appears and then use the embedded $K_{s,d}$ to decrypt $E(K_{s,d}, DATA)$. Other principals simply dump Payload. To minimize public-key decryptions, all the principals in $U_{a_1} \setminus \{u_s\}$ buffers $E(\{IP_d, u_s, u_d, K_{s,d}\})$. If seeing it in a later Payload, they can immediately decide whether that Payload is intended for them or not without having to do a public-key decryption. In this way, each involved principal in $U_{a_1}$ merely performs one public-key decryption per communication session between $u_s$ and $u_d$.

Payload appears to be sent by node $IP_s$ from the viewpoint of principals $U_{a_1} \setminus \{u_s\}$ because they cannot link $IP_s$ to $u_s$. We also assume that effective countermeasures (such as [8]) are in place such that when observing a packet output from a node, the adversary cannot differentiate whether the observed packet was initiated or just forwarded by that node. As a result, node $IP_s$ cannot be pinpointed as the packet initiator. This is very important because otherwise $u_d$ (if malicious) may be able to link $IP_s$ to principal $u_s$, thus breaking $u_s$’s address privacy.

#### 3.4.2 Case 2: $1 \leq l \leq \beta$

Recall that $u_s$ knows a $\lambda$-subset $\Psi_u$ of $IP_{a_1}$, the address block of swarm $U_1$. What $u_s$ need do is to unicast Payload to the closest node in $\Psi_u$.

If the underlying routing protocol is proactive, node $IP_s$ always maintains a route to each node in $\Psi_u$. Assume that $a_1$-th swarm. The Swarms scheme can be built upon any MANET routing protocol, be it proactive (like OLSR [2]) or reactive (like AODV [7]). We will explain the slightly different operations needed to be taken when either approach is used. We consider the following two cases.
the route to node $IP_{r_{a_1}} \in \Psi^{a_1}$ is the shortest among all such routes. Then $u_s$ transmits $Payload$ to node $IP_{r_{a_1}}$ while blind to the corresponding principal $u_{r_{a_1}}$.

If the underlying routing protocol is reactive, node $IP_a$ first searches its routing table for the shortest route to nodes $\Psi^{a_1}$, and then delivers $Payload$ along that route. If such a route does not exist, node $IP_a$ follows the route discovery process of the routing protocol to send a special route request for nodes $\Psi^{a_1}$. Any node receiving this request sends a route reply to node $IP_a$ if knowing a valid route to any node in $\Psi^{a_1}$. Then $IP_a$ chooses the shortest one among all route replies along which $Payload$ is delivered.

Assume that in both cases node $IP_{r_{a_1}}$ is the one receiving $Payload$ from $IP_a$. It first checks the embedded $a_1$ to determine whether he is in the destination swarm. If so (i.e., $a_1 = a_i$), it passes $Payload$ to its host principal $u_{r_{a_1}}$ who will process $Payload$ as described in Case 1. In addition, node $IP_{r_{a_1}}$ need multicast $Payload$ to nodes $IP_{a_1} \setminus \{IP_{r_{a_1}}\}$, no matter whether $u_{r_{a_1}}$ is the intended destination principal $u_{a_1}$ or not. In this way, attackers cannot guess whether $IP_{r_{a_1}} = IP_a$ based on whether $IP_{r_{a_1}}$ further multicasts $Payload$.

If not in the destination swarm (i.e., $a_1 \neq a_i$), node $IP_{r_{a_1}}$ need further forward $Payload$ to the next swarm $\Psi^{a_1}$, since it knows a $\lambda$-subset $\Psi^{a_2}_{a_2} \subseteq IP_{a_2}$, it follows what node $IP_a$ did to unicast $Payload$ to the closest node in $\Psi^{a_1}$. This process continues until $Payload$ reaches some node in $IP_{a_1}$ which in turn multicasts $Payload$ to all the other nodes in $IP_{a_1}$.

Fig. 3 gives an example, where $u_s \in U_0, u_d \in U_7$, and the shortest path is $U_0 \rightarrow U_1 \rightarrow U_2$. Note that each link there is virtual and may be a multi-hop link in the MANET substrate. As we can see, node mobility may result in a different relaying point in each involved swarm, which means that packets from $u_s$ to $u_d$ may traverse physically totally different routes in the MANET substrate. Destination $u_d$ can either directly receive each packet for him or get it from some node in $U_7$.

The packet-forwarding process bears some similarity to the packet-routing technique presented earlier but with better resilience to sporadic network partitions. In particular, if any middleman in the packet-routing technique is unreachable, packets from $u_s$ cannot be successfully delivered to $u_d$. In the Swarms scheme, this is unlikely to occur as long as at least one node in each intermediate swarm is reachable.

### 3.5 Enhancing Routing Efficiency

The routing process in Section 3.4 can be further improved for better routing efficiency. As an example, node $IP_a$ delivers a packet via a chosen route to $IP_{r_{a_1}} \in \Psi^{a_1}$ which it believes to be the best route to reach nodes $\Psi^{a_1}$. However, some intermediate nodes on the chosen route may have better routes to other nodes in $\Psi^{a_1}$ than the chosen route or even to a subsequent swarm $U_i (2 \leq i \leq l)$. It is better to allow these nodes to reroute the packet to improve the routing efficiency. To make this possible, $IP_a$ needs to append $\Psi^{a_1}$ to each packet. If each address is $\delta$ bits long, the resulting packet overhead is then $|\Psi^{a_1}| \delta$ bits long. This possibly large overhead can be reduced by using hash functions or a Bloom filter [1].

Due to the space limitation, we only present the former technique here and will report the latter in a full version of this paper. For ease of illustration, we let $IP_{n_{a_1}} = IP_a$ and consider the general case that node $IP_{n_{a_1}} \in IP_{a_1}$ need deliver $Payload$ to the next swarm $U_{a_{i+1}} (0 \leq i \leq l - 1)$.

If the underlying routing protocol is reactive, the route request from node $IP_{n_{a_1}}$ contains a list $\{H(IP_{e})|IP_e \in \Psi^{a_{i+1}}\}$ of hash values instead of $\Psi^{a_{i+1}}$, where $H(\cdot)$ denotes an arbitrary good hash function. Any node receiving this request will send a route reply to node $IP_{n_{a_1}}$ if it knows a route to any address whose hash value appears in $\{H(IP_{e})|IP_e \in \Psi^{a_{i+1}}\}$.

Assume that node $IP_{n_{a_1}} \in \Psi^{a_{i+1}}$ is the destination of the best route chosen by $IP_{n_{a_1}}$ under either a proactive or reactive routing protocol. Node $IP_{n_{a_1}}$ delivers $\langle\{H(IP_{e})|IP_e \in \Psi^{a_{i+1}}\}, Payload\rangle$ along the chosen route. Each intermediate node, say $IP_{a_{i+1}}$, attempts to reroute the packet as follows.

- Node $IP_a$ first searches its routing table for the best route to each swarm $U_i (i + 1 < j \leq l)$. If multiple routes are available, the one to the largest $j$ value is chosen to bypass as many intermediate swarms as possible. This is only possible when node $IP_a$ is in one landmark of $U_i$ and thus knows a $\lambda$-subset $\Psi^{a_{i+1}}$ of $IP_{a_{i+1}}$. Then $IP_a$ sends $\langle\{H(IP_{e})|IP_e \in \Psi_{a_{i+1}}\}, Payload\rangle$ along the chosen route.

- Otherwise, $IP_a$ checks whether it has a better route to another address whose hash value is in $\{H(IP_{e})|IP_e \in \Psi_{a_{i+1}}\}$. If multiple such routes are found, the best one is chosen to reroute $\langle\{H(IP_{e})|IP_e \in \Psi_{a_{i+1}}\}, Payload\rangle$.

- If no new route is found in either case, node $IP_a$ continues forwarding $\langle\{H(IP_{e})|IP_e \in \Psi_{a_{i+1}}\}, Payload\rangle$ along the current route to node $IP_{r_{a_{i+1}}}$.

Using hash functions may unfortunately introduce false positives, which occur when an address, say $IP_a$, is erroneously considered an element of $\Psi_{a_{i+1}}$ because $H(IP_a)$ appears in $\{H(IP_{e})|IP_e \in \Psi_{a_{i+1}}\}$. False positives may occur in both route discovery and packet forwarding. Node $IP_{r_{a_{i+1}}}$ can detect the false positives in route discovery by checking whether the destinations of reported routes indeed belong to $\Psi_{a_{i+1}}$. False positives in packet forwarding may cause $\langle\{H(IP_{e})|IP_e \in \Psi_{a_{i+1}}\}, Payload\rangle$ to be rerouted to a node not in $\Psi_{a_{i+1}}$.

Now let us analyze the false positive probability. To simplify the analysis, we assume that $\{H(IP_{e})|IP_e \in \Psi^{a_{i+1}}\}$
contains $\lambda = |\Psi^{\beta+1}_u|$ hash values, each of $\mu$ bits. Then the saving in packet overhead is $\frac{T_n}{2^\mu}$, and the false positive probability is given by $\text{prob}_{FP}^{\text{hash}} = 1 - (1 - \frac{1}{2^\mu})^\lambda$. Note that a single false positive occurring in an intermediate node does not necessarily mean that Payload will not reach the next swarm. The reason is that nodes on the wrong route will also attempt to reroute Payload and may possibly “correct” the false positive.

4. SECURITY ANALYSIS

Here we first define an address-privacy metric and then use it to quantitatively measure the address privacy offered by Swarms.

4.1 Address-Privacy Metric

We use an entropy-based metric to quantify the address privacy offered by Swarms. Assume that attackers intend to find out the network address $IP_d$ of a target principal $u_d$. Let $IP$ denote $\{IP_i\}_{i=1}^N$. After obtaining some information from the network, attackers assign each address $IP_a \in IP$ a probability $p_{w_a}$ as being $IP_d$. This a-posteriori probability distribution can be described by a discrete random variable $Z_d$ with probability mass function $p_w = Pr\{Z = IP_w\}$ s.t. $\sum_{IP_w \in IP} p_w = 1$.

Definition 1. The address-privacy degree of principal $u_d$

$$AP_d = H(Z_d) = -\sum_{IP_w \in IP} p_w \log_2(p_w).$$

$AP_d$ measures the uncertainty that attackers have about which address in $IP$ is $IP_d$. One can also interpret $AP_d$ as the number of bits of additional information that attackers need in order to precisely identify $IP_d$ within $IP$. It follows that $0 \leq AP_d \leq \log_2(|IP|) - \log_2(N)$ [3]. The lower bound is achieved when $IP_d$ is assigned a probability of one and each $IP_w \in IP \setminus \{IP_d\}$ is assigned a probability of zero; the upper bound is attained when each $IP_w \in IP$ is assigned an equal probability of $1/N$, meaning that all the addresses are equally likely to be $IP_d$ as viewed by attackers (the ideal case).

4.2 Analysis

Now we use the address-privacy degree to measure the address privacy provided by Swarms. For ease of illustration, we still use the previous example in which principal $u_e \in U_{a_0}$ is allowed to send packets to $u_d \in U_{a_1}$. Also recall that the shortest path from $U_{a_0}$ to $U_{a_1}$ on the de Bruijn graph is $U_{a_0} \rightarrow U_{a_1} \rightarrow \cdots \rightarrow U_{a_1}$. Assume that principal $u_s$ is compromised so that the adversary obtains the information that principal $u_a$ belongs to swarm $U_{s_1}$. Then attackers attempt to find out the network address $IP_d$ of $u_d$ who happens to have a critical role in the MANET. We assume that attackers are smart in the sense that they will additionally compromise only nodes in the swarms along the shortest path from $U_{a_0}$ to $U_{a_1}$. In doing so, they can more quickly and surreptitiously narrow down the search of $IP_d$ to $IP_{s_1}$. For simplicity, we have the following assumptions:

- Swarms $\{U_{a_0}^{2^a-1}\}_{a=0}^L$ are of equal cardinality $L$, i.e., $|U_a| = |IP_a| = L$ for $0 \leq a \leq 2^\beta - 1$, where $\beta \geq 1$;
- $N = 2^3 \times L$ (the network size);
- Principals in any landmark swarm of any swarm $U_a$ each know a random $\lambda$-subset of $IP_a$ (cf. Section 3.3), where $1 \leq \lambda \leq |IP_a| = L$.

We assume that the adversary also knows the above system parameters. Let $C$ denote the number of nodes (including node $IP_a$) the adversary has compromised before locating $IP_d$, so we have $1 \leq C \leq L + l - 1$. There are two cases to be considered.

4.2.1 Case 1: $l = 0$

This means that $u_s$ and $u_d$ are in the same swarm. From $u_s$, attackers know the address block $IP_{a_1}$. Let $Y \subseteq IP_{a_1}$ ($|Y| = L - C$) be the set of non-promised nodes, each of which is equally likely to be $IP_d$ as viewed by attackers. All the other nodes in $IP \setminus Y$ are impossible to be $IP_d$. Therefore, attackers assign the following probabilities:

$$p_w = \begin{cases} \frac{1}{|Y|} & IP_w \in Y \\ \frac{1 - C}{|Y|} & IP_w \in IP \setminus Y \end{cases}$$

We thus have

$$AP_d = \log_2(L - C),$$

where the superscript $s$ indicates $u_s$’s correspondent $u_a$ with regard to which the address-privacy degree is analyzed.

4.2.2 Case 2: $1 \leq l \leq \beta$

This means that $u_s$ and $u_d$ are in different swarms and separated by $(l - 1)$ intermediate swarms. After compromising $u_s$, the adversary sequentially compromises one node in each intermediate swarm until compromising one in $U_{a_1}$. Then he focuses on compromising nodes in $U_{a_1} \cap IP_{a_1}$.

In particular, recall that $u_s$ knows a $\lambda$-subset $\Psi^{\beta+1}_s$ of $IP_{s_1}$. Attackers then compromise a random node $IP_i \in \Psi^{\beta+1}_s$ from which they know a subset $\Psi^{\lambda+1}_s$ of $IP_{a_1}$. They proceed to compromise a random node $IP_i \in \Psi^{\lambda+1}_s$ to obtain a subset $\Psi^{\beta+1}_a$ of $IP_{a_1}$. This process continues until attackers compromise one node in $IP_{a_1} \cap IP_{s_1}$ (or $IP_{a_1}$ when $\lambda = L$) from which they know the whole address block $IP_{s_1}$. From then on, attackers focus on compromising the rest nodes in $IP_{a_1} \setminus IP_{s_1}$ to locate $IP_d$.

If $1 \leq C \leq l - 1$, then attackers have compromised one node in each of $\{IP_{a_1}^{C-1}_{j=0}\}$ and thus known $\{IP_{a_1}^{C-1}_{j=0}\}$. None of the nodes in $\bigcup_{j=0}^{C-1} IP_{a_1}$ are likely to be $IP_{s_1}$ while all the other nodes in $IP$ are equally likely to be $IP_{s_1}$. Therefore, attackers assign the following probabilities:

$$p_w = \begin{cases} 0 & IP_w \in \bigcup_{j=0}^{C-1} IP_{a_1} \\ \frac{1}{N - \sum_{j=0}^{C-1} |IP_{a_1}|} & IP_w \in IP \setminus \bigcup_{j=0}^{C-1} IP_{a_1} \end{cases}$$

It follows that

$$AP_d = \log_2(N - CL) = \log_2(L) + \log_2(2^3 - C).$$

If $C = l$, attackers have compromised exactly one node in $IP_{a_1}$, say $IP_i$, from which they know $IP_{a_1} \setminus IP_i$ and a $\lambda$-subset $\Psi^{\lambda+1}_s$ of $IP_{s_1}$. From the attackers’ point of view, all the nodes in $\Psi^{\lambda+1}_s$ are equally likely to be $IP_i$ with probability $\frac{1}{2^\lambda}$, all the nodes in $\bigcup_{j=0}^{C-1} IP_{a_1}$ are unlikely to be $IP_{s_1}$, and the rest nodes are equally likely to be $IP_i$ with probability $\frac{1 - \lambda}{2^\lambda - \lambda - 1}$, where $X = N - \sum_{j=0}^{C-1} |IP_{a_1}| - |\Psi^{\lambda+1}_s| = (2^3 - l)L - \lambda$. That is, attackers
assign the following probabilities:

$$p_w = \begin{cases} 
\frac{0}{2} & \text{if } IP_w \in \bigcup_{u} IP_{u_i} - 1 \text{ } \forall u \in U, \\
\frac{1}{2} \lambda_{1-\lambda/L} \lambda_L & \text{if } IP_w \in \bigcup_{u} IP_{u_i} \forall u \in U, \\
0 & \text{otherwise}.
\end{cases}$$

It follows that

$$AP_d^* = \frac{1}{2} \log_2(L) + \left(1 - \frac{1}{2}\right) \log_2\left(\frac{2^\beta - 1 + \lambda}{\lambda - \lambda_L}ight)$$

$$= \log_2(L) + \left(1 - \frac{1}{2}\right) \log_2\left(\frac{2^\beta - 1 + \lambda}{\lambda - \lambda_L}ight).$$

If $l + 1 \leq C \leq L + l - 1$, attackers have compromised at least one node in $IP_{u_i}$ to know $IP_{u_i}$ and started to focus on compromising nodes in $IP_{u_i}$. Let $T \subseteq IP_{u_i}$ be the set of non-compromised nodes, where $|T| = |IP_{u_i}| - (C - l) = L - C + l$. From the viewpoint of attackers, each node in $T$ are equally likely to be $IP_{a_i}$, while all the other nodes in $IP_{a_i}$ are impossible. Therefore, attackers assign the following probabilities:

$$p_w = \begin{cases} 
\frac{1}{2} \lambda_{1-\lambda/L} & \text{if } IP_w \in T, \\
0 & \text{if } IP_w \in IP_{a_i} \setminus T.
\end{cases}$$

We thus have

$$AP_d^* = \log_2(L - C + l).$$

Note that $AP_d^* = 0$ when $C = L + l - 1$. This means that the adversary is pretty sure that the only non-compromised node in $IP_{a_i}$ is the target $IP_d$.

4.3 Discussion

To analyze the above results, we use $AP_d^{(2)}$, $AP_{d'}^{(3)}$, $AP_{d'}^{(4)}$, and $AP_{d'}^{(5)}$ to denote the address-privacy degree derived in Eqs. (2), (3), (4), and (5), respectively.

We first discuss the impact of $\lambda$. It can easily shown that $AP_{d'}^{(4)}$ monotonically decreases with $\lambda$ if $\beta$ is sufficiently large (e.g., $\beta \geq 3$). Since $0 < \lambda \leq L$, we have $\log_2(L) \leq AP_{d'}^{(4)} < \log_2(L) + \log_2(2^\beta - 1)$. It is thus wise to choose a smaller $\lambda$ to achieve better address privacy. On the other hand, a larger $\lambda$ is preferable for better routing reliability and efficiency because more candidates routes towards next swarm will be available. It is necessary to strike a good balance between them in practice.

Now we check the impact of the number $C$ of compromised nodes. Obviously, $AP_{d'}^{(2)}$ monotonically decreases with $C$. So do $AP_{d'}^{(3)}$ and $AP_{d'}^{(5)}$. In particular,

$$\log_2(L) + \log_2(2^\beta - l + 1) \leq AP_{d'}^{(3)} \leq \log_2(L) + \log_2(2^\beta - 1)$$

and

$$0 \leq AP_{d'}^{(5)} \leq \log_2(L - 1).$$

Therefore, we have $AP_{d'}^{(5)} < AP_{d'}^{(4)} < AP_{d'}^{(3)}$ and can conclude that the address privacy of $u_d$ decreases as $C$ grows, which complies with the intuition.

Now let us discuss the impact of $l$, the distance in hops between $U_{u_i}$ and $U_{a_i}$ on the de Bruijn graph. Let $l_1, l_2$ be two integers satisfying $0 \leq l_1 < l_2 \leq \beta$. We can easily verify that $AP_{d'}^{(l = l_1)} \leq AP_{d'}^{(l = l_2)}$ for any given $C \in \{1, ..., L + l_2 - 1\}$. Therefore, the address privacy of $u_d$ in $U_{u_i}$ with regard to $u_{s_i} \in U_{u_i}$ is in direct ratio to $l$. Let $U_{d(i)} \subseteq U$ be the set of principals allowed to send messages directly to $u_d$. The address privacy of $u_{d(i)}$ is determined by the nearest correspondent $u_i \in U_{d(i)}$. That is, $AP_d = \min_{u_d \in U_{d(i)}} AP_{d'}^{(l = l_1)}$. Therefore, it is necessary to put $U_{d(i)}$ in swarms as distant from $U_{u_i}$ as possible. How to allocate principals to different swarms to satisfy diverse address-privacy requirements is an open problem worthy of further study.

5. CONCLUSION

This paper introduced address privacy as a new security requirement for MANETs. We presented the Swarms scheme to prevent attackers from establishing the one-on-one mappings between network addresses and MANET principals. The security of Swarms was quantitatively evaluated using an entropy-based information-theoretic metric. As the future work, we plan to investigate tradeoffs between address privacy and communication efficiency as well as strategies to satisfy diverse address-privacy requirements of MANET principals.

6. ACKNOWLEDGMENTS

This work was supported in part by ERIF (IIT) and the US National Science Foundation under grant CNS-0716302.

7. REFERENCES


