Secure Dynamic Distributed Routing Algorithm for Ad Hoc Wireless Networks

El-Khatib, K.; Korba, Larry; Song, Ronggong; Yee, George

Publisher's version / Version de l'éditeur:

First International Workshop on Wireless Security and Privacy (WiSPr 2003) [Proceedings], 2003

National Research Council Canada
Conseil national de recherches Canada

Institute for Information Technology
Institut de technologie de l'information

NRC · CNRC

*Secure Dynamic Distributed Routing Algorithm for Ad Hoc Wireless Networks **

El-Khatib, K., Korba, L, Song, R., and Yee, G.
October 2003

* published in First International Workshop on Wireless Security and Privacy (WiSPr 2003) Kaohsiung, Taiwan. October 6, 2003. NRC 46517.

Canada

Secure Dynamic Distributed Routing Algorithm for Ad Hoc Wireless Networks

Khalil El-Khatib, Larry Korba, Ronggong Song, and George Yee
Institute for Information Technology
National Research Council of Canada
Ottawa, Ontario K1A 0R6, Canada
E-mail: { Khalil.El-Khatib, Larry.Korba, Ronggong.Song, George.Yee}@nrc.ca

Abstract

An ad hoc wireless network permits wireless mobile nodes to communicate without prior infrastructure. Due to the limited range of each wireless node, communication sessions between two nodes are usually established through a number of intermediate nodes. Unfortunately, some of these intermediate nodes might be malicious, forming a threat to the security or confidentiality of exchanged data. While data encryption can protect the content exchanged between nodes, analysis of communication patterns may reveal valuable information about end users and their relationships. Using anonymous paths for communication provides security and privacy against traffic analysis. To establish these anonymous paths, all nodes build a global view of the network by exchanging routing information. In dynamic ad hoc networks, building this global view is not an option. In this paper, we propose and analyze a distributed route construction algorithm for use in the establishment of anonymous routing paths in ad hoc wireless networks.

1: Introduction

During the past few years, innovations in radio technologies such as Bluetooth, IEEE 802.11, or Hiperlan, have created a new concept of networking called ad hoc networking. Due to the short coverage range, each node functions as both a host and a router, and shares the control of the network with all other nodes[1]; two nodes can communicate directly if they are within the radio communication range of each other, or through other mobile nodes if they are separated by more than one radio operation range diameter in a relay fashion. Figure 1 shows an ad hoc wireless network with eight nodes. The dotted circle centered on each node shows the radio communication range of the node. Any connection between node *S* and *R* in the figure has to go through other intermediate nodes.

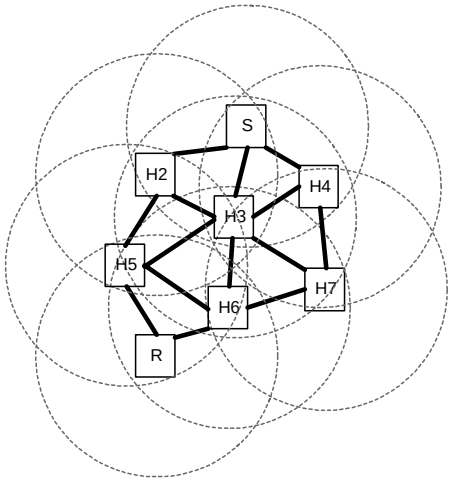

Figure 1. An ad hoc wireless network with eight hosts.

There are a number of circumstances where an ad hoc network is advantageous in the absence of network infrastructure. Examples of such circumstances include: battlefields where each soldier might be carrying a wireless transceiver; or on-the-fly conference environments where devices used by attendees collaborate to enable end-to-end communication between attendees, and perhaps Internet connection, without the requirement of a wireless infrastructure. In some environments, such as the military environment or even emergency/rescue environments, the information exchanged between two communicating parties might include highly sensitive data that must be secured when sent through intermediate nodes. While end-to-end security mechanisms can provide security for the data, valuable information regarding the nature and location of communicating entities may be determined from traffic analysis and routing data transported in the clear. Network-based anonymity techniques offer the prospect for hiding this information.

For the Internet, several network-based anonymity approaches provide anonymous communication between

[1] We will use the terms *node* and *host* interchangeably.

end-nodes. These approaches include DC-nets [1], Crowds [2], MIX networks [3], and Onion Routing [4]. Both MIX networks and Onion Routing share the same concept of establishing an anonymous path for the data transfer. To construct the anonymous path, a source node must store and maintain information about the topology of the network. Keeping up-to-date information about the topology of the network is complex in the absence of fixed infrastructure and in the presence of dynamic topology, as found in ad hoc wireless environments.

In this paper, we propose a distributed path construction protocol for anonymizing communication in wireless ad hoc networks. The protocol does not require the source node to gather and store information about the network topology. Instead, the source node initiates a broadcast message intended for a selected destination node. Intermediate nodes insert their identification (IDs) and a session key into the message and forward copies of this message to their neighbors until the message gets to its intended receiver. Intermediate nodes encrypt this information before adding it to the message, and only the intended receiver node is able to decrypt it. Once the receiver node receives the message, it retrieves from the message the information about all intermediate nodes, encapsulates this information in a multi-layered message, and sends it along a reverse path in the dissemination tree back to the source node. Each intermediate node along the reverse path removes one encrypted layer from the message, and forwards the message to its ancestor node until the message reaches the source node. When the protocol terminates, the source node ends-up with information about all the intermediate nodes on the discovered route as well as the session keys to encrypt the data transmitted through each of these nodes. The multicast mechanism and the layered encryption used in the protocol ensure the anonymity of the sender and receiver nodes.

The rest of the paper is organized as follows. Onion Routing technology is briefly reviewed in Section 2. In Section 3, we give a literature review of the techniques used to find anonymous paths in current anonymous communication systems, followed by a review of secure and non-secure routing algorithms for wireless ad hoc networks in Section 4. Our proposed distributed anonymous path computation algorithm is presented in Section 5. In Section 6, we discuss the characteristics of this algorithm. In Section 7, we analyze the algorithm for its security and anonymity. Section 8 offers our conclusion.

2: Anonymous Communication in the Onion Routing Protocol

A variety of widely known techniques or attacks may be used to infer the identities, locations, or relationships between communicating entities in a public network. Typical attacks include: message coding, timing, message volume, flooding, intersection and collusion. Onion Routing is a communication protocol that is resistance against some of these attacks.

The Onion Routing protocol employs a network of Chaum MIXes [3] to provide anonymous and secure communications. It provides a communication infrastructure that is reasonably resilient against both eavesdropping and traffic analysis. Using the protocol, communicating applications do not connect directly to each other; instead, they communicate through a sequence of networked computing nodes called onion routers. Onion routers are generally application layer routers that realize Chaum MIXes. Onion routing connections proceed in three phases: *connection setup* phase, *data transfer* phase and *connection termination* phase.

During the *connection setup* phase, an initiating application makes a socket connection to an application specific proxy on one onion router. The proxy determines the route through the onion routing network using its knowledge of the network topology and available onion routers. To protect the data and routing information, the proxy constructs a multi-layer encrypted data structure called an onion and sends it through the network. Each layer of the onion defines the next hop in the route. An onion router that receives an onion peels off the topmost layer, identifies the next hop, and sends the remaining onion to the next router. In addition to carrying next hop information, each onion layer contains key seed material from which keys are generated for decrypting and encrypting data sent forward or backward along the anonymous connection.

Once the anonymous connection is established, the *data transfer phase* begins and data can be sent in both directions. The initiator's onion proxy receives data from an application, breaks it into fixed sized cells, and encrypts each cell multiple times - once for each onion router the message traverses on its way to the destination. As a cell of data moves through the network, each onion router removes one layer of encryption and forwards the cell to the next node along the path. Eventually, the data emerges at the final onion router in the path. At this point, the recipient’s proxy regroups the cells into the data stream originally submitted by the application and, acting as the receiver proxy, forwards it to the destination application. For data moving backward, from the recipient to the initiator, the process occurs in the reverse order, with the recipient’s proxy breaking the traffic into cells, and successive onion routers encrypting the cells it for the return journey.

In the *connection termination phase*, the anonymous connection established in the *connection setup phase* is torn down. This involves the removal of encoded next hop

information in each onion router making up the connection.

3: Finding Anonymous Paths in Current Anonymous Communication Systems

Over the Internet, anonymous systems [5, 6, 7] use application level routing to provide anonymity through a fixed core set of MIXes, as we described earlier for the Onion Routing protocol. Each host keeps a global view of the network topology, and make anonymous connections through a sequence of MIXes instead of making direct socket connections to other hosts. The authors in [8] used an alternate Onion Routing approach to provide anonymous communications for mobile agents in the JADE environment. In this work, each JADE multi-agent platform in the system has several onion agents that provide an anonymous data forwarding service, and at least one onion monitor agent that keeps track of the location of all other onion agents in the system. Onion monitor agents exchange onion agent reachability information to maintain a valid topology of the onion agent network. Alternatively, Levien [9, 10] developed a monitoring utility that queries MIXes and publishes on a website the average latency and uptime for each MIX over the past 12 days. Remailer clients such as the Jack B. Nymble [11] can use the reliability statistics and keys provided on this website when selecting the MIX chain. Additional information such as configuration options and special features for each of the MIXes is also published.

Two recent works, Tarzan [12] and MorphMix [13], discuss the difficulties of constructing routes in dynamic environments. In Tarzan, the initiating node establishes the anonymous path by iteratively adding one node at a time to the path. In a single iteration, the initiator adds one node to the path, and receives the list of neighbors of that node. The initiator selects one of these neighboring nodes to be added to the path during the next iteration. A similar approach was used in MorphMiX but the difference is that in MorphMix, and instead of the initiator, a trusted third party makes the selection of the next node. Using the probability of appearance of nodes on the path, the path initiator can, up to a certain degree, determine existence of malicious collusions among the nodes on the path. The problem with Tarzan and MorphMix is that it takes a long time to construct the paths, which is a major problem for dynamic networks.

4: Routing Protocols in Ad hoc Networks

In packet-switched networks such as the Internet, routing protocols use either link-state or distance-vector routing algorithm [14]. Both algorithms allow a network node to determine the shortest distance to a destination node. Prominent examples of these are the link state protocol OSPF (Open Shortest Path First)[15] and the distance vector protocol RIP (Routing Information Protocol)[16]. While both link state and distance vector algorithms can be used in ad hoc networks, they incur high message complexity [17] as a result of node mobility. A number of routing protocols have been specifically designed to reduce this complexity in ad hoc networks. Examples of such protocols include the DSR (Dynamic Source Routing) [18] and AODV (Ad hoc On demand Distance Vector)[19].

In DSR, a source node that needs to discover a path to a destination floods the network with query packets. Each intermediate node that receives a query packet adds its identification to the list of nodes traversed so far by packet and forwards the packet to all neighboring nodes. Once the destination node receives a query packet, it answers with a packet reply that carries a copy of the route stored in the packet query message. AODV is similar in functionality to the DSR, except that intermediate nodes may cache routing information from the reply packets, and use this information later in subsequent route discovery sessions. Using cached routing information gives AODV some performance improvement over DSR. A comparative performance evaluation of DSR, AODV, and other routing protocols for ad hoc networks can be found in [20, 21].

Achieving security in wireless ad hoc networks is a complex task due to the nature of the wireless environment and the lack of infrastructure [22]. A number of protocols have been developed to add security to routing in ad hoc networks. Papadimitriou [23] proposed SRP (Secure Routing Protocol) based on DSR. The protocol assumes the existence of a security association between the source and destination to validate the integrity of a discovered route. Dahill [24] proposed the ARAN (Authenticated Routing for Ad hoc Networks) protocol that uses public key cryptography instead of the shared security association used in the SRP. Each intermediate node running the protocol verifies the integrity of the received message before forwarding it to its neighbor nodes. Source and destination nodes use certificates included in the route discovery and reply messages to authenticate each other. The protocol has an optional second discovery stage that provides non-repudiating route discovery. Yi [25] developed a generalized SAR (Security-Aware Ad-hoc Routing) protocol for discovering routes that meet a certain security criteria. The protocol requires that all nodes that meet a certain criteria share a common secret key.

Both SRP and ARAN ensure only the authenticity but not the privacy of the routing information, while SAR finds routes that meet a certain security level. In all these protocols, intermediate nodes that handle the control messages can easily find the identity of the communicating nodes, which we need to hide in case of anonymous communication. Our protocol proposed in the

next section provides anonymity for the path discovery phase (and hence for subsequent communications using this path), and uses the onion routing approach to provide protection against traffic analysis techniques.

5: Distributed Anonymous Route Discovery Protocol

In this section, we introduce our distributed protocol for establishing anonymous path in ad hoc wireless networks. The major objective of the protocol is to allow intermediate nodes to participate in the path construction protocol without jeopardizing the anonymity of the communicating nodes.

5.1: Overview

To send data anonymously to a receiver node R, a sender[2] node *S* has to discover and establish an anonymous path that connects the two nodes. Both the path discovery and establishment process should be carried out without jeopardizing the anonymity of the communicating nodes. Our approach presented here for constructing the anonymous path is inspired by the DSR [18] protocol and can replace it for anonymous communication in ad hoc networks. The process is divided into two phases: the *path discovery* phase and the *path reverse* phase. Distributed information gathering about intermediate nodes that can be used along an anonymous path is carried out during the *path discovery* phase, while passing this information to the source node takes place during the *path reverse* phase. We elaborate on these phases during the following sections, but we first introduce the notations that are used henceforth.

5.2: Notations and Terminologies

Notations and terminologies are defined as follows.

- ID_i : identity of node *i*, e.g. ID_R is the identity of the receiver *R*.
- PK_i : public key of node *i*, e.g. PK_R is the public key of the receiver *R*.
- *TPK* : a temporary one-time public key.
- *TSK* : the private (secret) key corresponding to *TPK*.
- K_i : symmetric (session) key generated by node *i*, e.g. K_S is the symmetric (session) key generated by the source node *S*.
- PL_S : the padding length set by the sender.
- P_S : a padding implemented by the sender.
- PL_R : the padding length made by the receiver node *R*.
- P_R : a padding made by the receiver node *R*.
- $E_{PK_i}(M)$: The message *M* is encrypted with a public key PK_i, e.g. *RSA*.
- $E_{K_i}(M)$: The message *M* is encrypted with the symmetric key K_i, *e.g. DES.*
- *H (M)*: The message *M* is hashed with a hash function, *e.g. MD5.*
- $Sign_S(M)$: The message *M* is signed with the source node *S*'s private key, *e.g. RSA.*
- $SN_{session\_ID_i}$: A random number generated by node ID_i for the current session.

5.3: Path Discovery Phase

The *path discovery* phase allows a source node *S* that wants to communicate securely and privately with node *R* to discover and establish a routing path through a number of intermediate wireless nodes. An important characteristic of this phase is that none of the intermediate nodes that participated in the route discovery phase can discover the identity of the sending node *S* and receiving node *R*.

The source node *S* triggers the *path discovery* phase by sending a *path discovery* message to all nodes within its wireless transmission range. The *path discovery* message has four parts. The first part is the open part. It consists of a one-time public key *TPK*, which is generated for each route discovery session and used by each intermediate node to encrypt routing information appended to the *path discovery* message. This key serves also as a unique identifier for the message. The second part contains the identifier ID_R of the intended receiver, and the symmetric key K_S generated by the source node, all encrypted with the public key PK_R of the receiver. The source node may learn about the public key PK_R of the destined receiver through a number of ways including using the service of a certificate authority (CA). The symmetric key K_S is used to encrypt the third part of the message as well as to protect against replay attacks. The third part consists of ID_S, PK_S, *TPK*, *TSK*, $SN_{Session\_ID_S}$ PL_S, P_S, and $Sign_S(M_S)$, all encrypted with K_S. The intended receiver uses the public key *TPK* and it corresponding private key *TSK* to decrypt and verify the routing information in the message. $SN_{Session\_ID_S}$ is a random number generated by the source node and is mapped to the encryption key K_S to use with which message. The padding P_S protects against message size attack, and; $Sign_S$ protects the integrity of the message. A complete analysis of the security of the protocol is provided in Section 7. The fourth part of the message holds information about intermediate nodes that

[2] We use "sender" and "source" interchangeably.

have already handled the message, as described below. Therefore, a message sent by a source node has the following format (fourth part not yet available):

$$TPK,\ E_{PK_R}(ID_R, K_S),\ E_{K_S}(ID_S, PK_S, TPK, TSK,\ SN_{Session\_ID_S}, PL_S, P_S, Sign_S(M_S))$$

where $M_S = H(TPK, TSK, ID_R, K_S, ID_S, PK_S, SN_{Session\_ID_S}, PL_S, P_S)$.

We assume that each node keeps an internal table for mapping the randomly generated number of a session to the encryption key for the session, as well as to the ancestor and successor node along the anonymous path for the session. Given an encrypted message and a randomly generated number, a node can use this mapping table to know which key to use to encrypt the message as well as the ID of the node to which to forward the message. Only the random number, session key, and ancestor node entries are added to the table during the *path discovery* phase, while the successor node entry is added later during the *path reverse* phase.

When a node *i* receives a *path discovery* message, it processes the message according to the following steps:

1. Check if the message has already been received from other nodes within its wireless transmission range using the *TPK* as the unique identifier for the message
2. If the message was received previously, drop it silently; **STOP**
3. Check if the node is the intended receiver (try to decrypt $E_{PK_R}(ID_R, K_S)$ with the private key of the node and compare the ID_R to the node's id)
4. If the node is NOT the intended receiver, then
   a. Add the following information to the message, all encrypted with the *TPK*: the id of the node, a session key K_i (shared encryption key generated by the node), a randomly generated number $SN_{Path\_ID_i}$ for the session, and the signature of the original received message.
   b. Broadcast the new message to other nodes in the wireless transmission range.
   c. Add $< SN_{Path\_ID_i}$, id of the ancestor node, $K_i >$ to the internal mapping table.
5. If the node is the destined receiver, then
   a. Use the session key K_S from $E_{PK_R}(ID_R, K_S)$ to decrypt the rest of the message and get *TSK*, then use the *TSK* to get session keys for all the nodes along the path of the message.
   b. Put all ids of the nodes and their session key in one message; encrypt the message several times, each time with the session key of a node along the path to the receiver. Use the reverse order of the keys in the message (same as the data flow in onion routing)
   c. Send the message to the first node in the reverse path

A *path discovery* message that has already traveled nodes 1..*i* on its way from the sender *S* to the receiver *R* would have the following format:

$$TPK,\ E_{PK_R}(ID_R, K_S),\ E_{K_S}(ID_S, PK_S, TPK, TSK,\ SN_{Session\_ID_S}, PL_S, P_S, Sign_S(M_S))$$
$$E_{TPK}\ (ID_1, K_1,\ SN_{Session\_ID_1},\ Sign_{ID_1}(M_{ID_1})),$$
$$E_{TPK}\ (ID_2, K_2,\ SN_{Session\_ID_2},\ Sign_{ID_2}(M_{ID_2})),$$
$$\vdots$$
$$E_{TPK}\ (ID_i, K_i,\ SN_{Session\_ID_i},\ Sign_{ID_i}(M_{ID_i}))$$

where M_S is as defined above, $M_{ID_i} = H(M_{prev}, ID_i, K_i, SN_{Path\_ID_i})$, and M_{prev} is the cumulative message that node *i* gets from its ancestor node *i-1*.

5.4: Path Reverse Phase

The *path discovery* message is forwarded from one node to the other in the network until it reaches the target receiver *R*, which triggers the *path reverse* phase.

When the intended receiver gets the *path discovery* message, it can use its private key to retrieve K_S. Then using K_S, it can obtain the temporary private (secret) key *TSK* encrypted in the third part of the message. Using *TSK*, the receiver node *R* can also retrieve the id's of all intermediate nodes and the session key to use with each one of these intermediate nodes, and the random number generated by each node. The receiver then composes a message that contains all these random numbers and the corresponding session keys, and encrypts the message with the session keys of all the nodes along the path to the source node. With each encryption, the receiver *R* adds a layer that contains the random number generated by the node along the reverse path to the sender. If the first node to get this message from the receiver is node *i*, the encrypted message constructed by the receiver *R* will have the following format:

$$E_{K_i}(...(E_{K_2}(E_{K_1}(E_{K_S}(\ SN_{Session\_ID_1}, K_1,\ SN_{Session\_ID_2}, K_2, \ldots, K_i,\ SN_{Session\_ID_R}, PL_R, P_R), SN_{Session\_ID_S}), SN_{Session\_ID_1}), SN_{Session\_ID_2}), \ldots), SN_{Session\_ID_{i-1}}),\ SN_{Session\_ID_i}$$

Each intermediate node that receives the *path reverse* message uses the $SN_{Session\_ID_i}$ to retrieve the key for the session, removes one encryption layer and forwards the message to the next node on the reverse path to the source node. The ID of the node from which the message was

received is added to the successor node entry corresponding to the random number into the mapping table.

When the source node receives the message, it decrypts the message and passes the information about all the intermediate nodes (i.e. the route) to the higher application.

5.5: Data Transfer Phase

The *path reverse* message is forwarded from node to node along the path that the receiver chooses, until it reaches the original sender *S*, which triggers the *data transfer* phase.

After the sender gets the *path reverse* message, it first checks whether or not the message is correct, and then uses the shared session keys of the intermediate nodes to make the following layer encryption for the data (Data), which the sender wants to transfer to the receiver.

$$E_{K_1}(E_{K_2}(...(E_{K_{i-1}}(E_{K_i}(E_{K_S}(Data_S), SN_{Session\_ID_R}), SN_{Session\_ID_i})\ldots), SN_{Session\_ID_2}), SN_{Session\_ID_1}$$

Each intermediate node just decrypts one encryption layer and forwards the message to the next node according to the id of the next node.

The receiver also can use the following layer encryption to send Data to the sender.

$$E_{K_i}(...(E_{K_2}(E_{K_1}(E_{K_S}(Data_R), SN_{Session\_ID_S}), SN_{Session\_ID_1}), SN_{Session\_ID_2}),\ldots \ldots), SN_{Session\_ID_{i-1}}), SN_{Session\_ID_i}$$

6: Protocol Characteristics

6.1 : Non Source-Based Routing

The proposed protocol has a major advantage over the standard routing protocol for anonymous communication systems in the way routing paths are constructed in that it does not require a global view of the network topology. In standard onion routing, the source onion node must know in advance the topology and link state of the network before it can establish a routing path. The source onion node must also know the public keys of all onion nodes on the path as well as the exit policies for the edge onion nodes. In the proposed protocol, each node in the network contributes toward the final routing path by forwarding the *path discovery* and *path reverse* messages. This approach eliminates the need for managing routing centrally.

Moreover, similar to other dynamic routing protocols, the proposed protocol gathers routing information only when the session is started or when the path breaks. Information about nodes that have joined or left the ad hoc network need not be propagated to all nodes. In standard onion routing this information is needed so that source onion nodes can build a routing path with viable nodes.

On the downside, the proposed approach takes away from the source node the ability of choosing the routing path according to certain criteria. For instance, the source onion node cannot specify its preferences for routing, such as trying to direct the routing path away from certain hazardous domains. As well, the source onion node cannot set a limit on the maximum number of nodes on the path. A large number of nodes on the routing path can render the path too slow for real-time interactive applications. These applications usually have an upper limit on the network delay, which limits the number of intermediate onion nodes. In addition, the encryption/decryption at each node requires a fair amount of computational power to be viable. Current wireless nodes may not have sufficient computational capability for such processing.

6.2: Resilience Against Path Hijacking

During the *path discovery* phase, each intermediate node receives a *path discovery* message and forwards it to all nodes within its wireless transmission range. While a well-behaved node forwards the message to all nodes in an unbiased way, a malicious node might forward the message only to other neighboring malicious nodes, resulting in a path with only malicious nodes. We refer to this situation as “path hijacking”.

The proposed protocol proves to be resilient against path hijacking. To confirm that, note that the protocol terminates successfully only after the trusted intended receiver triggers the *path reverse* phase, and after the *path reverse* message has made its way successfully to the source onion node. If malicious nodes keep on forwarding a *path discovery* message among themselves, the message will never get to the intended receiver and the source onion node will never get a *path reverse* message triggered by the *path discovery* message. Although in this case, the protocol may fail to return a suitable path, it is still resilient to path hijacking in the sense that the actual hijacking does not occur (Note that other *path discovery* messages might still have made their way to the intended receiver and triggered a successful *path reverse* phase). If on the other hand, a malicious node decided to break the cycle and forward the message to a non-malicious node, and if this message gets to the intended receiver and initiates a *path reverse* message, the path will be constructed through a number of malicious nodes. But this case does not threaten the anonymity of the data traffic as was shown in [26], although it is a partial path hijacking.

In any case, we only claim resilience to path hijacking, not immunity to it.

7: Security and Anonymity Analysis

In this section, we examine how the protocol provides an adequate level of security and anonymity for the sender and receiver during path establishment.

7.1: Security of the Algorithm

The proposed protocol uses two types of messages: the *path discovery* message and the *path reverse* message. In the following sections, we will demonstrate that the protocol we proposed here is secured against passive and active attacks, but not against Denial-of-Service attacks.

7.1.1: Passive Attacks

In the *path discovery* message, the identifier ID_S and public key PK_S of the sender are encrypted with a one-time session key K_S. This key and the identifier ID_R of the intended receiver are encrypted with the public key PK_R of the intended receiver. The one-time public key encrypts the identities of intermediate nodes and the shared session keys. Thus, an adversary cannot find the real sender, receiver, and all intermediate nodes just by looking at the *path discovery* message. The same conclusions can be made for the *path reverse* message.

7.1.2: Active Attacks

Our protocol provides protection against replay and modification attacks. Using the session key K_S in the *path discovery* message, and the one-time public key *TPK*, the sender can discover a replayed *path discovery* message. Additionally, if some adversaries want to change the *path discovery* message or impersonate the sender or some intermediate nodes, the receiver can easily find out by verifying the signature since the sender and intermediate nodes have hashed the message, which is cumulative from its ancestor node, and signed the hashing value in the *path discovery* message.

7.1.3: Denial-of-Service Attack

A Denial-of-Service (DoS) attack would be a very dangerous attack on the algorithm. The algorithm itself does not provide a mechanism against this kind of attack. For instance a powerful adversary may simply flood the network with path discovery messages. Additionally, the small computational power on all wireless carry-on devices makes the protocol more vulnerable to this attack. This problem is common though to all routing protocols in ad hoc networks.

7.2: Anonymity of the Sender and Receiver

During *path discovery* phase processing, we have the following cases for the analysis of the anonymity of the sender and receiver.

1. If all neighboring nodes of the sender joined in collusion, they would know which message originally came from the sender and which message was just forwarded by the sender, i.e., they would find the sender but will not know who is the intended receiver.
2. If all neighboring nodes of the receiver were in collusion together, they would know which message terminated in the receiver and which message was just forwarded by the receiver, i.e., they would find the potential receiver but would not know the identity of the sender.
3. If some intermediate nodes were in collusion together, they would only know that the message was forwarded. They therefore cannot confirm which node is the sender and which node is the receiver.

During *path reverse* phase processing, we have the following additional cases for the analysis of the anonymity of the sender and receiver:

1. If all neighboring nodes of the receiver joined in collusion, they would be able to determine who is the receiver. The collusion of all neighboring nodes can reveal the fact that the circled node is the node that started the *path reverse* phase, and hence it must be the intended receiver in the *path discovery* phase.
2. If all neighboring nodes of the sender were in collusion, they would be able to determine who is the sender. The situation is same as the above.
3. If some or all of the intermediate nodes were in collusion together, although they would know part of the path chain, they still would not be sure who the sender and receiver are since they would not know if the end node of the *path reverse* message is the sender or just another intermediate node, and the start node of the *path reverse* message is the receiver or just another intermediate node.

8: Conclusion

Privacy is a key issue in wireless ad hoc networking. The protocol presented here creates routes dynamically to support onion routing without the originator knowing the keys of the mix nodes or the topology of the network. We have shown that the protocol provides an adequate level of security and anonymity for the sender and receiver during path establishment. The benefits of this approach include:

- Non-source-based routing – source node does not need to know global topology and link availability; route computation shared among many nodes; easy adaptability to changes in network topology;

- Resilience against path hijacking – resilience against malicious nodes compromising the communication through collusion.

Some disadvantages to the protocol include the inability to direct the route away from hazardous or failed nodes, the inability to limit the number of nodes in the route, and the need for fairly large computation power and storage requirements for participating nodes. With our approach, we trade the overhead of constantly maintaining the wireless routing nodes in an ad hoc network for the overhead of discovering paths whenever a new connection must be made, or an existing one is broken due to loss of a node in the routing chain. This is accomplished at the expense of all nodes having to perform calculations required to discover routes.

We are continuing our research to look at possible solutions to these disadvantages. For example, the first disadvantage might be overcome by modifying the protocol to only send discovery messages to nodes that are reputable or healthy (assumes a way of looking up this information). The third disadvantage will be overcome with the availability of more powerful wireless nodes.

Reference

1. D. Chaum: The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. Journal of Cryptography, vol. 1, no. 1, pages 65-75, 1988.
2. M. K. Reiter and A. D. Rubin: Crowds: Anonymity for Web Transactions. ACM Transactions on Information and System Security, vol. 1, no. 1, pages 66-92, Nov. 1998.
3. D. Chaum: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM, vol. 24, no. 2, pages 84-88, Feb. 1981.
4. M. Reed, P. Syverson, and D. Goldschlag: Proxies for anonymous routing. In 12th Annual Computer Security Applications Conference, pages 95-104. IEEE, Dec. 1995.
5. ELECTRONIC FRONTIERS GEORGIA (EFGA). Anonymous remailer information. http://anon.efga.org/Remailers/.
6. I. Goldberg, and A. Shostack: Freedom network 1.0 architecture, November 1999.
7. P. F. Syverson, D. M. Goldschlag, and M. G. Reed: Anonymous connections and onion routing. In Proceedings of the IEEE Symposium on Security and Privacy (Oakland, California, May1997), pp. 44–54
8. L. Korba, R. Song, and G. Yee: Anonymous Communications for Mobile Agents. MATA 2002: 171-181
9. http://www.sendfakemail.com/~raph/remailer-list.html
10. http://www2.pro-ns.net/~crypto/chapter8.html
11. http://www.theinternet.cc/potatoware/jbn2/index.html
12. M. J. Freedman, R. Morris: Tarzan: A peer-to-peer anonymizing network layer. In Proceedings of the First International Workshop on Peer-to-Peer Systems (Cambridge, MA, Mar. 2002).
13. M. Rennhard. MorphMix: Peer-to-Peer based Anonymous Internet Usage with Collusion Detection. Technical Report Nr. 147, TIK, ETH Zurich, Switzerland, August 2002
14. S. Keshav: An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network. Chapter 11. Addison-Wesley, 1997.
15. J. Moy: OSPF version 2. RFC 1247, July 1991.
16. C. Hedrick: Routing information protocol. RFC 1058, June 1988.
17. M. Corson, S. Batsell, and J. Macker: Architectural considerations for mobile mesh networking. http://tonnant.itd.nrl.navy.mil/mmnet/mmnetRFC.txt, May 1996. Request for Comments Draft.
18. D. Johnson and D. Maltz: Dynamic source routing in ad hoc wireless networks. In T. Imielinski and H. Korth, editors, Mobile computing. Kluwer Academic, 1996.
19. C. Perkins: Ad hoc on demand distance vector (AODV) routing. http://www.ietf.org/internet-drafts/ draft-ietf-manet-aodv-00.txt, 1997. IETF Internet Draft.
20. J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu and J. Jetcheva: A performance comparison of multi-hop wireless ad hoc network routing protocols. In Proceedings of MOBICOM ’98 (1998).
21. S.R. Das, R. Castaneda, J. Yan and R. Sengupta: Comparative performance evaluation of routing protocols for mobile, ad hoc networks. In Proceedings of IEEE IC3N ’98 (1998).
22. J. Lundberg: Routing Security in Ad Hoc Networks. http://citeseer.nj.nec.com/400961.html
23. P. Papadimitratos and Z. J. Haas: Secure Routing for Mobile Ad hoc Networks. SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002), San Antonio, TX, January 27-31, 2002.
24. K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer: A Secure Routing Protocol for Ad Hoc Networks, In Proceedings of 2002 IEEE International Conference on Network Protocols (ICNP). November 2002.
25. S. Yi, P. Naldurg, and R.Kravets: Security-Aware Ad Hoc Routing Protocol for Wireless Networks. The 6th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI 2002), 2002.
26. J. Raymond: Traffic Analysis: Protocols, Attacks, Design Issues, and Open Problems. In H.Federrath, editor, Anonymity 2000, Volume 2009 of Lecture Notes in Computer Science, pages 10-29, Springer-Verlag, 2000.