September 28, 2004

Dear Editors in Chief,

Please find enclosed our article revised to address all reviewers' comments. Our dilemma is that while our original submission was 17 pages, our revision is limited to 12 pages, and the reviewers' comment mainly were concerned with omissions. Therefore, we have provided many more details in this cover letter than appear in our revision.

The simplest concern for us to address was omitted related work. The revision contains all missing references. Other comments require us to respond in more detail.

For readability, we have broken the reviewers' concerns into sections on denial-of-service attacks, energy consumption, comparisons to related work, and miscellaneous other comments. Reviewers' comments appear as italicized text.

Sincerely,

Kimaya Sanzgiri
on behalf of the other authors

Denial-of-Service Attacks:

- *The paper does not consider several sorts of denial-of-service attacks that could be easily perpetrated, perhaps more easily than the sorts of attacks they do consider, and yet could have a devastating effect on the network. Some of these are identified in the detailed comments from reviewers.*
- *You don't seem to consider denial-of-service attacks in which nodes generate many route requests, even in the section on erratic behavior. For example, a node could generate a bunch of different route requests to a bunch of different destinations, all of which are valid in terms of authenticity, but none of which are needed. Given the amount of transmission and processing required for route requests and replies, this is a simple and effective way to congest the network.*

We have carefully considered the comments and agree that some attacks mentioned by the reviewers apply to ARAN, but most others are no worse than attacks possible at other layers of the network. We have added a brief new discussion in the paper on DoS attacks, and we present a full discussion of these attacks and address the reviewers' concerns here.

There are three general resources that an attacker can target within an ad hoc network: transmission bandwidth, processing cycles at receivers, and power from batteries at receivers. Powering the radio is typically two-to-three times as expensive as powering the CPU, as we detail in our section on energy. Because of this asymmetric cost, attacks that require victims to transmit or receive are more dangerous than attacks that require CPU operations. (In this discussion, we generally consider attacker nodes to have the same capabilities as benign nodes, except where noted.)

DoS attacks are always possible against a universally-accessible radio channel. Our goal is to determine if ARAN provides a more direct or effective mechanism for DoS attacks than can be achieved by attacking the lower and higher network layers.

At the physical layer, constant jamming of the channel is possible. This can consume all available bandwidth, and because receivers listen to the channel when it is active, it drains power. Perhaps the most effective attack is periodic physical-layer jamming. An attacker can transmit only occasionally and put her interface to sleep in between transmits; this avoids listening to the channel, which, in terms of power consumption, is almost as expensive as transmitting. If the frequency is timed correctly, the attacker can interrupt data transmission before the sender is finished but after the RTS/CTS exchange, forcing the sender (and the recipient) to repeat the entire exchange. A short jamming burst is all that is needed, which makes the attack more costly for the victims than the attacker.

At the MAC layer, similar attacks make it possible to monopolize transmission bandwidth from nearby nodes by violating the access control protocol; e.g., a node can ignore the back off assumptions when attempting to access the channel. Like jamming, these attacks prevent nodes from transmitting, and they drain power since all network cards listen to the channel when packets are sent to any destination.

On the other side of the stack, at the transport level, it is possible for a node to send an excessive amount of data across the network. Depending on the intent of the sender, this might be large files that are legitimate, or it might be garbage data. Attackers might also manipulate the transport protocol as is done in TCP Dayton [1]. Because data traverses the network, transport-level attacks can affect nodes far from the attacker. Additionally, because nodes must forward packets, the energy expenditure is much higher. Bandwidth is also consumed, although other nodes still have the opportunity to send their own traffic since the MAC protocol is being followed.

Our first question is whether the mechanisms within ARAN can be used to facilitate similar attacks. The second question is whether the attacker can have more success with ARAN than she can with attacks at other layers. We examine two cases: one in which the attacker lacks a valid ARAN certificate, and one in which it possesses one or more valid ARAN certificates.

- In the case where the attacker lacks a valid certificate, she can send many route requests that have invalid signatures. These requests only go as far as nodes in radio range. At these neighbors, the signatures are unverifiable, and the packets are dropped. Like the physical- and MAC-layer attacks, the bandwidth of only direct neighbors is affected. However, the ARAN-based attack is not as damaging in general because the link layer will provide neighbors with a fair opportunity to transmit. The bogus-signature attack costs processing at receivers because signatures need to be verified, but these costs are not significant in context. The attack is more expensive for the attacker than the victim. First, the receivers must keep their radios powered and idle whether the attack is executed or not. Second, reception costs for the victim is two-thirds the cost of the attacker's transmission [2], and these invalid packets are not forwarded by the victim. For example, this difference can total 450mW or 575mW [2]. Finally, given that the receivers' devices would be powered regardless of the attack (including voltage regulation, memory refresh, and possibly display), the CPU energy cost of signature verification is much less than the difference between the victim's cost of reception and the attacker's cost of transmission.

  If we assume that the goal of a DoS attacker is to deny availability of the network for the longest period of time possible, her best approach (when she does not have a valid certificate) is to jam the channel selectively as stated above. In other words, given that ARAN cannot control access to the channel, its DoS attacks, when the attacker lacks a valid certificate, are no worse than attacks available at the physical and link layers.

- In the second case, where attackers possess valid certificates, they can conduct some successful attacks, because packets are forwarded along the network.

  Nodes can flood valid route requests, just as they can flood data without reason at the transport layer. Route creation in ARAN involves checking of two signatures, which can be expensive computationally. However, since data packets can be larger in size (up to the IP limit) and transmission is the most expensive operation, transport layer floods are more effective at wasting node energy resources than route request flooding.

In essence, we agree with the reviewer comments that some attacks are possible, particularly in the open environment. We have added language to the paper to make these limitations clear and how they compare to attacks at other layers.

Packet Forwarding Attacks:

- *I was surprised that you did not consider packet forwarding attacks, such as redirecting, dropping, or replaying data packets. These can be devastating to the users trying to get traffic through the network. ARAN assumes a particular type of packet forwarding, namely forwarding based on the destination address contained in the packet, and this has some implications for the security of packet forwarding. Note that it may be more difficult to detect when packets are being incorrectly forwarded and to infer the source of the forwarding problems, with destination-based forwarding than with source routing. For example, with source routing, the designated recipient of a packet can look at the list of nodes in the source route carried in the packet and determine whether it is on the list.*
- *The authors have neglected security problems related to forwarding data packets along the routes discovered, and this is a significant omission from the paper, since route selection and packet forwarding are closely coupled.*
- *The authors do not address attacks on packet forwarding over the routes discovered or whether particular approaches to routing might help prevent or detect such forwarding problems. Since packet forwarding is contingent upon the output of the routing procedure, security issues associated with packet forwarding should be considered together with those associated with routing.*

Nodes can drop packets for no reason and re-direct the packets to the wrong node. While there is no protection against this attack, we are unaware of any routing protocol for unicast, multicast, or ad hoc routing that forces nodes to behave correctly in terms of forwarding. However, the attack is detectable in that the source will find a path to the destination but not be able to get data through (this can be detected by mechanisms such as transport/application layer acknowledgements). In this case it is possible to add a mechanism to ARAN that can trace the path between source and destination and return the route. If nodes report their successor on the route, misbehaving nodes will always be reported. This can be used to help determine the misbehaving node.

The main limitation of the use of ARAN certificates, therefore, is that certificates are available in the *open environment* for almost nothing. Therefore, any attacker that is blacklisted by a reputation system can obtain another certificate and repeat the attack. Notice this is not true of the managed environments.

The problem faced by ARAN is identical to the Sybil attack problem, but fortunately, solutions are the same as well.

A solution is suggested in what we have already proposed in the paper for preventing DoS attacks on the DHCP server, in that nodes be made to perform work to obtain certificates in an open environment. Alternatively, new certificates can require a monetary exchange. Essentially, what we require is that certificates are easy to get once, but obtaining many is costly in time or money.

Energy Costs:

- *Although the authors have considered the effect of computing and transmitting cryptographic information with respect to latency, they have not considered the effect on energy consumption, which is critical for battery-powered mobile devices.*
- *The authors do not assess the energy costs of their cryptographic approach, although they do consider the delays incurred in signature generation, verification, and transmission. Energy is a scarce commodity in many small mobile devices, because of limited battery power, and should at least be mentioned as a concern, especially for schemes that require signature verification and generation at each hop for each routing message.*

We have updated the paper to include a discussion on energy costs. In comparison to unsecured ad hoc routing protocols, the additional energy costs of ARAN come from signature generation/verification and transmission/reception of larger packets.

CPU Energy Cost. ARAN's energy expenditure is high in comparison to protocols that employ hash chains, like Ariadne. This is because ARAN spends more time verifying signatures. However, these costs must be viewed in context of other energy costs of the handheld device. It is important to realize that in an ad hoc network the handheld device must be powered at all times for successful reception of route requests. The question we must ask is, what is the additional energy spent for ARAN's cryptographic operations?

The largest energy drain on a handheld device is due to operating a wireless network interface card (NIC), as several researchers have found. Shih et al. [2] measured two NICs in particular: the Orinoco Wave LAN consumes 805mW when idle, 950mW when receiving, and 1400mW when transmitting; the Cisco AIR-PM350 consumes 1080mW when idle, 1300mW receiving, and 1875mW transmitting. Stemm et al. [3] reported in 1998 that most of the energy cost of running a wireless devices can be attributed to the large amount of time the NIC is *idle* (which is not equivalent to *sleep* mode). Transmission and reception of data are more expensive, but tend to occur less often than idle periods, even during TCP transfers.

Cho [4] measured an idle Linux-based Compaq iPAQ as consuming 470 mW (with the display backlight off and no wireless NIC). The energy cost of computation is much smaller when compared to operating the NIC, and it is often smaller than the other subsystems, including powering memory or the display, as well as voltage regulation. Kremer et al. [5] found that a Compaq iPAQ H3600 (not significantly different than our 3850) executing their custom-built networked image-recognition program dissipated 2200mW with a Orinoco WaveLaAN connected, but 1250mW without the card (without powering the display in both cases). They found for that for an iPAQ, less than 12% of the overall power budget was spent on the processor for their computationally-intense application.

From our experiments (see Table 3 in the paper), we know that the running time for an iPAQ to process an RDP packet is 45ms. If we use Kremer's, Bahl's, and Cho's measurements (all of the same iPAQ and wireless devices) as a baseline, we can provide an estimate of the costs of ARAN's cryptographic operation.

If we set the CPU power cost as 12% of 1250mW as per Kremer's measurements, then the energy usage for processing an RDP packet is $150mW \cdot 0.045sec = 6.8mJ$. Costs equal to ARAN's CPU operations will be spent by an idle radio (805mW [2]) coupled with an idle iPAQ (470mW [4]) every 5ms. This is a minor cost compared to the overall operation of the device.

Operating the NIC. As we stated above, idle radios are the largest power drain on a handheld: even so, we can compute the costs of transmitting packets in ARAN. ARAN has significantly larger packet sizes for route creation and data forwarding than AODV. For each RREQ, ARAN requires statement of a source and destination (32 bits each), two signatures (128 bits each) and a nonce (40 bits). ARAN also appends the certified public key of the source as a method of key distribution (512 bits + 128-bit signature), but we will exclude this cost for a fair comparison since Ariadne/TESLA [6], [7]completely ignores key distribution (note that TESLA relies on public keys to initialize hash chains). ARAN RREP packets are the same size. Data packets require source and destination (32 bits each) and a timestamp (32 bits).

In comparison, AODV has 40-byte RREQ and RREP packets, and requires the source and destination for data packets (64 bits).

Ariadne/TESLA route requests require eight fields: source (32 bits), destination (32 bits), id (128-bit nonce), time interval (32 bits), hash chain (128 bits), node list (32 bits per hop), MAC list (128 bits per hop). The reply includes the source (32 bits), destination (32 bits), time interval (32 bits), node list (32 bits per hop), MAC list (128 bits per hop), target MAC (128 bits), key list (128 bits per hop). There is no explicit definition of the Ariadne data header, so we imagine the source-routed data packets require a list of the IPs from the source to the destination (32 bits each), a MAC of the current packet's contents (128 bits), time interval (32 bits), and a disclosure of the previous interval's key (128 bits).

A summary of the header bit counts for each protocol and packet type follows (where x is the number of hops in the path to the destination):

	RREQ (bits)	RREP (bits)	Data (bits)
ARAN	360	360	64
AODV	40	40	64
Ariadne	352 + 160x	224 + 288x	320 + 32x

Note from the table that Ariadne requires larger packet headers after only a single hop for all packet types.

Feeney and Nilsson measured energy costs of a Lucent 802.11b WaveLAN card for a packet of S bytes: $(2.1S + 272)\mu J$ per byte to send a broadcast (RREQ) packet; $(0.26S + 50)\mu J$ to receive a broadcast (RREQ) packet; $(.48S + 431)\mu J$ per byte to send a point-to-point (RREP or data) packet; and $(0.12S + 50)\mu J$ to receive a point-to-point (RREP or data) packet. According to those equations, we can estimate the energy costs of ARAN, AODV, and Ariadne/TESLA:

	RREQ (μJ)	RREP (μJ)	Data (μJ)
ARAN	367	367	289
AODV	282	282	289
Ariadne	364 + 42x	330 + 76x	356 + 8x

The power costs of ARAN are certainly worse than AODV but better than Ariadne, which suffers from its source-routed approach that causes the packets to grow in size as the path count increases (note that the table represents energy not power). In any case, for all protocols, it is clear that the majority of the radio cost of transmission/reception is from data packet forwarding and not route creation. Moreover, these costs in micro-joules are insignificant compared to the milli-joules drained during idle periods.

Comparison to Related Work:

- *The major weakness of the protocol is the requirement of a pre-deployment phase during which certificates are exchanged. There is no mechanism for the renewal of certificates as they expire. Therefore, the protocol can be in operation for finite time duration only. Routing possibilities diminish with time (this weakness is already acknowledged by the authors).*
  Unauthorized nodes may be involved in routing for a while because the participants haven't been updated yet by the certificate server (this weakness is already acknowledged by the authors).

All secure routing protocols to date use public key encryption, ARAN just provides the most explicit details on their use. Ariadne allows three options for authentication: pre-distribution of pairwise symmetric shared keys (an admitted impracticality); pre-distribution of public keys; and the TESLA protocol which uses public keys to initialize hash chains. For all three options, Ariadne assumes a certification authority is not used and the cost is explicitly ignored during analysis. No renewal, expiration, or revocation mechanisms are proposed.

While the protocol SEAD "does not use asymmetric cryptographic operations in the protocol", it assumes some mechanism for a node to distribute an authentic hash chain, with suggestions including "public keys" and "PGP-like certificates", as well as a centralized "trusted node" (i.e., a certification authority). Again, the operation and costs of these mechanisms are ignored, as is revocation.

It is not the use of public keys within ARAN that is problematic — all protocols use them to avoid a priori pairwise shared-key distribution. The larger question is if the way they are used by the protocol affects performance. We've already discussed above how NIC energy costs dominate CPU energy costs. We compare delays below.

- *The paper does not provide a thorough comparison with other, already published solutions (Ariadne, SEAD,...)*

While we did not significantly expand the discussion in the paper because of severe space limitations, we have addressed this in some detail in this letter, and could move the discussion into the paper if it were deemed critical.

We also do not include a quantitative comparison of processor costs as it is clear we have a larger processing requirement at each hop due to the RSA signature verification. Our evaluation of ARAN is worst-case: key exchange between nodes is simple when public keys are already known. As one reviewer thoughtfully pointed out:

> *(Note that a shared secret need not even be negotiated if a node is pre-configured with certain types of public keys for other nodes in the network, provided the other nodes are known ahead of time. For example, if $a^x \bmod p$ is the public key of X and x is the private key, then X and Y can compute a shared secret, $a^x y$, each using their own private key and the public key of the other.)*

There are advantages to ARAN over other protocols. Unlike TESLA, ARAN does not require nodes in the network to estimate the round trip time between all peers, which, in a mobile network, is dynamic. The security of TESLA is based on a *disclosure delay* between when keys are used by the sender and when keys are revealed to all receivers. The disclosure delay is slightly larger than the largest roundtrip time in the network [7] (i.e., the diameter RTT).

If the diameter RTT estimation by each peer in TESLA is too short then a series of packets will be dropped as bogus by receivers. The estimation cannot be too long because all packets are delayed by that estimation (once for RREQs and again for RREPs on the return path). This is because, in TESLA, packets must be buffered at receivers for that duration before they can be authenticated. Even using the *immediate authentication* variant of TESLA (*ibid.*) packets must be delayed at the sender by at least the longest RTT in the network before they can be sent, even if the destination is the neighbor.

If the disclosure delay in TESLA is no shorter than our additional processing costs, then the schemes are equivalent in terms of delay. Unfortunately, it is not clear how to estimate the longest RTT of a real ad hoc network since implementations are, to date, only lab exercises.

We can compare the two approaches informally. First, note that the route request packets in Ariadne/TESLA are delayed by at least the diameter RTT plus the processing/transmission delays of each peer along the path to the destination, and then

both types of delay again in the reverse direction. Second, note our delays are due only to the processing/transmission delays in both directions. The delays of the protocols are equivalent if twice the longest roundtrip time in the network plus hash chain processing (the delay of Ariadne/TESLA) is equal to the double the signature generation and verification delay at each hop of the path (the delay of ARAN).

As our experimental results show, we were able to achieve route acquisition latency delays of 237ms for a 512-bit RSA key with a series of three iPAQs. As an informal point of comparison, in its simulations, Ariadne was set to use a 200ms disclosure delay for a 1500m-by-300m field. This implies all route acquisition latencies are at least 400ms not including processing costs: 200ms delay for the RREQ and another 200ms delay for the RREP. Note that the longest distance in the 1500m-by-300m field (1529m along the diagonal) can be covered by three-to-four nodes at the corners with 250m radios.

Our results also show that with Pentium-3 processors, the delay in signature verification reduced to 22ms. With a symmetric key exchange between neighbors, our delays between iPAQs would be likely be two or three orders of magnitude shorter than we show in our Table 4. Interestingly, our key sizes do not need to increase in length with improvements in CPUs of handheld devices because it is based on the resources of the attacker (who we assume has a super computer). However, the delays in TESLA will continue to be based on the longest network RTT.

In sum, we do not believe ARAN's delays are significantly larger than Ariadne/TESLA, and in some cases may even be smaller, though we cannot offer direct evidence. Power consumption from processing is likely to be higher, but is not likely to be the dominant factor in the device as described above.

Other concerns:

- *You really should seriously consider using message authentication codes based on hashing for the hop-by-hop authentication required for route requests and replies. Granted, you do mention briefly mention this type of approach at the end of section 5.0.2, but you really should consider adopting it because it is much less expensive than the approach you propose.*

We agree and have re-emphasized the option in the paper. Thank you for pointing out the simple key exchange. We felt that the evaluation should continue to be on the worst case scenario. If it is desired, we can re-run latency evaluations to examine the lower costs of key exchange between neighbors.

- *While the emphasis of this paper is on route discovery, the authors should mention the security advantages and disadvantages of other approaches to routing in ad hoc networks, to set their work in context. For example, link-state routing, while not as popular as the route-discovery approaches, is a viable approach to routing in ad hoc networks, and versions designed specifically for such networks exist. One property of link-state routing, namely that update contents do not change from hop to hop, means that authentication is straightforward and relatively inexpensive. It requires one signing operation at the source and n verification operations, one for each of the n nodes that receives the update. The authors should at least mention the relative costs of securing different types of routing approaches for ad hoc networks.*

There are a number of valid approaches to the problem, but given that we had to reduce the size of the paper by nearly 30%, we were unable to include such a discussion. If it were deemed critical, we could omit something else to include it.

- *Your description of how a recipient of a route request or reply processes a packet is unclear in one respect. The implication from the text, both in the algorithm description in section 5 and in the overhead description in section 7 is that only the first-hop recipient of the request or reply verifies the signature of the source of the packet, while the remaining hops verify the signature of the previous hop of the packet but not the source of the packet. Is this really the behavior you intend?*
- *Do your results in table 3 reflect this, or are these results just for one signature generation and one signature verification? Please clarify.*

We have expanded the description to make clear that each hop checks both its neighbors signature and that of the source. The simulations did take this into account, and we have added language to make this clear as well.

Authenticated Routing for Ad hoc Networks

Kimaya Sanzgiri* Daniel LaFlamme† 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 20057

Abstract*— Initial work in ad hoc routing has considered only the problem of providing efficient mechanisms for finding paths in very dynamic networks, without considering security. Because of this, there are a number of attacks that can be used to manipulate the routing in an ad hoc network. In this paper, we describe these threats, specifically showing their effects on AODV and DSR. Our protocol, named Authenticated Routing for Ad hoc Networks (ARAN), uses public-key cryptographic mechanisms to defeat all identified attacks. We detail how ARAN can secure routing in environments where nodes are authorized to participate but untrusted to cooperate, as well as environments where participants do not need to be authorized to participate. Through both simulation and experimentation with our publicly-available implementation, we characterize and evaluate ARAN and show that it is able to effectively and efficiently discover secure routes within an ad hoc network.

I. Introduction

Securing protocols for mobile ad hoc networks presents unique challenges due to characteristics such as lack of pre-deployed infrastructure, centralized policy and control. In this paper, we make a number of contributions to the design of secure ad hoc routing protocols[1]. First, we describe exploits that are possible against ad hoc routing protocols. We show specifically that two protocols that are under consideration by the IETF for standardization, AODV [9] and DSR [10], although efficient in terms of network performance, are replete with security flaws.

Second, we define and distinguish the heterogeneous environments that make use of ad hoc routing and differ in their assumed pre-deployment and security requirements. This approach is important because satisfying a tighter set of security requirements than an application requires is unwarranted and wasteful of resources.

Third, we propose a secure routing protocol, Authenticated Routing for Ad hoc Networks (ARAN), that detects and protects against malicious actions by third parties and peers. ARAN introduces *authentication*, *message integrity*, and *non-repudiation* to routing in an ad hoc environment as a part of a minimal security policy.

We detail how ARAN can be used in two environments: where mobile users are federated and can be pre-certified (e.g., on a campus) though remain untrusted; and where they are unknown to each other and cannot be pre-certified (e.g., a "rooftop" access point). To our knowledge, ARAN is the first proposal for securing ad hoc routing for rooftop networks.

We analyze the security of ARAN and evaluate its network performance through measurement of both our publicly-available implementation and extensive simulations. We find that although there is a greater performance cost to ARAN as compared to DSR or AODV, the increase in cost is minimal and outweighed by the increased security.

In comparison against related work (e.g.,[6], [7]), ARAN has higher computational costs at each node, which has implications for power costs and latency. However, the dominant energy cost of wireless networking on handheld devices is the idle system with an idle radio [3]; the costs of ARAN's cryptography represent a small price in comparison. ARAN's computational delays are comparable to the mandatory authentication delays required by TESLA [7], a hash-chain-based approach to security. TESLA mandates delays equal to twice the diameter RTT of the network in addition to processing delays, even if the path is between direct neighbors.

This paper is organized as follows. Section II is an overview of recent work on ad hoc network security. Section III presents the security exploits possible in ad hoc routing protocols. Section IV defines three ad hoc environments and the security requirements of any ad hoc network. Section V presents the secure ad hoc routing protocol, ARAN. Section VI provides a security analysis of ARAN while section VII evaluates ARAN through implementation and simulations. Finally, section VIII offers concluding remarks.

II. Background

Several proposed ad hoc routing protocols, for example [9], [10], [11], [12], [13], 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 [9] and DSR [10].

The fundamental differences between ad hoc networks and standard IP networks necessitate the development of new security services. This point has been recognized, and several researchers have examined security problems in ad hoc networks. Numerous solutions have been proposed for providing a secure and reliable certification authority in ad hoc networks [14], [15], [16], [17]. Another problem that has

Supported in part by National Science Foundation awards ANI-522564 and EIA-0080199, and in part by U.S. Dept. 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 or NSF.

[1]This paper represents many refinements and extensions to our original work from IEEE ICNP 2002 [8].

received attention is that of stimulating cooperation among nodes in an ad hoc network and addressing malicious packet dropping [18], [19], [20], [21], [22], [23]. Strategies used include detecting and punishing non-cooperating nodes, rewarding nodes for forwarding packets, concealing the true destination of packets from intermediate nodes, and using redundant data transmissions over multiple paths.

The issue of secure routing in particular has received significant attention. Hu et al. have proposed Ariadne [6], a secure version of DSR. Ariadne can use pre-deployed pairwise symmetric keys or pre-deployed asymmetric cryptography for authentication. The former is more efficient, but requires shared secrets between communicating nodes, which may not always be feasible to establish. A third option for Ariadne is the TESLA authentication scheme, which is also based on asymmetric encryption, thus requiring a certification authority or pre-deployed keys. TESLA requires that packets are delayed by the longest RTT in the network before they are sent (thus route creation incurs this delay in both request and response phases).

Chu et al. developed a secure proactive routing protocol based on DSDV [13] called SEAD [24], which is also based on public-key signed hash chains.

SAODV [25], an early attempt to secure the AODV routing protocol, has numerous security vulnerabilities. For instance, it allows a malicious intermediate node to spoof its identity, illegally modify the hop count on route request messages, and fabricate route error messages.

Yi et al. [26] propose the use of security parameters, such as the trust level of a node in a hierarchical organization, as a routing metric. To secure the scheme, they suggest that all nodes at the same level of trust should share a common secret. This is not very practical, and has many key-management issues.

Papadimitratos et al. [27] propose the Secure Routing Protocol (SRP), which is vulnerable to attacks such as fabricated route error messages. Routing security in sensor networks has been analyzed in [28].

The wormhole attack against secure ad hoc routing protocols is studied and a solution is presented in [29], though implementing the solution requires specialized hardware to achieve a high degree of clock synchronization. Awerbuch et al. design a flooding-free reactive routing protocol based on Swarm Intelligence and the Distributed Reinforcement Learning paradigm [30], which is secure against a dynamic Byzantine adversarial model. Finally, intrusion detection techniques for ad hoc networks have been studied [31], [32].

Our work differs from other work in that we do not assume any hardware modifications or synchronized clocks, and only minimal advance keying from a trusted authority. We also account for the costs of distributing cryptographic material instead of assuming it is pre-deployed.

III. Exploits Against Existing Protocols

Several popular ad hoc routing protocols allow for many different types of attacks. In this section, we classify and briefly describe *modification*, *impersonation*, and *fabrication* exploits against ad hoc routing protocols. Detailed descriptions of the attacks can be found in our previous work [8]. In addition, several attacks are possible in the forwarding operation. Data packets can be dropped, replayed, or redirected. In Section V, we propose a protocol that is not exploitable in these ways.

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 I
VULNERABILITIES OF AODV, DSR, AND ARAN.

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. Additionally, denial-of-service attacks based on non-cooperation and packet dropping, or resource depletion by aggressive route request flooding, are possible in all ad hoc routing protocols. We do not deal with the issue of ensuring protocol compliance, and look only at security problems arising from manipulation of the network routing.

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 I provides a summary of each protocol's vulnerability to the following exploits.

A. 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. Below we briefly describe several modification attacks against AODV and DSR.

1) Redirection by Modified Route Sequence Numbers: Protocols such as AODV and DSDV assign monotonically increasing sequence numbers to routes towards specific destinations. A route with a higher sequence number is preferred over one with a lower sequence number. Thus, 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.

2) Redirection with Modified Hop Counts: In AODV, 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. 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, described in Section III-B.

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, such that the packet can no longer be delivered to the destination.

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 collaborate to encapsulate and exchange messages along existing data paths. Such collaborating nodes can pretend to be neighbors, and falsely represent the length of available paths by preventing honest intermediate nodes from correctly incrementing the path length metric.

It is also possible that instead of tunneling through existing multi-hop routes, the malicious nodes can use a long-range directional wireless link or a wired link between them. Such a link gives the attackers an unfair advantage towards occurring on the shortest delay route between a source and destination. This has been referred to as the wormhole attack in recent literature [6], [29]. However, if the malicious nodes truly lie on the shortest delay path, it could be argued that the selection of this path is not a subversion of the routing protocol. A mechanism for defending against wormhole attacks is presented in [29].

B. 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 other attacks, such as those based on modification. The advantage of spoofing is that the attack cannot be traced back to the malicious node.

C. Attacks Using Fabrication

Fabrication attacks involve the generation of false routing messages. 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.

1) Falsifying Route Errors in AODV and DSR: In AODV and DSR, 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. This message causes the corresponding route to be invalidated in all upstream nodes. A denial-of-service attack can be launched by continually sending route error messages indicating a broken link on the route, thereby preventing the source from communicating with the destination.

2) Route Cache Poisoning in DSR: In DSR, 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. An attacker could easily exploit this method of learning routes and poison route caches by transmitting packets containing invalid routes in their headers.

IV. Security Requirements of Ad hoc Networks

Applications for ad hoc networks include military operations, emergency rescue missions, and simple provisioning of wireless network access, such as at a conference or in a classroom. In this section, we classify ad hoc networks into three distinct environments that differ in security needs and assumed pre-deployment. These classes are defined because it is difficult to construct a single secure ad hoc routing protocol to suit the needs of many heterogeneous wireless applications. The lower security requirements of some environments do not justify use of costly protocols that satisfy stricter security policies. The environments defined in this section enable us to clearly state where we expect to apply our secure protocol.

A. Three Environments

A good secure routing algorithm prevents each of the exploits presented in Section III; it must ensure that no node can prevent successful route discovery and maintenance between any other nodes other than by non-participation.

We define a set of three discrete ad hoc wireless environments: *open*, *managed-open*, and *managed-hostile*. These differ not only in the level of security needed, but also in that some have opportunity for exchange of security parameters before the nodes are deployed.

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.

These requirements help define an *open* environment along with the following distinction: all nodes can be considered authorized. This scenario might exist, for example, for a user walking through an urban environment or driving on a highway.

Managed-open environments are accordingly distinguished by an additional requirement: (6) Unauthorized nodes must 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 predeployment or exchange of public keys, session keys, or certificates. We expect mobile nodes in this environment reside within some common context or geographic proximity. Such an ad hoc network might be formed by peers at a conference, or students on a campus.

We define a *managed-hostile* environment to have requirements listed above as well as the following: (7) The network

K_{A+}	Public key of node A.
K_{A-}	Private key of node A.
K_{AB}	Symmetric key shared by nodes A and B.
$\{d\}K_{A+}$	Encryption of data d with key K_{A+}.
$[d]K_{A-}$	Data d digitally signed by node A.
cert_A	Certificate belonging to node A.
e	Certificate expiration time.
N_A	Nonce issued by node A.
IP_A	IP address of node A.
RDP	Route Discovery Packet identifier.
REP	REPly packet identifier.
t	timestamp.

TABLE II
TABLE OF VARIABLES AND NOTATION.

topology must neither be exposed to adversaries nor to authorized nodes by the routing messages. A managed-hostile environment is formed, for example, by military nodes in a battle environment, or perhaps by emergency response crews in a disaster area. In such an environment, nodes are deployed by a common source. Consequently, there may be opportunity for pre-deployed exchange of security parameters. The distinguishing security threat of the managed-hostile environment is that every node is vulnerable to physical *capture and take-over* of equipment, where hostile entities can then pose as friendly entities at a compromised node. Therefore, exposure of node location from the routing protocol messages is not desirable, else adversaries may gain an opportunity to annihilate users.

In the next section we present the ARAN protocol, which meets the needs of the managed-open and open environments. It does not provide a solution to the managed-hostile environment because it exposes the routing topology.

V. AUTHENTICATED ROUTING FOR AD HOC NETWORKS

In this section, we detail the operation of ARAN. ARAN uses cryptographic certificates to prevent most of the attacks presented in Section III and detect erratic behavior.

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, and does not include routing optimizations present in the latter. It should be noted that these optimizations are the chief cause of most exploits listed in Section III. Route discovery in ARAN is accomplished by a broadcast route discovery message from a source node that is replied to by the destination node. The routing messages are authenticated end-to-end and only authorized nodes participate at each hop between source and destination.

A. Certification of Authorized Nodes

ARAN uses cryptographic certificates to bring authentication, message-integrity and non-repudiation to the route discovery process. ARAN therefore requires the use of a trusted certificate server T, whose public key is known to all valid nodes (or multiple servers may be used [17]). Nodes use these certificates to authenticate themselves to other nodes during the exchange of routing messages. The use of public keys and certificates is common in many secure ad hoc routing protocols, but most assume the existence of such information without any explicit description of how it is transmitted. While ARAN may appear more expensive, it is in part because we account for the distribution of the cryptographic keying material.

In managed-open environments, 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. Details of how certificates are revoked are explained below in Section V-G. Section V-H describes the certification process for open environments.

A node A receives a certificate from T as follows:

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

The certificate contains the IP address of A (IP_A), the public key of A (K_{A+}), a timestamp t of when the certificate was created, and a time e at which the certificate expires. Table II summarizes our notation. These variables are concatenated and signed by T. All nodes must maintain fresh certificates with the trusted server.

B. Authenticated Route Discovery

The goal of end-to-end authentication is for the source to verify that the intended destination was reached. The source trusts the destination to select the return path.

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

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

The RDP includes a packet type identifier ("RDP"), the IP address of the destination (IP_X), A's certificate (cert_A) and a nonce N_A, all signed with A's private key. Note that the RDP is only signed by the source and not encrypted, so the contents can be viewed publicly. The purpose of the nonce is to uniquely identify an RDP coming from a source. Each time A performs route discovery, it monotonically increases the nonce. The nonce is 5 bytes in size, and is thus large enough that it will not need to be recycled within the lifetime of the network.[2] 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

[2]If a source sends a new RDP every millisecond, with a 5 byte nonce, it would take more than 34 years for the value to wrap around.

already processed this RDP. Nodes do not forward messages with already-seen tuples; otherwise, the receiving 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.

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

Upon receiving the RDP, B's neighbor C validates the signatures for both A, the RDP initiator, and B, the neighbor it received the RDP from, using the certificates in the RDP. C then removes B's certificate and signature, records B as its predecessor, signs the contents of the message originally broadcast by A and appends its own certificate. C then rebroadcasts the RDP.

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

Each intermediate node along the path repeats the same steps as C.

C. 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. This RDP need not have traveled along the path with the least number of hops; the least-hop path may have a higher delay, either legitimately or maliciously manifested. In this case, however, a non-congested, non-least-hop path is likely to be preferred to a congested least-hop 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 III.

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.

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

The REP includes a packet type identifier ("REP"), the IP address of A (IP_a), the certificate belonging to X (cert_x) and the nonce 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.

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

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.

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

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 re-play of X's message. When the source receives the REP, it verifies the destination's signature and the nonce returned by the destination.

D. Route Maintenance

ARAN is an on-demand protocol. When no traffic has occurred on an existing route for that route's lifetime, the route is simply de-activated in the route table. Data received on an inactive route causes nodes to generate an Error (ERR) message. 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:

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

This message is forwarded along the path toward the source without modification. A nonce ensures 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, the signature on the message prevents impersonation and enables non-repudiation. A node that transmits a large number of ERR messages, whether the ERR messages are valid or fabricated, should be avoided.

E. 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. We discuss how susceptible ARAN is to this behavior in the next section.

F. Potential Optimizations

Although we have specified the use of public certificates here, it is clear that intermediary nodes (B and C in our examples) can easily agree upon and exchange session keys using the certificates that authenticate their participation in route creation. Two nodes can easily share a symmetric key generated with their own private key and the public key of the other. A session key can last the duration of their juxtaposition and can be a symmetric key, K_{BC} to reduce processing costs; equivalently, juxtaposed peers can create low-cost hash chains between themselves for authentication of future messages. Using these optimizations would decrease computational overhead and power consumption. However, even if these optimizations are used, we require that sources and destinations must include full public-key signatures for end-to-end route discovery and setup messages.

G. 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 the lower standards of security sought in the managed-open and open environments, 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:

$$T \rightarrow \text{broadcast} : [revoke, \text{cert}_r]K_{T-} \quad (9)$$

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.

At the time that the revoked certificate should have expired, the untrusted node is unable to renew the certificate, and routing across that node ceases. Additionally, to detect this situation and to hasten the propagation of revocation notices, when a node meets a new neighbor, it can exchange a summary of its revocation notices with that neighbor; if these summaries do not match, the actual signed notices can be forwarded and re-broadcasted to restart propagation of the notice.

H. ARAN in Open Environments

One of the key characteristics of ARAN is that attackers gain little advantage within ARAN by having additional certificates. This makes ARAN well suited for use in open environments where no user is unauthorized to participate in route creation (see Section IV). Open 802.11 networks (often called “rooftop networks”) in particular have become widespread: `http://www.nodedb.com` lists 8,900 open access points around the world.

Open wireless access points running open DHCP can extend their coverage if participating nodes run ARAN. Nodes can register for a DHCP address and then request that a public key they provide is signed by the DHCP/certificate server.

Up till now, we have assumed that only authorized nodes can participate in ARAN route creation; however, even participating nodes are prevented from malicious actions such as introducing loops, blackholes, and other attacks covered in Section III. Therefore, ARAN itself does not need to be modified. The remaining risk is that attacking nodes can repeatedly change their MAC addresses and continually ask for new DHCP addresses as well as new certificates. Thus, the open environment does have limitations. In particular, it allows certificate holders to flood the network with data packets. This attack is an option in the managed environment, except that the certificate can be revoked without giving the user the ability to receive renewed authorization.

Jakobsson and Juels have an excellent method of combating this problem: proof of work protocols [33]. To summarize this approach, clients are required to solve a puzzle before a request is satisfied, such as factoring a number. The puzzles could require additional work as resources become more scarce. This increases the resources required of attackers to successfully attack the system proportional to the threat of the attack. Alternatively, certificates can cost money, limiting the ability the attackers to requests them limitlessly. A short lifetime on certificates can also help manage the network.

VI. Security Analysis

In this section, we provide a security analysis of ARAN by evaluating its robustness in the presence of the attacks introduced in Section III. As mentioned earlier, we do not consider denial-of-service attacks based on non-cooperation or aggressive participation, which are possible against all ad hoc routing protocols.

Unauthorized participation: Since all ARAN packets must be signed, a node cannot participate in routing without authorization from the trusted certificate server. This access control therefore rests in the security of the trusted authority, the authorization mechanisms employed by the trusted authority, the strength of the issued certificates, and the revocation mechanism. Although we do not detail these functions explicitly, except for certificate revocation, they have been extensively studied by others.

In practice, many single-hop 802.11 deployments already use 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 [34]. 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 [17]).

Spoofed Route Signaling: Route discovery packets contain the certificate of the source node and are signed with the source's private key. 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: Since all routing messages must include the sending node's certificate and signature, ARAN ensures non-repudiation and prevents spoofing and unauthorized participation in routing. 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 detected,

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 III-A.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 could save some processing time by not verifying the previous hop's signature on the RDP packet, thus increasing their chances of being on the quickest route. However such an attack is likely to succeed only if it is executed by multiple malicious nodes on a route, or if a malicious node is already on one of many quick routes to the destination. Malicious nodes also 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.

Forwarding Attacks: We have not detailed a specific method of secure forwarding. This could be accomplished using the cryptographic material available to ARAN, but would add overhead to the cost of data transmission. A simple method of protecting data packets would be to use the route reply process to instantiate shared keys between neighbors, and to use that shared key the basis for a pair-wise HMAC. This enforces that only certificate owners can forward data. It does not prevent certificate holders from replay attacks, but in any protocol, authorized participants can just as effectively attack the system by flooding the network with valid data packets for routes they create. End-to-end integrity can be ensured by the shared key derivable from the two peers' public keys.

Denial-of-Service Attacks: Denial-of-service attacks can be conducted by nodes with or without valid ARAN certificates. In the certificateless case, all possible attacks are limited to the attacker's immediate neighbors because unsigned route requests are dropped. There are more severe attacks available at the MAC and physical layer than ARAN provides. Nodes with valid certificates can conduct effective attacks, however, by sending many unnecessary route requests. Because these are broadcast and forwarded across the network, an attacker can cause widespread congestion and power-loss to all nodes in the network. Because it is difficult to infer the node's intent at the network level, it can be hard to differentiate between legitimate and malicious RREQs.

VII. Network Performance

In this section, we evaluate the performance of ARAN using measurements obtained through both simulation and implementation. Simulations enable us to measure the effectiveness and efficiency of ARAN in reasonably large networks, with and without the presence of malicious nodes. Although simulation is a useful tool for anticipating protocol performance in real networks, it needs to be complemented with protocol implementation in order to obtain a more realistic evaluation of the protocol. With this motivation, we begin this section with a characterization of our ARAN implementation over a three-node network; we then use the quantitative results obtained as input to a simulation of a 50-node network.

A. ARAN Implementation

Our open-source implementation of the ARAN protocol, called `arand`, is publicly available from `http://signl.cs.umass.edu/software/arand`. It is a user-space routing daemon designed to run on Linux systems with kernel 2.4 or higher. The daemon is written in `C` and utilizes the Ad hoc Support Library (ASL) written by Kawadia, Zhang, and Gupta [35]. The ASL and its accompanying Linux kernel module are designed to provide a layer of abstraction that serves as a consistent interface to system functionality required by all ad hoc network protocols. These services include adding and deleting kernel routes as well as notification to the user-space daemon that a route to another host is needed. The library and module also provide functionality to keep track of when routes were last used. This allows routing daemons to delete routes that may no longer exist due to node movement.

The cryptographic functions of `arand` make use of the OpenSSL library (`http://www.openssl.org`), which provides functions for general purpose cryptographic tasks such as public and private key encryption/decryption, signing, and certificate management. Each mobile node is issued an X.509 certificate signed by a common Certification Authority. The certification authority and mobile node certificates can be created and managed using the `aranca` script that is available on the `arand` project site. All routing related communication between the mobile nodes is done using UDP datagrams. These messages include the message types specified by the ARAN protocol such as RDP, REP, and ERR, as well as signed *hello* messages that are used by nodes to discover neighbors.

A typical interaction between mobile nodes running `arand` proceeds as follows. A user on node A attempts to establish a network connection to node C. The kernel on A searches its routing table for a route to C, but does not find one if A and C are out of signal range and cannot receive each other's hello messages or if a previous route between A and C has expired and been deleted from the kernel routing table. `arand` is notified of the need for a route to C by the Ad hoc Support Library, which in turn uses the TUN/TAP feature of the Linux kernel. `arand` running on A checks its state information and determines that it does not have any pending route requests for destination C. It then creates a new RDP message signed with its private key and broadcasts this route request on the network. The protocol then follows the steps specified in Section V.

Each node must cryptographically sign and verify each routing message along a path. These cryptographic operations are relatively expensive, especially when compared to other ad hoc routing protocols that do very little computation per message.

	Average (ms) ± Std. Dev.
Laptop:	
512 bit RSA key:	2.2 ± 0.44
768 bit RSA key:	4.3 ± 0.52
1024 bit RSA key:	7.6 ± 0.62
iPAQ:	
512 bit RSA key:	45.4 ± 1.14
768 bit RSA key:	109.2 ± 1.64
1024 bit RSA key:	199.7 ± 2.21

TABLE III
RAW TIME TO PROCESS AN RDP PACKET. LAPTOP: 1200MHZ PENTIUM 3, 512MB RAM. IPAQ: COMPAQ IPAQ 3850 206MHZ INTEL STRONG ARM 32-BIT RISC PROCESSOR, 64 MB RAM

It is important to note however, that only the routing control messages between nodes are subject to signing/verifying. Data packets exchanged between nodes after a route has been set up are not processed by `arand` in any way; they do not have attached certificates and are not signed. Once a route is set up, the routing daemon is out of the picture until that route becomes invalid and is needed again.

B. ARAND Performance

We have conducted two types of tests to determine the overhead of using certificates and signatures in ARAN. These tests include measurements of raw processing time per routing packet for different key sizes, and measurements of the average route acquisition latency.

We note that the energy cost of cryptographic operations could be of some concern, particularly in resource-constrained mobile devices. However, the energy consumed by wireless communication is significantly higher; a single bit transmission consumes over 1000 times more energy than a single 32-bit computation [36]. Additionally, route discovery is performed infrequently in most ad hoc networks. We therefore do not consider the energy consumption of cryptographic computations to be significant, and do not measure it in our experiments. The wireless communication overhead is quantified in section VII-C.

1) Message Processing Time: We examined the raw processing time expended at a node for an ARAN packet. Specifically, we measured the processing time required for a node to receive an RDP message from a neighbor that is not the initial sender of the RDP, verify that the certificate attached by the neighbor the message was received from is valid, verify the neighbor's signature on the message, strip off the neighbor's certificate, add its own certificate, sign the message, and then rebroadcast the message. We make the distinction between a forwarded RDP and one received from the initial sender because the former is larger since it includes the certificate and signature of the neighbor as well that of the node that originally sent the RDP: both signatures are checked, and our simulations reflect this. Measuring per node processing time on this type of message gives us an upper bound on the processing time for a routing message at each node. Hello messages and error messages (RERR) require less processing time.

We conducted this test by mirroring the sequence of function calls that are performed when an RDP message is received by `arand`; however, we do not consider the time spent performing operations on the state that is maintained in `arand` (such as looking through a list of RDPs to determine whether we have already seen a particular message). This simplified the test and allowed us to focus on the time spent on the cryptographic operations instead of state maintenance, which is negligible in comparison.

The main purpose of this performance test was to illustrate the expense of processing routing messages with two different types of devices that are likely to participate in an ad hoc network using the ARAN protocol. Table III shows our results. We measured processing time for both a laptop and a handheld computer over three different RSA key sizes: 512, 768, and 1024 bits. For both devices, increasing the key size by 256 bits roughly doubles processing time. Perhaps most striking in Table III is the difference in processing times between the laptop and the handheld. For each key size, the processing time is between 20 and 30 times slower on the handheld than on the laptop. From this, it is clear that the processing power of the nodes expected to participate in an ad hoc network can limit key sizes if routing overhead is a limiting factor. In other words, ARAN is not appropriate for low-resource devices when node mobility is high and route changes are very frequent.

2) Route Acquisition Latency: We also measured the average route acquisition latency, which is the delay from route request initiation to the receipt of a corresponding reply. The results of measuring latency in this way depend on the number and topology of network nodes. For simplicity, and because creating an elaborate topology with actual machines would be unwieldy, we created a simple topology with three nodes oriented in a straight line topology: $A \leftrightarrow B \leftrightarrow C$. The node in the middle is within range of the two end nodes, but the end nodes are not in range of each other. We measured the route acquisition latency for a route request from A to C. All routing messages are sent through the intermediate node, B.

For comparison, we have executed this for both AODV using the AODV-UIUC version 0.5 (`http://aslib.sourceforge.net`), which is an AODV daemon written to use the Ad hoc Support Library, and for ARAN using `arand` version 0.3.2. All nodes running `arand` are using version 0.9.6d of the OpenSSL library. Both routing daemons were modified to record the time when a route request is sent and when its corresponding route reply is received. Also, when a route reply is received, we disabled the actual addition of the newly discovered route to the kernel to allow us to continually request routes without restarting the daemon. On the sender node A, we ran a script that automatically generated network traffic for destination C, causing the daemon to request a route. The script then sleeps for a random number of seconds before generating traffic again. We measure route latency across three different RSA key sizes used by the nodes. However, in each case the CA signed client certificates were signed with a 1024-bit RSA key.

	Laptop→iPAQ→iPAQ avg. (ms) ± std. dev.	Laptop→Laptop→Laptop avg. (ms) ± std. dev.
arand:		
512 bit RSA key:	237.0 ± 16.8	22.2 ± 0.9
768 bit RSA key:	435.3 ± 34.8	31.9 ± 0.9
1024 bit RSA key:	729.3 ± 71.0	46.2 ± 1.4
aodvd:		
no keys used	4.2 ± 1.4	2.0 ± 0.2

TABLE IV

THE AVERAGE LATENCY TO ACQUIRE A ROUTE IN TWO NETWORKS: FROM A LAPTOP THROUGH TWO IPAQS, AND THROUGH A ROUTE CONSISTING OF THREE LAPTOPS.

We measured average route acquisition latency using two topologies consisting of different types of devices. Table IV shows average route acquisition latency for the topology where node A was a laptop, and nodes B and C were iPAQs. The last column of Table IV shows the average route acquisition latency for the topology where nodes A, B, and C are all Pentium 3 laptops. As can be seen from Table IV, in the iPAQ topology, the route acquisition latency using `arand` is between 56 and 175 times slower than `aodvd`, depending on the size of the key used on the nodes running `arand`. When laptops, which have significantly greater processing power than the iPAQs, are used, `arand` is only between 11 and 23 times slower than `aodvd`, depending on the size of the key. Even with the comparatively high route acquisition latencies experienced with the iPAQ topology, these latencies are still rather small compared with the duration of the typical connection between nodes in an ad hoc network. It is important to note that after a route is set up between two nodes, data packets exchanged between the nodes do not involve the ARAN routing daemon in any way, so this cost is incurred only once unless the route breaks. The lifetime of a route between a pair of nodes will typically be much longer than the time necessary for route acquisition in all ad hoc networks, except those with the most rapidly changing topologies. The initially higher cost to acquire a route will likely turn out to be an acceptable price to pay to ensure node authentication and prevention of modified or forged routing messages.

C. Simulated Network Performance

We performed our evaluations using the Global Mobile Information Systems Simulation Library (GloMoSim) [37]. 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. 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 ARAN and AODV, we assumed a routing packet processing delay of 1ms. This value was obtained through field testing of the AODV protocol implementation [38]. An additional processing delay of 2.2ms was added for ARAN to account for the cryptographic operations. This value was obtained through the implementation testing of ARAN, as reported in table III. Additionally, a random delay between 0 and 10ms was introduced before the transmission of a broadcast packet in order to minimize collisions. This is required since the 802.11 MAC protocol does not perform an RTS/CTS exchange for broadcast packets. Since we are working with fairly dense networks, the probability of collision of broadcast packets becomes quite high in the absence of this random delay.

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:

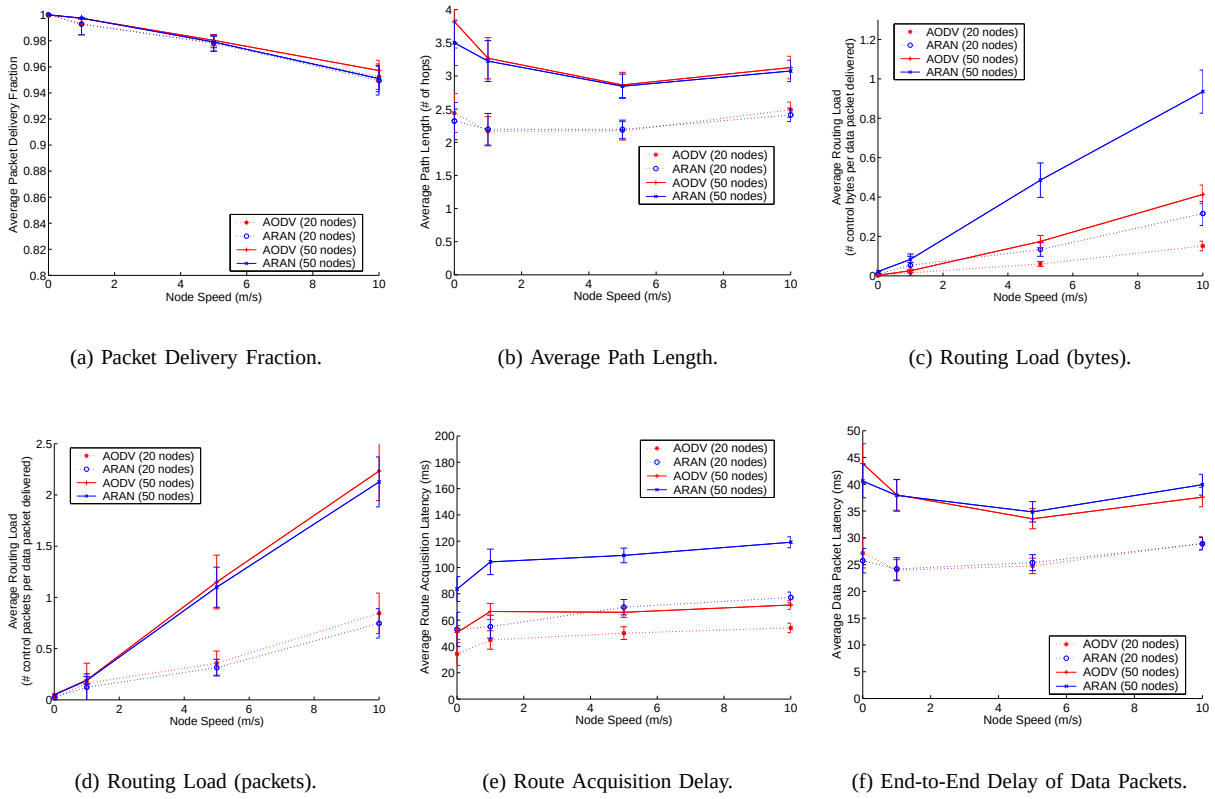
(1) **Packet Delivery Fraction:** This is the fraction of the data packets generated by the CBR sources that are delivered to the destination. This evaluates the ability of the protocol to discover routes.

(2) **Routing Load (bytes):** This is the ratio of overhead bytes to delivered data bytes. The transmission of a control byte 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. Please notice that many other secure routing protocols assume the existence of key information without accounting for the costs of distributing it: while ARAN may appear more expensive it is in part because our analysis is more complete.

(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



(a) Packet Delivery Fraction. (b) Average Path Length. (c) Routing Load (bytes).

(d) Routing Load (packets). (e) Route Acquisition Delay. (f) End-to-End Delay of Data Packets.

Fig. 1. Simulation Results.

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, and retransmission delays at the MAC layer.

1) Performance Results: Figures 1(a) through 1(f) show 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.

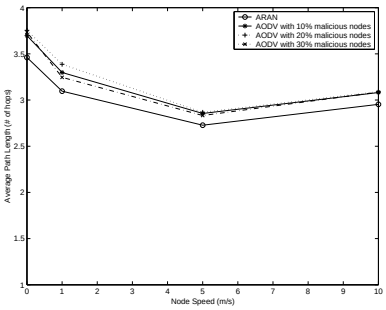
As shown in Fig. 1(a), the packet delivery fraction obtained using ARAN is 95% or higher in all scenarios and almost identical to that obtained using AODV. This suggests that ARAN is highly effective in discovering and maintaining routes for delivery of data packets, even with relatively high node mobility.

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).

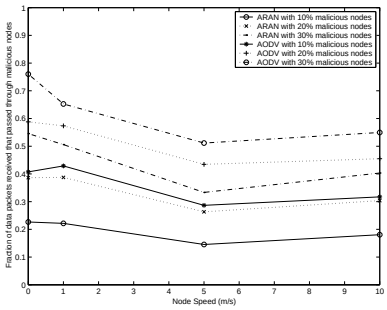
The average path length graphs are almost identical for the two protocols, as shown in Fig. 1(b). This indicates that even though ARAN does not explicitly seek shortest paths, the first route discovery packet to reach the destination usually travels along the shortest path. Hence ARAN is as effective 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.

Figs. 1(c) and 1(d) show the routing load measurements in bytes and packets, respectively. ARAN's byte routing load is significantly higher and increases to nearly 94% for 50 nodes moving at 10 m/s, as compared to 42% for AODV. This is due to the security data. However, the number of control packets transmitted by the two protocols is roughly equivalent. Fig. 1(d) shows the average number of control packets 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. 1(e) shows that the average route acquisition latency for ARAN is approximately double that for AODV. While



(a) Average path length.



(b) Fraction of data packets received that passed through malicious nodes.

Fig. 2. Effect of malicious node behavior.

processing ARAN 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. The cryptographic operations cause additional delays at each hop, and so the route acquisition latency increases.

We found through our implementation testing that the route acquisition latency using ARAN is 11 to 23 times higher than that using AODV, as reported in table IV. On the other hand, our simulations show that it is less than twice as high, as shown in figure 1(e). The reason for this discrepancy is the random delay we introduced before transmitting broadcast packets in the simulations, as described in section VII-C. Since the network used in the implementation testing is simple and not dense, the random delay was not required there. However, it is necessary in the relatively dense simulated networks for reducing collisions.

The data packet latencies for the two protocols are again almost identical as shown in Fig. 1(f). 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.

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, 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 these nodes, and could potentially be modified or dropped.

Fig. 2 illustrates the results of the experiments. As seen in Fig. 2a, the average path length increases about 10% for AODV in the presence of malicious nodes. Figure 2b 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.

D. Energy Costs

ARAN's energy expenditure is high in comparison to protocols that employ hash chains, like Ariadne. This is because ARAN spends longer time verifying signatures. These costs must be viewed in context of other energy costs of the handheld device. It is important to realize that in an ad hoc network, the handheld device must be powered at all times for successful reception of route requests. The question we must ask is, what is the additional energy spent during ARAN's cryptographic operations?

The largest energy drain on a handheld device is due to operating a wireless network interface card (NIC), as several researchers have found. From our experiments (see Table 3 in the paper), we know the running time for an iPAQ to process an RDP packet is 45ms. Many measurement studies exist on our equipment. Using values record by Kremer et al [5], Bahl et al [2], and Cho [4] as a baseline, we can provide a estimate of the costs of ARAN's cryptographic operation.

If we set the CPU power cost as 12% of 1250mW as per Kremer's measurements, then the energy usage for processing an RDP packet is $150mW \cdot 0.045sec = 6.8mJ$. Costs equal to ARAN's CPU operations will be spent by an idle radio (805mW [2]) coupled with an idle iPAQ (470mW [4]) every 5ms.

VIII. Conclusion

Popular ad hoc routing protocols are subject to a variety of attacks, which, through modification or fabrication of routing messages or impersonation of other nodes, 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.

Our protocol, ARAN, provides secure routing for the managed-open and open environments. ARAN provides authentication and non-repudiation services using 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 and experiments show that ARAN is as effective as AODV in discovering and maintaining routes. The cost of ARAN is larger routing packets, which result in a higher overall routing load, and higher latency in route discovery because of the cryptographic computation that must occur.

References

[1] S. Savage, N. Cardwell, D. Wetherall, and T. Anderson, "TCP Congestion Contorl with a Misbehaving Receiver," *Computer Communication Review*, vol. 29, no. 5, 1999.

[2] E. Shih, P. Bahl, and M. Sinclair, " Wake on Wireless: An Event Driven Energy Saving Strategy for Battery Operated Devices," in *Proc. MobiCom*, 2002.

[3] M. Stemm and R. H. Katz, "Measuring and Reducing Energy Consumption of Network Interfaces in Handheld Devices," *IEEE Transactions on Communications*, vol. E80-B, no. 8, pp. 1125–1131, 1997.

[4] S. Cho, "Power Management of the iPAQ," USC/ISI Technical Report, http://pads.east.isi.edu/presentations/misc/ sjcho-pm-report.pdf.

[5] U. Kremer, J. Hicks, and J. Rehg, "A Compilation Framework for Power and Energy Management on Mobile Computers," in *Proc. Workshop on Parallel Computing (LCPC)*, Aug. 2001.

[6] Y. Hu, A. Perrig, and D. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," in *Proc. MobiCom*, Sep. 2002.

[7] R. Canetti, D. Song, A. Perrig, and D. Tygar, "Efficient and Secure Source Authentication for Multicast," in *Proc. ISOC NDSS*, Feb. 2001.

[8] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer, "A Secure Protocol for Ad hoc Networks," in *Proc. ICNP*, Nov. 2002.

[9] C. E. Perkins and E. M. Royer, "Ad hoc On-Demand Distance Vector Routing," in *Proc. WMCSA*, Feb. 1999.

[10] D. Johnson, D. Maltz, and Y.-C. Hu, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)," *IEEE Internet Draft*, Apr. 2003.

[11] S. Murthy and J. Garcia-Lunca-Aceves, "An Efficient Routing Protocol for Wireless Networks," *ACM Mobile Networks and Applications Journal*, Oct. 1996.

[12] V. Park and M. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," in *Proc. Infocom*, Apr. 1997.

[13] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *Computer Communications Review*, Oct. 1994.

[14] J.-P. HuBaux, L. Buttyan, and S. Capkun, "The Quest for Security in Mobile Ad Hoc Networks," in *Proc. MobiCom*, Oct. 2001.

[15] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-Organized Public-Key Management for Mobile Ad hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 2, no. 1, Jan.-Mar. 2003.

[16] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing Robust and Ubiquitous Security Support for Wireless Mobile Networks," in *Proc. ICNP*, Nov. 2001.

[17] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks," *IEEE Network*, vol. 13, no. 6, pp. 24–30, 1999.

[18] S. Buchegger and J.-Y. Le Boudec, "Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes: Fairness In Dynamic Ad-hoc NetTworks)," in *Proc. MobiCom*, Sep. 2002.

[19] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," in *Proc. MobiCom*, Aug. 2000.

[20] M. Just, E. Kranakis, and T. Wan, "Resisting Malicious Packet Dropping in Wireless Ad Hoc Networks," in *Proc. ADHOC-NOW*, Oct. 2003.

[21] L. Buttyan and J. Hubaux, "Enforcing Service Availability in Mobile Ad-Hoc WANs," in *Proc. MobiHoc*, Aug. 2000.

[22] S. Sundaramurthy and E. Belding-Royer, "The AD-MIX Protocol for Encouraging Participation in Mobile Ad hoc Networks," in *Proc. IEEE ICNP*, Nov. 2003.

[23] P. Papadimitratos and Z. Haas, "Secure Message Transmission in Mobile Ad hoc Networks," in *Proc. ACM WiSe*, Sep. 2003.

[24] Y. Hu, D. Johnson, and A. Perrig, "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks," in *Proc. WMCSA*, Jun. 2002.

[25] M. G. Zapata and N. Asokan, "Securing Ad hoc Routing Protocols," in *Proc. ACM WiSe*, Sep. 2002.

[26] S. Yi, P. Naldurg, and R. Kravets, "Security-Aware Ad Hoc Routing for Wireless Networks," in *Proc. MobiHoc*, Oct. 2001.

[27] P. Papadimitratos and Z. Haas, "Secure Routing for Mobile Ad hoc Networks," in *Proc. SCS CNDS*, Jan. 2002.

[28] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," in *Proc. IEEE SNPA*, May 2003.

[29] Y. Hu, A. Perrig, and D. Johnson, "Packet Leashes: A Defense against Wormhole Attacks in Wireless Networks," in *Proc. Infocom*, Apr. 2003.

[30] B. Awerbuch, D. Holmer, and H. Rubens, "Provably Secure Competitive Routing against Proactive Byzantine Adversaries via Reinforcement Learning," John Hopkins University, Tech. Rep., May 2003.

[31] Y. Zhang and W. Lee, "Intrusion Detection in Wireless Ad-Hoc Networks," in *Proc. MobiCom*, Aug. 2000.

[32] S. Gwalani, K. Srinivasan, G. Vigna, E. Belding-Royer, and R. Kemmerer, "An Intrusion Detection Tool for AODV-based Ad hoc Wireless Networks," in *Proc. ACSAC*, Dec. 2004.

[33] M. Jakobsson and A. Juels, "Proofs of Work and Breadpudding Protocols," in *Proc. Communications and Multimedia Security*, Sep. 1999.

[34] B. Schneier, *Applied Cryptography*, 2nd ed. John Wiley & Sons, Inc., New York, 1996.

[35] V. Kawadia, Y. Zhang, and B. Gupta, "System Services for Implementing Ad-Hoc Routing: Architecture, Implementation and Experiences," in *Proc. MobiSys*, May 2003.

[36] K. Barr and K. Asanovic, "Energy Aware Lossless Data Compression," in *Proc. MobiSys*, May 2003.

[37] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla, "GlomoSim: A Scalable Network Simulation Environment," UCLA, Tech. Rep. CSD Technical Report #990027, 1997.
[38] I. D. Chakeres and E. M. Belding-Royer, "The Utility of Hello Messages for Determining Link Connectivity," in *Proc. WPMC*, Oct. 2002.