A Secure Routing Protocol for Ad Hoc Networks[*]

Kimaya Sanzgiri[*] Bridget Dahill[†]
Brian Neil Levine[†] Clay Shields[‡] Elizabeth M. Belding-Royer[*]

[*] Dept. of Computer Science, University of California, Santa Barbara, CA 93106

[†] Dept. of Computer Science, University of Massachusetts, Amherst, MA 01060

[‡] Dept. of Computer Science, Georgetown University, Washington, DC 20007

Abstract

Most recent ad hoc network research has focused on providing routing services without considering security. In this paper, we detail security threats against ad hoc routing protocols, specifically examining AODV and DSR. In light of these threats, we identify three different environments with distinct security requirements. We propose a solution to one, the managed-open scenario where no network infrastructure is pre-deployed, but a small amount of prior security coordination is expected. Our protocol, ARAN, is based on certificates and successfully defeats all identified attacks.

1 Introduction

Ad hoc wireless networks assume no pre-deployed infrastructure is available for routing packets end-to-end in a network, and instead rely on intermediary peers. Securing ad hoc routing presents challenges because each user brings to the network their own mobile unit, without the centralized policy or control of a traditional network. Many ad hoc routing protocols have been proposed previously [9, 12, 13, 14, 15, 3], but none of the proposals have defined security requirements, and all inherently trust all participants.

In this paper, *we demonstrate exploits that are possible against ad hoc routing protocols, define various security environments, and offer a secure solution with an authenticated routing protocol.* We detail the exploits against two protocols that are under consideration by the IETF for standardization: the Ad hoc On-demand Distance Vector routing protocol (AODV) [15] and the Dynamic Source Routing protocol (DSR) [9]. AODV and DSR are efficient in terms of network performance, but they allow attackers to easily advertise falsified route information, to redirect routes, and to launch denial-of-service attacks.

Our proposed protocol, Authenticated Routing for Ad hoc Networks (ARAN), detects and protects against malicious actions by third parties and peers in one particular ad hoc environment. ARAN introduces *authentication*, *message integrity*, and *non-repudiation* to an ad hoc environment as a part of a minimal security policy. Our evaluations show ARAN has minimal performance costs for the increased security in terms of processing and networking overhead.

This paper is organized as follows. Section 2 is an overview of recent work on ad hoc routing protocols. Section 3 presents the security exploits possible in ad hoc routing protocols. Section 4 defines three ad hoc environments and the security requirements of any ad hoc network. Section 5 presents the secure ad hoc routing protocol, ARAN. Section 6 shows the results of security and network performance analyses of ARAN, and Section 7 offers concluding remarks.

2 Background

An ad hoc network forms when a collection of mobile nodes join together and create a network by agreeing to route messages for each other. There is no shared infrastructure in an ad hoc network, such as centralized routers or defined administrative policy. All proposed protocols [9, 12, 13, 14, 15] have security vulnerabilities and exposures that easily allow for routing attacks. While these vulnerabilities are common to many protocols, in this paper we focus on two protocols that are under consideration by the IETF for standardization: AODV and DSR [15, 9].

The fundamental differences between ad hoc networks and standard IP networks necessitate the development of new security services. In particular, the measures proposed for IPSec [7] help only in end-to-end authentication and security between two network entities that already have routing between them; IPSec does not secure the routing protocol.

This point has been recognized by others. Zhou and Haas have proposed a using threshold cryptography for providing security to the network [22]. Hubaux, et al. have proposed a method that is designed to ensure equal participation among members of the ad hoc group, and that gives each node the authority to issue certificates [8]. Kong, et al. [10] have proposed a secure ad hoc routing protocol based on secret sharing; unfortunately, this protocol is based on erroneous assumptions, e.g., that each node cannot impersonate the MAC address of multiple other nodes. Yi, et al. also have proposed a general framework for secure ad hoc routing [21].

[*]Supported in part by National Science Foundation awards ANI-522564 and EIA-0080199, and in part by U.S. Department of Justice, Office of Justice Programs grant 2000-DT-CX-K001. Contents are solely the responsibility of the authors and do not necessarily represent the official views of the DoJ.

Attack	AODV	DSR	ARAN
Remote redirection			
modif. of seq. numbers	Yes	No	No
modif. of hop counts	Yes	No	No
modif. of source routes	No	Yes	No
tunneling	Yes	Yes	Yes, but only to lengthen path
Spoofing	Yes	Yes	No
Fabrication			
fabr. of error messages	Yes	Yes	Yes, but non-repudiable
fabr. of source routes (cache poisoning)	No	Yes	No

Table 1. Vulnerabilities of AODV and DSR.

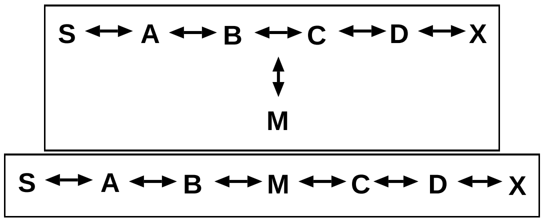


Figure 1. (A) A simple ad hoc network. (B) Another example ad hoc network.

3 Exploits allowed by existing protocols

The current proposed routing protocols for ad hoc wireless networks allow for many different types of attacks. Analogous exploits exist in wired networks [20], but are more easily defended against by infrastructure present in a wired network. In this section, we classify *modification*, *impersonation*, and *fabrication* exploits against ad hoc routing protocols. In Section 5, we propose a protocol not exploitable in these ways.

Our focus is on vulnerabilities and exposures that result from the specification of the ad hoc routing protocol, and not from problems with IEEE 802.11 [2, 4, 18]. Additionally, trivial denial-of-service attacks based on interception and non-cooperation are possible in all ad hoc routing protocols. While these attacks are possible, they are not achieved through subversion of the routing protocol.

The attacks presented below are described in terms of the AODV and DSR protocols, which we use as representatives of ad hoc on-demand protocols. Table 1 provides a summary of each protocol's vulnerability to the following exploits.

3.1 Attacks Using Modification

Malicious nodes can cause redirection of network traffic and DoS attacks by altering control message fields or by forwarding routing messages with falsified values. For example, in the network illustrated in Fig. 1a, a malicious node M could keep traffic from reaching X by consistently advertising to B a shorter route to X than the route to X that C advertises. Below are detailed several of the attacks that can occur if particular fields of routing messages in specific routing protocols are altered or falsified.

3.1.1 Redirection by modified route sequence numbers

Protocols such as AODV and DSDV [14] instantiate and maintain routes by assigning monotonically increasing sequence numbers to routes toward specific destinations. In AODV, any node may divert traffic through itself by advertising a route to a node with a *destination_ sequence_num* greater than the authentic value. Fig. 1b illustrates an example ad hoc network. Suppose a malicious node, M, receives the RREQ that originated from S for destination X after it is re-broadcast by B during route discovery. M redirects traffic toward itself by unicasting to B an RREP containing a much higher *destination_sequence_num* for X than the value last advertised by X.

Eventually, the RREQ broadcast by B will reach a node with a valid route to X and a valid RREP will be unicast back toward S. However, at that point B will have already received the false RREP from M. If the *destination_sequence_num* for X that M used in the false RREP is higher than the *destination_sequence_num* for X in the valid RREP, B will drop the valid RREP, thinking that the valid route is stale. All subsequent traffic destined for X that travels through B will be directed toward M. The situation will not be corrected until either a legitimate RREQ or a legitimate RREP with a *destination_sequence_num* for X higher than that of M's false RREP enters the network.

3.1.2 Redirection with modified hop counts

A redirection attack is possible by modification of the hop count field in route discovery messages. When routing decisions cannot be made by other metrics, AODV uses the hop count field to determine a shortest path. In AODV, malicious nodes can increase the chances they are included on a newly created route by resetting the hop count field of the RREQ to zero. Similarly, by setting the hop count field of the RREQ to infinity, created routes will tend to not include the malicious node. Such an attack is most threatening when combined with spoofing, as detailed in Section 3.2.

3.1.3 Denial-of-service with modified source routes

DSR utilizes source routes, thereby explicitly stating routes in data packets. These routes lack any integrity checks and a simple denial-of-service attack can be launched in DSR by altering the source routes in packet headers.

Assume a shortest path exists from S to X as in Fig. 1b. Also assume that C and X cannot hear each other, that nodes B and C cannot hear each other, and that M is a malicious node attempting a denial-of-service attack. Suppose S wishes

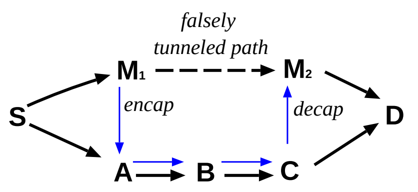


Figure 2. Path lengths spoofed by tunneling.

to communicate with X and that S has an unexpired route to X in its route cache. S transmits a data packet toward X, with the source route $S \rightarrow A \rightarrow B \rightarrow M \rightarrow C \rightarrow D \rightarrow X$ contained in the packet's header. When M receives the packet, it can alter the source route in the packet's header, such as deleting D from the source route. Consequently, when C receives the altered packet, it attempts to forward the packet to X. Since X cannot hear C, the transmission is unsuccessful.

DSR provides a route maintenance mechanism such that a node forwarding a packet is responsible for confirming that the packet has been received by the next hop along the path. If no confirmation of receipt is received after retransmitting the packet a specified maximum number of attempts, this node should return a route error message to the source node. In this case, C would send a route error message to S. Since M would be the first hop the route error takes on its path back to S, M can continue the denial-of-service attack by dropping this route error message.

DSR implements another route maintenance mechanism called *route salvaging* to recover from broken links along a path. When a break occurs, the node immediately upstream can check its route cache, and if it has a different route to that destination, it can use that route instead. In the example C would check its route cache for an alternate route. If C only knows of the erroneous route to X, the DoS attack can be completed.

Modifications to source routes in DSR may also include the introduction of loops in the specified path. Although DSR prevents looping during the route discovery process, there are insufficient safeguards to prevent the insertion of loops into a source route after a route has been salvaged[1].

3.1.4 Tunneling

Ad hoc networks have an implicit assumption that any node can be located adjacent to any other node. A *tunneling* attack is where two or more nodes may collaborate to encapsulate and exchange messages between them along existing data routes. One vulnerability is that two such nodes may collaborate to falsely represent the length of available paths by encapsulating and tunneling between them legitimate routing messages generated by other nodes. In this case, tunneling prevents honest intermediate nodes from correctly incrementing the metric used to measure path lengths.

Fig. 2 illustrates such an attack where M_1 and M_2 are malicious nodes collaborating to misrepresent available path lengths by tunneling route request packets (e.g., an RREQ in AODV). Solid lines denote actual paths between nodes, the thin line denotes the tunnel, and the dotted line denotes the path that M_1 and M_2 falsely claim is between them. Node S wishes to form a route to D and initiates route discovery.

When M_1 receives a RREQ from S, M_1 encapsulates the RREQ and tunnels it to M_2 through an existing data route, in this case $\{M_1 \rightarrow A \rightarrow B \rightarrow C \rightarrow M_2\}$. When M_2 receives the encapsulated RREQ, it forwards the RREQ on to D as if it had only traveled $\{S \rightarrow M_1 \rightarrow M_2 \rightarrow D\}$. Neither M_1 nor M_2 update the packet header to reflect that the RREQ also traveled the path $\{A \rightarrow B \rightarrow C\}$. After route discovery it appears to the destination that there are two routes from S of unequal length: $\{S \rightarrow A \rightarrow B \rightarrow C \rightarrow D\}$; and $\{S \rightarrow M_1 \rightarrow M_2 \rightarrow D\}$. If M_2 tunnels the RREP back to M_1, S would falsely consider the path to D via M_1 a better choice (in terms of path length) than the path to D via A.

Similarly, tunneling attacks are also a security threat to *multipath* routing protocols, which look for maximally disjoint paths [11]. In Fig. 2, two malicious nodes M_1 and M_2 may collaborate to tunnel routing messages to one another so that D falsely believes that the shortest route from S is $\{S \rightarrow M_1 \rightarrow M_2 \rightarrow D\}$, as in the above attack. The paths $\{S \rightarrow A \rightarrow B \rightarrow C \rightarrow D\}$ and $\{S \rightarrow M_1 \rightarrow M_2 \rightarrow D\}$ would appear completely disjoint, but actually share three common intermediate nodes, A, B, and C.

It is difficult to guarantee the integrity of path lengths with metrics like hop count. If route instantiation is determined by metrics that are governed solely by the operation of the routing protocol (such as a hop count metric), tunneling can cause routing metrics to be misrepresented. Only an unalterable physical metric such as time delay can provide a dependable measure of path length. Specifically, a secure protocol must regard as the shortest path, the path that had the shortest delay of routing messages.

3.2 Attacks Using Impersonation

Spoofing occurs when a node misrepresents its identity in the network, such as by altering its MAC or IP address in outgoing packets, and is readily combined with modification attacks. The following example illustrates how an impersonation attack can work in AODV. Similar attacks are possible in DSR (see Table 1).

3.2.1 Forming Loops by Spoofing

Assume a path exists between the five nodes illustrated in Fig. 3a toward some remote destination, X, as would follow after an AODV RREQ/RREP exchange. In this example, A can hear B and D; B can hear A and C; D can hear A and C;

[1]There is also a potential for loops to form during route salvaging. An intermediate node salvaging the path replaces the source route in the packet with a new route from its route cache. DSR prevents infinite looping in this case by allowing a packet to only be salvaged a finite number of times.

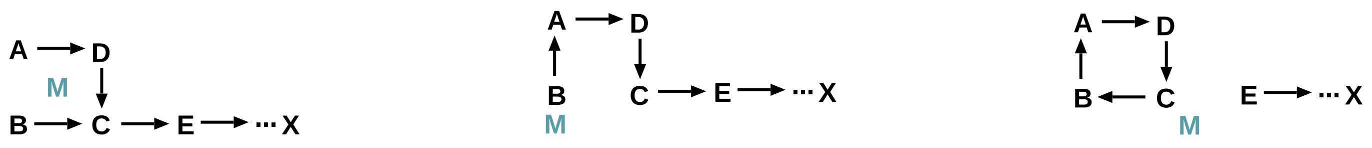


Figure 3. A sequence of events that form loops by spoofing of packets.

and C can hear B, D, and E. M can hear A, B, C, and D. E can hear C and the next hop on the path toward X.

A malicious attacker, M, can learn this topology by listening to the RREQ/RREP exchanges during route discovery. M can then form a routing loop so that none of the four nodes can reach the destination. To start the attack, M changes its MAC address to match A's, moves closer to B and out of the range of A. It then sends an RREP to B that contains a hop count to X that is less than the one sent by C, e.g., zero. B therefore changes its route to the destination, X, to go through A, as illustrated in Fig. 3b. M then changes its MAC address to match B's, moves closer to C and out of range of B, and then sends to C an RREP with a hop-count to X lower than what was advertised by E. C then routes to X through B, as shown in Fig. 3c. At this point a loop is formed and X is unreachable from the four nodes. The attack is possible with a single malicious attacker, however, multiple attackers may collaborate for the same result.

3.3 Attacks Using Fabrication

The generation of false routing messages can be classified as fabrication attacks. Such attacks can be difficult to verify as invalid constructs, especially in the case of fabricated error messages that claim a neighbor cannot be contacted.

3.3.1 Falsifying Route Errors in AODV and DSR

AODV and DSR implement path maintenance to recover broken paths when nodes move. If the source node moves and the route is still needed, route discovery is re-initiated with a new route request message. If the destination node or an intermediate node along an active path moves, the node upstream of the link break broadcasts a *route error* message to all active upstream neighbors. The node also invalidates the route for this destination in its routing table[2].

The vulnerability is that routing attacks can be launched by sending false route error messages. Suppose node S has a route to node X via nodes A, B, C, and D, as in Fig. 1. A malicious node M can launch a denial-of-service attack against X by continually sending route error messages to B spoofing node C, indicating a broken link between nodes C and X. B receives the spoofed route error message thinking that it came from C. B deletes its routing table entry for X and forwards the route error message on to A, who then also deletes its routing table entry. If M listens and broadcasts spoofed route error messages whenever a route is established from S to X, M can successfully prevent communications between S and X.

[2]In DSR the source route is removed from the node's route cache.

3.3.2 Route Cache Poisoning in DSR

Corrupting routing state is a passive attack against routing integrity. This occurs when information stored in routing tables at routers is either deleted, altered or injected with false information. Wired networks have been vulnerable to similar attacks [16, 19] but can often be defended against by security measures at routers.

Poisoning of route caches is a common example of this attack. The following details such an attack in DSR. In addition to learning routes from headers of packets that a node is processing along a path, routes in DSR may also be learned from promiscuously received packets. A node overhearing any packet may add the routing information contained in that packet's header to its own route cache, even if that node is not on the path from source to destination. For example, in Fig. 1 a path exists from node S to node X via nodes A, B, C and D. If a packet traveling along the source route from S to X is overheard by another node, that node may then add the route $<S,A,B,C,D,X>$ to its route cache.

The vulnerability is that an attacker could easily exploit this method of learning routes and poison route caches. Suppose a malicious node M wanted to poison routes to node X. If M were to broadcast spoofed packets with source routes to X via itself, neighboring nodes that overhear the packet transmission may add the route to their route cache. Since this route discovery feature of caching overheard routing information is optional in DSR, this exploit can be easily patched by disabling this feature in the network. The downside of this is that without this feature DSR operates at a loss in efficiency.

4 Security Requirements of Ad hoc Networks

A good secure routing algorithm prevents each of the exploits presented in Section 3; it must ensure that no node can prevent successful route discovery and maintenance between any other nodes other than by non-participation. In sum, all secure ad hoc routing protocols must satisfy the following requirements to ensure that path discovery from source to destination functions correctly in the presence of malicious adversaries: (1) Route signaling cannot be spoofed; (2) Fabricated routing messages cannot be injected into the network; (3) Routing messages cannot be altered in transit, except according to the normal functionality of the routing protocol; (4) Routing loops cannot be formed through malicious action; (5) Routes cannot be redirected from the shortest path by malicious action.

The above requirements comprise the security needs of an *open* environments. The following additional requirement distinguishes a *managed open* environment: (6) Unauthorized nodes should be excluded from route computation and discovery. This requirement does not preclude the fact that authenticated peers may act maliciously as well. Additionally, we assume that the managed-open environment has the opportunity for pre-deployment or exchange of public keys, session keys, or certificates.

We define a *managed hostile* environment to have requirements listed above as well as the following: (7) The network topology must not be exposed neither to adversaries nor to authorized nodes by the routing messages. Exposure of the network topology may be an advantage for adversaries trying to destroy or capture nodes.

5 Authenticated Routing for Ad hoc Networks

ARAN makes use of cryptographic certificates to offer routing security. Such certificates are already seeing deployment as part of one-hop 802.11 networks; this is the case on the UMass campus, where an 802.11 VPN is deployed and certificates are carried by nodes.

ARAN consists of a preliminary certification process followed by a route instantiation process that guarantees end-to-end authentication. The protocol is simple compared to most non-secured ad hoc routing protocols. It should be noted that the exploits listed in Section 3 are primarily due to the optimizations that have been introduced into ad hoc routing protocols for route computation and creation. Route discovery in ARAN is accomplished by a broadcast route discovery message from a source node which is replied to unicast by the destination node, such that the routing messages are authenticated at each hop from source to destination, as well as on the reverse path from the destination to the source.

5.1.3 Certification

ARAN requires the use of a trusted certificate server T, whose public key is known to all valid nodes. Keys are a priori generated and exchanged through an existing, perhaps out of band, relationship between T and each node. Before entering the ad hoc network, each node must request a certificate from T. Each node receives exactly one certificate after securely authenticating their identity to T. The methods for secure authentication to the certificate server are left to the developers. Details of how certificates are revoked are explained below in Section 5.4. A node A receives a certificate from T as follows:

$$T \rightarrow A : \text{cert}_A = [IP_A, K_{A+}, t, e]K_{T-} \tag{1}$$

The certificate contains the IP address of A, the public key of A, a timestamp t of when the certificate was created, and a time e at which the certificate expires. Fig. 4 summarizes our notation. These variables are concatenated and signed by T. All nodes must maintain fresh certificates with the trusted server. Nodes use these certificates to authenticate themselves to other nodes during the exchange of routing messages.

5.1.4 Authenticated Route Discovery

The goal of end-to-end authentication is for the source to verify that the intended destination was reached. In this process, the source trusts the destination to chose the return path.

Source node, A, begins route instantiation to destination X by broadcasting to its neighbors a *route discovery packet* (RDP):

$$A \rightarrow \text{brdcast} : [\text{RDP}, \text{IP}_X, \text{cert}_A, N_A, t]K_{A-} \tag{2}$$

The RDP includes a packet type identifier ("RDP"), the IP address of the destination (IP_X), A's certificate (cert_A), a nonce N_A, and the current time t, all signed with A's private key. Each time A performs route discovery, it monotonically increases the nonce. The nonce and timestamp are used in conjunction with each other to allow for ease of nonce recycling. The nonce is made large enough that it will not need to be recycled within the probable clock skew between receivers. Other nodes then store the nonce they have last seen for a particular node along with its timestamp. If a nonce later re-appears in a valid packet that has a later timestamp, the nonce is assumed to have wrapped around, and is therefore accepted. Note that a hop count is not included with the message.

When a node receives an RDP message, it sets up a reverse path back to the source by recording the neighbor from which it received the RDP. This is in anticipation of eventually receiving a reply message that it will need to forward back to the source. The receiving node uses A's public key, which it extracts from A's certificate, to validate the signature and verify that A's certificate has not expired. The receiving node also checks the (N_A, IP_A) tuple to verify that it has not already processed this RDP. Nodes do not forward messages for which they have already seen the tuple; otherwise, the node signs the contents of the message, appends its own certificate, and forward broadcasts the message to each of its neighbors. The signature prevents spoofing attacks that may alter the route or form loops.

Let B be a neighbor that has received from A the RDP broadcast, which it subsequently rebroadcasts.

$$B \rightarrow \text{brdcast} : [[\text{RDP}, \text{IP}_X, \text{cert}_A, N_A, t]K_{A-}]K_{B-}, \text{cert}_B \tag{3}$$

Upon receiving the RDP, B's neighbor C validates the signature with the given certificate. C then removes B's certificate and signature, records B as its predecessor, signs the contents of the message originally broadcast by A, appends its own certificate, and forward broadcasts the message. C then rebroadcasts the RDP.

$$C \rightarrow \text{brdcast} : [[\text{RDP}, \text{IP}_X, \text{cert}_A, N_A, t]K_{A-}]K_{C-}, \text{cert}_C \tag{4}$$

K_{A+}	Public-key of node A.	N_a	Nonce issued by node A.
K_{A-}	Private-key of node A.	IP_A	IP address of node A.
$\{d\}K_{A+}$	Encryption of data d with key K_{A+}.	RDP	Route Discovery Packet identifier.
$[d]K_{A-}$	Data d digitally signed by node A.	REP	REPly packet identifier.
cert_A	Certificate belonging to node A.	SPC	Shortest Path Confirmation packet identifier.
t	timestamp.	RSP	Recorded Shortest Path packet identifier.
e	Certificate expiration time.	ERR	ERRor packet identifier.

Figure 4. Table of variables and notation.

Each node along the path repeats these steps of validating the previous node's signature, removing the previous node's certificate and signature, recording the previous node's IP address, signing the original contents of the message, appending its own certificate and forward broadcasting the message.

5.1.5 Authenticated Route Setup

Eventually, the message is received by the destination, X, who replies to the first RDP that it receives for a source and a given nonce. There is no guarantee that the first RDP received traveled along the shortest path from the source. An RDP that travels along the shortest path may be prevented from reaching the destination first if it encounters congestion or network delay, either legitimately or maliciously manifested. In this case, however, a non-congested, non-shortest path is likely to be preferred to a congested shortest path because of the reduction in delay. Because RDPs do not contain a hop count or specific recorded source route, and because messages are signed at each hop, malicious nodes have no opportunity to redirect traffic with the exploits we described in Section 3.

After receiving the RDP, the destination unicasts a Reply (REP) packet back along the reverse path to the source. Let the first node that receives the REP sent by X be node D.

$$X \rightarrow D : [\text{REP}, \text{IP}_a, \text{cert}_x, N_A, t]K_{X-} \tag{5}$$

The REP includes a packet type identifier ("REP"), the IP address of A (IP_a), the certificate belonging to X (cert_x), the nonce and associated timestamp sent by A. Nodes that receive the REP forward the packet back to the predecessor from which they received the original RDP. Each node along the reverse path back to the source signs the REP and appends its own certificate before forwarding the REP to the next hop. Let D's next hop to the source be node C.

$$D \rightarrow C : [[\text{REP}, \text{IP}_a, \text{cert}_x, N_A, t]K_{X-}]K_{D-}, \text{cert}_D \tag{6}$$

C validates D's signature on the received message, removes the signature and certificate, then signs the contents of the message and appends its own certificate before unicasting the REP to B.

$$C \rightarrow B : [[\text{REP}, \text{IP}_a, \text{cert}_x, N_A, t]K_{X-}]K_{C-}, \text{cert}_C \tag{7}$$

Each node checks the nonce and signature of the previous hop as the REP is returned to the source. This avoids attacks where malicious nodes instantiate routes by impersonation and replay of X's message. When the source receives the REP, it verifies the destination's signature and the nonce returned by the destination.

5.2 Route Maintenance

ARAN is an on-demand protocol. Nodes keep track of whether routes are active. When no traffic has occurred on an existing route for that route's lifetime, the route is simply deactivated in the route table. Data received on an inactive route causes nodes to generate an Error (ERR) message that travels the reverse path toward the source. Nodes also use ERR messages to report links in active routes that are broken due to node movement. All ERR messages must be signed. For a route between source A and destination X, a node B generates the ERR message for its neighbor C as follows:

$$B \rightarrow C : [ERR, \text{IP}_A, \text{IP}_X, \text{cert}_b, N_b, t]K_{B-} \tag{8}$$

This message is forwarded along the path toward the source without modification. A nonce and timestamp ensure that the ERR message is fresh.

It is extremely difficult to detect when ERR messages are fabricated for links that are truly active and not broken. However, because messages are signed, malicious nodes cannot generate ERR messages for other nodes. The non-repudiation provided by the signed ERR message allows a node to be verified as the source of each ERR message that it sends. A node that transmits a large number of ERR messages, whether the ERR messages are valid or fabricated, should be avoided.

5.3 Responses to Erratic Behavior

Erratic behavior can come from a malicious node, but it can also come from a friendly node that is malfunctioning. ARAN's response does not differentiate between the two and regards all erratic behavior as the same. Erratic behavior includes the use of invalid certificates, improperly signed messages, and misuse of route error messages. ARAN's response to erratic behavior is a local decision and the details are left to implementors.

5.4 Key Revocation

In some environments with strict security criteria, the required certificate revocation mechanism must be very reliable and expensive. Due to the desired low-overhead in wireless networks, and to the lower standards of security sought in the managed-open environment, a best-effort immediate revocation service can be provided that is backed up by the use of limited-time certificates.

In the event that a certificate needs to be revoked, the trusted certificate server, T, sends a broadcast message to the ad hoc group that announces the revocation. Calling the revoked certificate $cert_r$, the transmission appears as:

$$T \rightarrow \text{brdcast} : [revoke, \text{cert}_r]K_{T-} \tag{9}$$

Any node receiving this message re-broadcasts it to its neighbors. Revocation notices need to be stored until the revoked certificate would have expired normally. Any neighbor of the node with the revoked certificate needs to reform routing as necessary to avoid transmission through the now-untrusted node. This method is not failsafe. In some cases, the untrusted node that is having its certificate revoked may be the sole connection between two parts of the ad hoc network. In this case, the untrusted node may not forward the notice of revocation for its certificate, resulting in a partition of the network, that lasts until the untrusted node is no longer the sole connection between the two partitions.

6 Security & Network Performance Analyses

In this section, we provide a security analysis of ARAN by evaluating its robustness in the presence of the attacks introduced in Section 3. We also compare through simulation the performance of ARAN to the AODV routing protocol [15].

Unauthorized participation: ARAN participants accept only packets that have been signed with a certified key issued by the trusted authority. In practice, many single-hop 802.11 deployments are already using VPN certificates; this is the case on the UMass campus. Mechanisms for authenticating users to a trusted certificate authority are numerous; a significant list is provided by Schneier [17]. The trusted authority is also a single point of failure and attack, however, multiple redundant authorities may be used (e.g., as by Zhou and Haas [22]).

Spoofed Route Signaling: Since only the source node can sign with its own private key, nodes cannot spoof other nodes in route instantiation. Similarly, reply packets include the destination node's certificate and signature, ensuring that only the destination can respond to route discovery. This prevents impersonation attacks where either the source or destination nodes is spoofed.

Fabricated Routing Messages: Messages can be fabricated only by nodes with certificates. In that case, ARAN does not prevent fabrication of routing messages, but it does offer a deterrent by ensuring non-repudiation. A node that continues to inject false messages into the network, may be excluded from future route computation.

Alteration of Routing Messages: ARAN specifies that all fields of RDP and REP packets remain unchanged between source and destination. Since both packet types are signed by the initiating node, any alterations in transit would be immediately detected by intermediary nodes along the path, and the altered packet would be subsequently discarded. Repeated instances of altering packets could cause other nodes to exclude the errant node from routing, though that possibility is not considered here. Thus, modification attacks are prevented.

Securing Shortest Paths: We believe there is no way to guarantee that one path is shorter than another in terms of hop count. Tunneling attacks, such as the one presented in Section 3.1.4, are possible in ARAN as they are in any secure routing protocol. Securing a shortest path cannot be done by any means except by physical metrics such as a timestamp in routing messages. Accordingly, ARAN does not guarantee a shortest path, but offers a *quickest* path which is chosen by the RDP that reaches the destination first. Malicious nodes do have the opportunity in ARAN to lengthen the measured time of a path by delaying REPs as they propagate, in the worse case by dropping REPs, as well as delaying routing after path instantiation. Finally, malicious nodes using ARAN could also conspire to elongate all routes but one, forcing the source and destination to pick the unaltered route; clearly, a difficult task.

Replay Attacks: Replay attacks are prevented by including a nonce and a timestamp with routing messages.

6.1 Network Performance

We performed our evaluations using the Global Mobile Information Systems Simulation Library (GloMoSim) [1]. We used a 802.11 mac layer and CBR traffic over UDP.

We simulated two types of field configurations: 20 nodes distributed over a 670m x 670m terrain, and 50 nodes over a 1000m x 1000m terrain. The initial positions of the nodes were random. Node mobility was simulated according to the random waypoint mobility model [5], in which each node travels to a randomly selected location at a configured speed and then pauses for a configured pause time, before choosing another random location and repeating the same steps. Node transmission range was 250m. We ran simulations for constant node speeds of 0, 1, 5 and 10 m/s, with pause time fixed at 30 seconds. We simulated five CBR sessions in each run, with random source and destination pairs. Each session generated 1000 data packets of 512 bytes each at the rate of 4 packets per second.

ARAN was simulated using a 512 bit key and 16 byte signature. These values are reasonable to prevent compromise during the short time nodes spend away from the certificate authority and in the ad hoc network.

For both protocols, we assumed a routing packet processing delay of 2ms. This value was obtained through field testing of the AODV protocol implementation [6]. Additionally, a digital signature generation delay of 8.5ms and verification delay of 0.5ms was simulated for ARAN. These values were obtained by measuring the multiple running times of the RSA digital signature and verification algorithm on a laptop computer with a Mobile Pentium III (750/600 MHz) processor and 128 MB RAM, running Red Hat Linux 7.2. Additionally, a random delay between 0 and 10ms was introduced before a broadcast packet is transmitted in order to minimize collisions.

In order to compare the performance of ARAN and AODV, both protocols were run under identical mobility and traffic scenarios. A basic version of AODV was used, which did not include optimizations such as the expanding ring search and local repair of routes. This enables a consistent comparison of results.

We evaluated six performance metrics:

(1) **Packet Delivery Fraction:** This is the fraction of the data packets generated by the CBR sources that are delivered to the destination. This evaluates the ability of the protocol to discover routes.

(2) **Routing Load (bytes):** This is the ratio of overhead bytes to delivered data bytes. The transmission at each hop along the route was counted as one transmission in the calculation of this metric. ARAN suffers from larger control overhead due to certificates and signatures stored in packets.

(3) **Routing Load (packets):** Similar to the above metric, but a ratio of control packet overhead to data packet overhead.

(4) **Average Path Length:** This is the average length of the paths discovered by the protocol. It was calculated by averaging the number of hops taken by each data packet to reach the destination.

(5) **Average Route Acquisition Latency:** This is the average delay between the sending of a route request/discovery packet by a source for discovering a route to a destination and the receipt of the first corresponding route reply. If a route request timed out and needed to be retransmitted, the sending time of the first transmission was used for calculating the latency.

(6) **Average End-to-End Delay of Data Packets:** This is the average delay between the sending of the data packet by the CBR source and its receipt at the corresponding CBR receiver. This includes all the delays caused during route acquisition, buffering and processing at intermediate nodes, retransmission delays at the MAC layer, etc.

6.1.1 Performance Results

Figures 5 shows the observed results for both the 20 and 50 node networks. Each data point is an average of 10 simulation runs with identical configuration but different randomly generated mobility patterns. Error bars report 95% confidence intervals and are small in all cases.

As shown in Fig. 5 (top-left), the packet delivery fraction obtained using ARAN is above 95% in all scenarios and almost identical to that obtained using AODV. This suggests that ARAN is highly effective in discovering and maintaining routes for delivery of data packets, even with relatively high node mobility.

Traditionally, the shortest path to a destination (in terms of number of hops) is considered to be the best routing path. AODV explicitly seeks shortest paths using the hop count field in the route request/reply packets. ARAN, on the other hand, assumes that the first route discovery packet to reach the destination must have traveled along the best path (i.e., the path with the least congestion).

The average path length graphs are almost identical for the two protocols, as shown in Fig. 5 (bottom-left). This indicates that even though ARAN does not explicitly seek shortest paths, the first route discovery packet to reach the destination usually travels along the shortest path. Hence ARAN is as effective in finding the shortest path as AODV. It should be noted, however, that in networks with significantly heavier data traffic loads, congestion could prevent the discovery of the shortest path with ARAN.

Fig. 5 (top-middle) shows routing load measurements. ARAN's byte routing load is significantly higher and increases to nearly 100% for 50 nodes moving at 10 m/s, as compared to 45% for AODV. This due to the security data.

While the number of control bytes transmitted by ARAN is larger than that of AODV, the number of control packets transmitted by the two protocols is roughly equivalent. Fig. 5 (bottom-middle) shows the average number of control packet transmitted per delivered data packet. AODV has the advantage of smaller control packets; smaller packets have a higher probability of successful reception at the destination. However, due to the IEEE 802.11 MAC layer overhead for unicast transmissions, a significant part of the overhead of control packets is in acquiring the channel. In this respect, the two protocols demonstrate nearly the same amount of packet overhead.

Fig. 5 (top-right) shows that the average route acquisition latency for ARAN is approximately double that for AODV. While processing ARAN routing control packets, each node has to verify the digital signature of the previous node, and then replace this with its own digital signature, in addition to the normal processing of the packet as done by AODV. This signature generation and verification causes additional delays at each hop, and so the route acquisition latency increases. (In the course of the experiments, we found that with the expanding ring search enabled, AODV's route acquisition latency becomes significantly greater than that of ARAN for two and three hop routes.)

The data packet latencies for the two protocols are again almost identical (see Fig. 5 (bottom-right)). Although ARAN has a higher route acquisition latency, the number of route discoveries performed is a small fraction of the number of data packets delivered. Hence the effect of the route acquisition latency on average end-to-end delay of data packets is not significant. The processing of data packets is identical when using either protocol, and so the average latency is nearly the same.

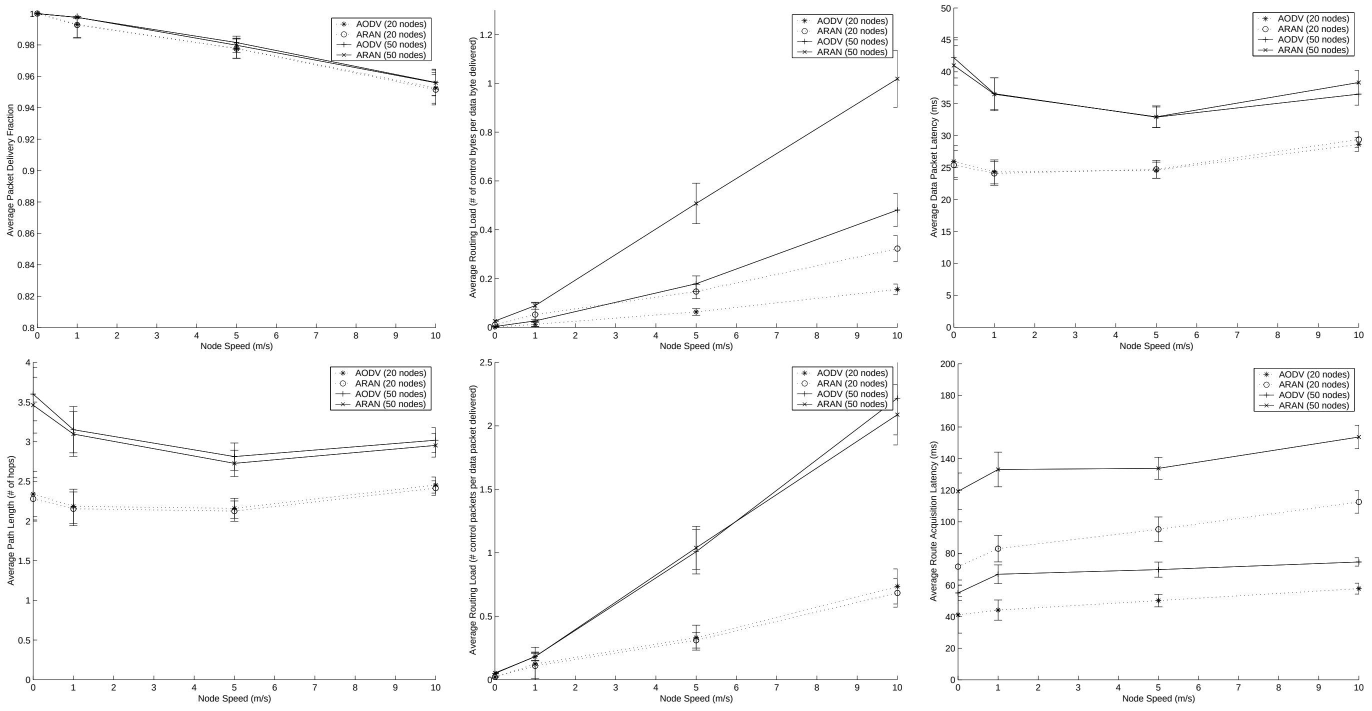


Figure 5. (Left) top: packet delivery fraction; bottom: average path length. (Middle) routing load, top: in bytes; bottom: in packets. (Right) top: avg route acquisition delay; bottom: avg end-to-end delay of data packets.

6.1.2 Effect of Malicious Node Behavior

The experiments described in the previous sections compare the performance of ARAN and AODV when all the nodes in the network are well-behaved or benign. We conducted additional experiments to determine the effect of malicious node behavior on the two protocols. We used a field configuration of 50 nodes distributed over a 1000m x 1000m area.

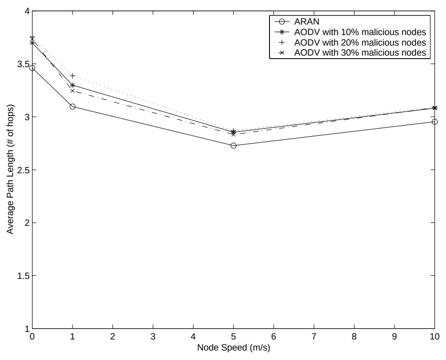
As illustrated earlier in the paper, various types of malicious behavior are possible when using AODV. The malicious behavior simulated in these experiments is as follows: whenever a malicious node forwards an RREQ or RREP packet, it illegally resets the hop count field to 0, thus pretending to be only one hop away from the source or destination node, respectively. The objective of a malicious node is to try to force the selected routes to pass through itself by exploiting the routing protocol, so that it is able to overhear and potentially modify or drop data packets. The effect of this behavior is that non-shortest paths containing malicious nodes are likely to be selected, and the average path length increases. ARAN, on the other hand, cannot be exploited in this fashion. When using ARAN, the selected route could still pass through a malicious node; however, the routing protocol cannot be manipulated to force this behavior.

We ran simulations with 10%, 20% and 30% malicious nodes for each protocol. The malicious nodes were selected randomly. We measured the following metrics:

Average Path Length: Malicious nodes can exploit AODV so that non-shortest paths are selected, while such exploitation is not possible with ARAN. This metric indicates the extent of path elongation in AODV in the presence of different percentages of malicious nodes. The metric is important because longer routes result in greater routing overhead and longer data packet delays.

Fraction of Data Packets Received that passed through Malicious Nodes: This metric indicates the fraction of data packets that traverse malicious nodes when using each routing protocol, in the presence of different percentages of malicious nodes. The metric is important because data packets passing through malicious nodes are overheard by the malicious nodes, and could potentially be modified or dropped.

Fig. 6 illustrates the results of the experiments. As seen in Fig. 6a, the average path length increases about 10% for AODV in the presence of malicious nodes. Figure 6b shows that when using AODV, a much larger fraction of data packets passes through malicious nodes, as compared to using ARAN. For instance, in the presence of 10% malicious nodes with no node mobility, only 22% of data packets pass through malicious nodes when using ARAN, as compared to almost 40% when using AODV. This is because malicious nodes can potentially manipulate AODV to make routes pass through themselves.



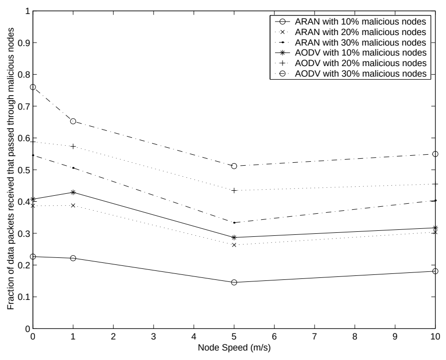


Figure 6. Effect of Malicious Node Behavior: (a) Average path lengths; (b) Fraction of data packets received that passed through malicious nodes.

7 Conclusion

Existing ad hoc routing protocols are subject to a variety of attacks that can allow attackers to influence a victim's selection of routes or enable denial-of-service attacks. We have shown a number of such attacks, and how they are easily exploited in two ad hoc routing protocols under consideration by the IETF. In particular, we introduced the notion of a tunneling attack, in which collaborating malicious nodes can encapsulate messages between them to subvert routing metrics.

Our protocol, ARAN, provides a solution for securing routing in the managed-open environment. ARAN provides authentication and non-repudiation services using pre-determined cryptographic certificates that guarantees end-to-end authentication. In doing so, ARAN limits or prevents attacks that can afflict other insecure protocols.

ARAN is a simple protocol that does not require significant additional work from nodes within the group. Our simulations show that ARAN is as efficient as AODV in discovering and maintaining routes, at the cost of using larger routing packets which result in a higher overall routing load, and at the cost of higher latency in route discovery because of the cryptographic computation that must occur.

References

[1] The global mobile information systems simulation library (glomosim). http://pcs.cs.ucla.edu/projects/glomosim.

[2] W. Arbaugh, N. Shankar, and Y.C. Wan. Your 802.11 wireless network has no clothes. Technical report, Dept. of Computer Science, University of Maryland, March 2001.

[3] E.M. Belding-Royer and C.-K. Toh. A review of current routing protocols for ad-hoc mobile wireless networks. *IEEE Personal Communications Magazine*, pages 46–55, April 1999.

[4] N. Borisov, I. Goldberg, and D. Wagner. Intercepting mobile communications: The insecurity of 802.11. http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html.

[5] 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 *Proc. ACM MOBICOM*, pages 85–97, Oct. 1998.

[6] I.D. Chakeres and E.M. Belding-Royer. A quantitative analysis of simulation and implementation performance for the aodv routing protocol. Submitted for publication.

[7] C. R. Davis. *IPSec: Securing VPNs*. McGraw-Hill, New York, 2000.

[8] J.-P. HuBaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad hoc networks. In *Proc. ACM MOBICOM*, Oct. 2001.

[9] D. Johnson, D. Maltz, Y.-C. Hu, and J. Jetcheva. The dynamic source routing protocol for mobile ad hoc networks. *IEEE Internet Draft*, March 2001. draft-ietf-manet-dsr-05.txt (work in progress).

[10] J. Kong et al. Providing robust and ubiquitous security support for mobile ad-hoc networks. In *Proc. IEEE ICNP*, pages 251–260, 2001.

[11] S.-J. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks.

[12] S. Murthy and J.J. Garcia-Lunca-Aceves. An efficient routing protocol for wireless networks. *ACM Mobile Networks and Applications Journal*, pages 183–197, Oct. 1996.

[13] V. Park and M. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proc. INFOCOMM*, April 1997.

[14] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. *Computer Communications Review*, pages 234–244, Oct. 1994.

[15] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance vector routing. In *IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, Feb. 1999.

[16] J. Scambray, S. McClure, and G. Kurtz. *Hacking Exposed*, chapter 10, pages 453–456. McGraw-Hill, 2nd edition, 2001.

[17] B. Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* John Wiley & Sons, Inc., New York, 1996.

[18] A. Stubblefield, J. Ioannidis, and A. D. Rubin. Using the fluhrer, mantin, and shamir attack to break wep. Technical Report TD-4ZCPZZ, AT&T Labs, August 2001.

[19] M. Tripunitara and P. Dutta. A Middleware Approach to Asynchronous and Backward-Compatible Detection and Prevention of ARP Cache Poisoning. In *Proc. ACSAC*, pages 303–309, Dec. 1999.

[20] F. Wang, B. Vetter, and S. Wu. Secure routing protocols: Theory and practice. Technical report, North Carolina State University, May 1997.

[21] S. Yi, P. Naldurg, and R. Kravets. Security-aware ad hoc routing for wireless networks. In *Proc. ACM Mobihoc*, 2001.

[22] L. Zhou and Z. J. Haas. Securing ad hoc networks. *IEEE Network*, 13(6):24–30, 1999.