Department of Computer Science
George Mason University Technical Reports

4400 University Drive MS#4A5
Fairfax, VA 22030-4444 USA
http://cs.gmu.edu/ 703-993-1530

EPIC: Efficient Path-Independent Capabilities in Mobile Ad Hoc Networks - Design, Performance, and Security

Eric Swankoski
eswankos@gmu.edu

Sanjeev Setia
setia@gmu.edu

Technical Report GMU-CS-TR-2013-3

Abstract

In this paper, we investigate the issues that arise in the use of network capabilities to facilitate DoS prevention and mitigation in mobile ad hoc networks. Most proposals for capability-based protocols in wired networks depend upon routers on the path between the sender and receiver maintaining state that enables them to verify a capability. Frequent route changes make it necessary for any capability-based protocol for MANETs to re-establish this state as efficiently as possible. We propose EPIC, a method based on reverse-disclosure hash chains to support the establishment of path-independent capabilities and show how they can be efficiently maintained in a high mobility environment. Simulation results show that EPIC provides a 21.3% increase in performance and a 38.3% increase in efficiency over route-dependent capabilities.

1 Introduction

In recent years, there has been a great deal of research on securing mobile ad hoc networks [1, 2]. Several authors have proposed novel approaches for secure key establishment, secure neighbor discovery, and secure routing, among others. An issue that still remains to be fully addressed is the problem of preventing or mitigating denial-of-service attacks launched by malicious or compromised nodes.

Consider, for example, the situation where a node that has been compromised by an adversary injects packets into the MANET with the goal of depleting the resources of the nodes relaying the packets. While researchers have proposed mechanisms for preventing outsider adversaries from launching such attacks in MANETs, these mechanisms do not prevent insider adversaries (i.e. malicious or compromised nodes that possess the cryptographic credentials to join the MANET) from launching the same attacks [3, 4]. We note that, given the limited bandwidth and resource constraints of nodes in MANETs, even a single compromised node can launch an effective DoS attack that affects the entire MANET.

In contrast, the problem of preventing DoS attacks in wired networks has received a great deal of attention [5–9]. An important component of the solutions that have been proposed is the use of network capabilities, which are tokens of authorization associated with a node or network flow that are issued by receivers to authorized senders. These tokens are embedded in each packet that is part of a network flow from the sender to the receiver. Routers on the path from the sender to the receiver drop packets that do not include a legitimate capability, thus ensuring that only authorized traffic can flow between the source and destination.

Most of the previous research on the use of capabilities has focused on wired networks; there is little to no work on how they can be used for disallowing unauthorized traffic in MANETs. In this paper, we investigate the issues that arise in the use of network capabilities to facilitate DoS prevention and mitigation in mobile ad hoc networks and introduce EPIC (Efficient Path-Independent Capabilities), an efficient capability mechanism for mobile ad hoc networks.

The use of capabilities in MANETs poses several challenges that do not arise in wired networks. First, and most importantly, routes in MANETs change frequently due to node mobility. Most proposals for capability-based protocols depend upon routers on the path between the sender and receiver maintaining state that enables them to verify a capability. Frequent route changes make it necessary for any capability-based protocol for MANETs to re-establish this state as efficiently as possible. Second, nodes in MANETs are not dedicated routers, and are likely to be much more resource-constrained than routers in a wired network.

In this paper, we propose EPIC, a method that combines reverse-disclosure hash chains, identity-based cryptography, and hop-by-hop verification to support the establishment of path-independent capabilities and to show how they can be efficiently operated and maintained in a high mobility environment. EPIC can operate as an extension to the routing protocol or as a separate network-layer mechanism, allowing for greater flexibility of implementation. We show how EPIC can be used in conjunction with three different routing protocols - AODV, OLSR, and LAR1. Simulation results show that EPIC provides as much as a 21.3% increase in performance and a 38.3% increase in over route-dependent capabilities (RDC).

The remainder of this paper is structured as follows. Section 2 discusses related work. Section 3 presents an overview of EPIC. Section 4 provides a discussion on the security of EPIC, and Section 5 describes our simulation methodology and results. Finally, Section 6 discusses our conclusions.

2 Related Work

Most of the previous research on capability mechanisms has focused on wired networks. In an early work, Anderson et al proposed a network capability mechanism based upon reverse disclosure one-way hash chains [5]. Yang et al proposed the TVA (Traffic Validation Architecture), a network architecture which extended the early work on the use of capabilities [7]. TVA adopted the approach presented in SIFF [10] whereby route-dependent network capabilities were constructed based on bits contributed by routers on the path between the sender and receiver. In [6], Wolf proposed a route-dependent capability mechanism based on Bloom filters [2].

Other work notes that capabilities themselves are susceptible to attack via denial of capability (DoC), and in a capability-enabled network a successful DoC attack is equivalent to a successful DoS attack. Parno et al proposed Portcullis, a method for securing the connection setup and capability establishment phase via requiring computational puzzle completion prior to capability request and establishment [8]. Argyraki and Cheriton argued that protecting capability establishment eliminated the need for capabilities altogether, but we assume that preventing DoC does not automatically constitute a de facto prevention of DoS [9]. Countermeasures against both DoC and DoS are required and the problems are mutually exclusive.

From the existing literature, we can establish four key requirements for capabilities: they should be unforgeable by malicious entities; routers must be able to verify capabilities; capabilities must be non-permanent; and they are granted to the sender by the receiver. It is not difficult to see how capability methods for wired networks would be unsuitable for mobile ad hoc networks. Requiring that routers be able to verify capabilities becomes extremely problematic when we consider that the nodes, which must necessarily also act as routers in MANETs, are potentially highly mobile.

Little work has been done on the topic of capabilities in MANETs. In [11] and [12], Alicherry et al make the assumption that one or more universally trusted group controller entities exist to aid in the initial bootstrapping of the network by authorizing nodes to communicate. While this approach prevents colluding attackers from creating arbitrarily large capabilities by instituting some global policy limitations, it may not be practical or even possible in all cases. Alicherry's later research formalized and implemented the approach, providing some validation for the earlier theoretical work [13]. This centralized approach requires that intermediate nodes store capability information in a local database. Also, route changes due to mobility require additional communication, making the architecture potentially inefficient for MANETs.

3 EPIC: Adapting Capabilities to Mobile Ad Hoc Networks

The traditional capability architecture was originally proposed for wired networks. MANETs typically have some combination of high node turnover, high node mobility, and node heterogeneity, all of which introduce additional complexity [14]. In this section, we discuss the challenges in using traditional capability-based protocols in MANETs and describe the EPIC approach for using capabilities. To motivate EPIC, we first provide an overview of the various steps in establishing and using capabilities in protocols designed for wired networks and discuss the challenges and modifications necessary for using capabilities in MANETs.

3.1 Operation of Capability-Enabled Protocols in Wired Networks

The main steps in the operation of capability-based protocols in wired networks are described below:

1. *Capability Requests:* Once a node determines it needs to communicate with another node, it first checks to see whether it has a capability for that node. If it does not, it issues a request. This request identifies the sender, the terms requested for the capability, and, optionally, the sender's authentication information. Some mechanisms also implement some protection against denial-of-capability attacks such that capability establishment is allowed in a timely manner [5–7, 10]. Most protocols propose piggybacking capability requests on TCP SYN packets or specialized RTS packets.

2. *Capability Response:* Typically, the destination issues capabilities in response to a request if it is willing to receive packets from the source node [5, 7, 10]. Alternatives exist where routers rather than destinations issue capabilities provided the sender can successfully answer a challenge [6].

3. *Capability Verification:* Once a sender receives a capability from the source, the capability is included in every packet it sends. Verification of capabilities in wired networks is done by the intermediate routers. It is important to note that in most of the prominent proposals for capability-based protocols in wired networks, the capability itself is closely tied to the route between the source and destination. In [7], routers tag requests with their own identifying information and as such are explicitly identified in the capability itself. In [6], the routers themselves establish the capabilities and use Bloom filters to verify them. In [10], routers verify capabilities by calculating a hash based on a combination of the router addresses as well as the source and destination addresses.

4. *Capability Maintenance and Renewal:* As a rule, capabilities should be valid only for a limited amount of time. Thus, if a capability is compromised, the damage is limited because even without detection its validity period is finite. When a capability reaches or approaches its expiration, it is renewed by the destination. Routers can verify that the new capability value and the previous value are assigned to the same communication flow.

3.2 EPIC: Motivation and Overview

As discussed above, capabilities in wired networks are typically bound to the route between the sender and the destination. In MANETs, where the route between two nodes may change frequently, this is problematic since an existing capability associated with a flow between two nodes becomes invalid once there is a route change.

We illustrate this with an example in Figure 1. Suppose that A initiates a request for a capability with F and establishes a capability along the path A,B,C,E,F. Recall that with traditional mechanisms, no consideration is given to the possibility of that route breaking and when routes change, capabilities are simply re-established along the new route. Now suppose that the route breaks as C leaves and is no longer within transmission range of B. First, B would need to detect that C is not properly forwarding packets. B then repairs the route locally and establishes a route through D. However, when D receives packets containing a capability for the flow A,B,C,E,F, it drops them as it cannot verify them. The capability is not re-established until A detects, either through explicit notification or lack of acknowledgement, that its route to F needs repair. Node A must then re-initiate the capability negotiation process and it establishes a new capability along the path A,B,D,E,F. Each time a route changes, this process repeats.

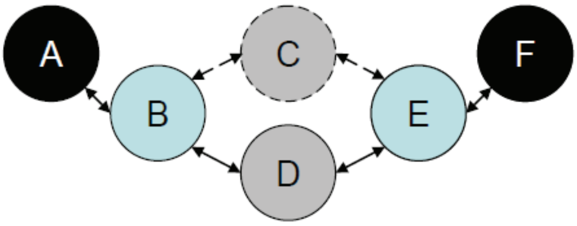

Figure 1: Illustrating Capability Maintenance Problems Associated with Node Movement

This is precisely the situation we want to avoid. In the ideal case, B detects that C is no longer reachable, repairs the route locally through D, and transmits a packet that includes a capability to F along the new route without requiring any additional information or retransmission from A. This imposes two requirements. First, the capability must not include anything that is tied to C - the capability should be path-independent. Second, the information included in the path-independent capability should be unforgeable (cryptographically secure).

Route-dependent mechanisms do not meet the first requirement, and this is a major motivation for the development of EPIC. Using route-dependent capabilities would present significant problems as nodes would be forced to renegotiate capabilities every time a route changes. In EPIC, capabilities do not include any route-dependent information. Instead, each capability includes the sender, the receiver, and a hash value, which is part of a one-way hash chain, as discussed in detail in the next section.

To ensure that capabilities are effectively unforgeable, each packet includes a field (which we call the base authoritative value) that is digitally signed. When a node receives the first packet that is part of a flow, it verifies the signature for the initial capability to check its validity. In EPIC, we propose to use digital signatures based on identity-based crypto-systems because of the performance benefits of using identity-based crypto-systems over traditional public key crypto-systems in MANETs [15,16]. Specifically, a node receiving a packet that includes a signed capability can derive the public key of the signing node from its identifier and use that public key to verify the signature. We note that EPIC can work in conjunction with a traditional public key infrastructure. However, the overhead of the protocol will be larger since a node receiving a packet with a signed capability will need to obtain the public key of the signing node before it can verify the validity of the signature.

We also note that in MANETs, it is important to minimize expensive cryptographic operations such as digital signature generation and verification. In EPIC, a node needs to verify a signature only when it receives the first packet in a flow; for subsequent packets, only a computationally inexpensive hash operation is needed.

3.3 EPIC Protocol Operation

In this section, we describe the operation of the EPIC protocol. Specifically, we discuss the steps followed by the sender and the receiver in obtaining a capability for a flow as well as the steps taken by an intermediate node when receiving a packet that is part of a capability-enabled flow between a sender and receiver.

3.3.1 Notation

We use the following notation in this section:

R, S	R and S are communicating principals: S, the source, wishes to communicate with the destination R
ID_X	The identifier for a given node X
K_X^{-1}	Node X's private key
K_X	Node X's public key, derivable from X's identifier ID_X
$\langle M \rangle_{K_X^{-1}}$	Field or message M signed with node X's private key
$E\{M\}_{K_X}$	Field or message M encrypted with node X's public key
f_h	One-way hash function used for generating a hash chain
BC	The base capability
CAP_i	Capability with sequence number i
W	Individual capability use limit

3.3.2 Capability Request

- *S: Compute* $M_1 = (RFC, T_0)$
- $S \rightarrow R$: $\langle M_1 \rangle_{K_S^{-1}}$

To initiate communication with the destination, the source sends a digitally signed and timestamped request for capability (RFC) packet containing a timestamp T_0. Depending on the application and the transport layer protocols being used, the capability request can be an independent packet or piggybacked on a transport protocol packet such as a TCP SYN packet. If an on-demand routing protocol such as AODV is being used, the capability request can be piggybacked on the route request (RREQ) packets. There are two security concerns with regards to initial capability requests. First, successful denial-of-capability attacks are effectively denial-of-service attacks and thus we need to protect capability setup. Fortunately, we can adapt methods similar to those proposed for wired networks [5–7, 10]. Second, if requests do not carry digital signatures, then any node can request a capability for any other node. Initial requests thus require authentication. We note that such a requirement also serves as motivation to reduce the incidence of capability negotiation.

3.3.3 Capability Response

- *R: Compute* CH_n *and* CH_0
- *R: Compute* $BC = (T_0, T_{CV}, ID_S, ID_R, CH_0)$
- $R \rightarrow S$: $E\{\langle BC \rangle_{K_R^{-1}}, CH_n\}_{K_S}$

When a node R receives an authenticated capability request and is willing to accept packets from the source node S, it will calculate the base capability BC and securely transmit it to the source along with additional information as described below.

EPIC is based upon the use of reverse-disclosure one-way hash chains. The receiver calculates a value CH_n based on a one-way hash function f_n, which takes as input a unique combination of the sender and receiver identification ID_S and ID_R, the initial timestamp T_0, and a nonce R_0 as shown below. Subsequently, CH_0 is calculated using n successive applications of a one-way hash function f_h to produce a sequence of values $(CH_n, CH_{n-1}, ..., CH_1, CH_0)$.

$$CH_n = f_n(ID_S, ID_R, T_0, R_0)$$

$$CH_0 = f_h(f_h(...(f_h(CH_n))))$$

The base capability BC is composed of the parameters associated with the network flow for which the capability was requested - namely the node identifiers ID_R and ID_S, the initial timestamp T_0, the validity time period for the capability T_{CV}, and the anchor of the one-way hash chain CH_0. Further, BC is digitally signed by R, so that any node that receives BC will be able to verify its authenticity using the public key of R. Note that in this paper, we assume the use of identity-based cryptosystems so the public key of R can be derived from its identifier.

Node R then transmits its response to S, which includes both BC and CH_n. This message is digitally signed by R and encrypted with the sender's public key K_S. This ensures that the sender is the only entity with access to the capability.

3.3.4 Protocol Operation: Authorized Packets

- *S: Compute Capability Hash Values (*CH_n, CH_{n-1}, ..., CH_0*)*
- *S: Compute Individual Capabilities* $CAP_i = \{BC, CH_i, i\}$
- $S \rightarrow R$: *Data Packet* $|$ CAP_i
- *Intermediate Nodes with no prior knowledge of BC: Verify authenticity of BC with* K_R*. Verify that* $CH_0 = f_h^i(CH_i)$ *using* CH_0 *included in BC. Check that capability has not expired using* T_0 *and* T_{CV} *included in BC. Cache BC and* CH_i*.*

- *Intermediate Nodes with cached BC: Verify $CH_{i-1} = f_h(CH_i)$. Check that capability has not expired.*

After receiving the base capability BC and CH_n, node S can calculate its own per-packet capabilities. First, it uses the value CH_n to mint exactly n capability hash values. Once the sender S has these n values, it can construct its per-packet capabilities for the next (nW) packets, associating each successive capability with W packets. Each capability is represented by the value CAP_i and consists of the base capability BC, the current hash value CH_i, and the sequence number i.

The basic idea behind this approach is that the hash values CH_i are disclosed in reverse order of their generation. A node that obtains CH_i will not be able to derive CH_{i+1} because of the one-way aspect of the hash function, but will be able to verify that CH_i is part of a hash chain if it has cached a previously disclosed hash value CH_k, where $0 < k < i$.

Normal operation of EPIC does not require any knowledge of the underlying routing protocol. The process for verifying a capability is identical regardless of whether a node is receiving the first packet in an authorized flow or it is receiving a packet from an established flow as a router in a new route resulting from a route repair by the underlying routing protocol. Intermediate nodes upon first receipt of a packet belonging to a flow must verify the included capability by verifying the authenticity of the base capability BC using R's public key and, if necessary, performing some number of hash calculations to ensure that the included hash value CH_i is part of the same hash chain terminating with the value CH_0 included in BC.

Once a node has received and verified a capability, for subsequent packets it need only verify that the current packet capability hash value CH_i is part of the same chain as the signed capability CH_0. The sequence number i can be used along with the time values T_0 and T_{CV} to determine the approximate number of hash computations required to ensure that CH_i is a predecessor of CH_0 in the hash chain.

As discussed earlier, in EPIC, we propose to use digital signatures based on identity-based cryptosystems. Specifically, a node receiving a packet that includes a signed capability can derive the public key of the signing node from its identifier and use that public key to verify the signature. We note that EPIC can work in conjunction with a traditional public key infrastructure if the base capability BC includes the certificate for R's public key, where the certificate is signed by a well-known certification authority.

The inclusion of the initial timestamp T_0 and the capability validity period T_{CV} prevents malicious nodes from replaying the capability after the time period specified by T_{CV}, which represents the length of time any particular capability value can be considered valid. With unique per-packet authorization, a capability will be valid until either it reaches its use limit W or its validity period expires, whichever comes first. In a static route, an intermediate node will have to perform no more than one hash computation to verify that the current hash value is the immediate predecessor of the prior known value in the chain. We assume that nodes will cache capability information to reduce redundant computations. We make the assumption that all nodes in the network and temporally synchronized within some error margin δ such that if the current absolute time is considered to be T_a, then any node in the network will have its own time T_X such that $(T_a - \delta) \leq T_X \leq (T_a + \delta)$. In this manner, at most one additional hash calculation is required in the event the current time T_0 falls within 2δ of the beginning or end of a capability validity period.

3.3.5 Capability Maintenance: Normal Operation

The only maintenance required for EPIC capabilities is the periodic renewal as capabilities approach the end of their validity period. The protocol interaction for maintenance between senders and receivers is identical to that of the initial negotiation. We require that requests for renewed capabilities take place within some defined but arbitrary time period prior to their expiration, thus preventing a sender possessing a valid capability for some time period T_x to request a capability for time period T_{x+1} at the start of time period T_x. Capabilities should not be stockpiled, and the use of unique per-packet capabilities helps prevent this. Maintenance request messages can be appended to capability-enabled data messages independent of the underlying routing protocol and are only valid if the sender possesses a valid capability.

4 Security Analysis

4.1 Unauthorized Interception

Capability information, including the values used as the terminal value of the hash chain and those used to generate the hash chain, are transmitted both signed and encrypted from the destination to the source. Without breaking the cryptographic methods, the only nodes with access to the means to mint capabilities are the sender and the receiver of a given negotiation. For a malicious node to intercept a capability and use it for its own ends, it would have to perform a complete node takeover (up to and including physical compromise). Without this, it has no method to recover the node's private key, and as such it cannot decipher the values used to create capabilities. Capability values are transmitted in the open and this is necessary to support the requirement of universal verification. While a capability is bound to a specific source and destination, it is not bound to any given route. Co-opting the capability would only allow an attacker authorized communication with the established destination. For a node to use this capability for its own ends, it must be able to successfully spoof a source, including the IP and MAC addresses, it must be able to break the hashing scheme, and it must be able to modify messages without being detected.

EPIC is resistant to this sort of attack because the combination of factors that would need to occur to allow this attack is essentially impossible. In particular, if a node can successfully spoof the source and modify messages but it cannot break the hashing scheme (a reasonable assumption based on the cryptographic strength of a given algorithm, especially over the relatively short lifespan of the capability token), then it can send at most one authorized packet with that capability. If it cannot modify packets or spoof the source, then it can send only the original packet in an attempted authorized flooding attack. We note that these attacks are detectable.

4.2 Flooding Attacks

Given the extreme difficulty associated with attempted compromise of a known capability, an attacker might instead try to create havoc in the network by flooding authorized packets. Since capabilities are universally verifiable, any node that receives a capability-enabled packet might attempt to verify and forward the packet. The broadcast nature of the medium allows nodes not on the intended route to both hear and understand these packets just as it allows malicious nodes to forward these packets to their neighbors, causing unnecessary computation for the receiving nodes as well as unnecessary delays for authorized traffic.

However, we note that EPIC allows each capability to be used a finite number of times, represented by some value W. This value represents a tradeoff between security and convenience. At $W = 1$, if a node were able to successfully modify a message and send it to the destination (or a node prior to the destination also used by the legitimate sender), it could prevent the legitimate sender's packets from being recognized. Increasing the value of W limits or eliminates this inconvenience to the legitimate source at the cost of increasing unauthorized traffic. In the worst case, if a malicious node has n neighbors not on the expected route of a capability, then it can cause at most n unnecessary route requests and at most (nW) unnecessary forwarded packets. Each subsequent node will forward a maximum of W packets. For each subsequent packet, the one-hop damage is limited to at most (nW) unnecessary authorized packets. Since W can be a very small number and most nodes have a limited number of one-hop neighbors, the amount of extraneous traffic generated is quite small.

It is assumed that legitimate nodes will properly mark the packets they forward. Thus, any legitimate intermediate node on a route up to and including the destination can detect such an attack and inform one or more neighbors to reduce the priority for packets containing a given capability (or stop forwarding them altogether). If that node stops receiving packets on another route, it can reinstate the priority or authorization for a different route, minimizing the inconvenience to the legitimate sender.

If colluding attackers are able to utilize a wormhole to inject packets at a distant location, they may use strategic locations to conduct an effective DoS attack against a legitimate sender by retransmitting an overheard or intercepted capability. In Figure 2, the sender S communicates with the receiver R. Locally, attackers X and Y are present and inject duplicate capability-enabled packets to nodes C and D. Nodes on the actual route are more likely to reach the forwarding limit and thus block packets - in this case, B can reject duplicate packets. This is because it is more difficult to find a disjoint route locally. However, if X can use a wormhole to transmit its packets to Z, a distant colluding attacker, then Z can inject packets to its neighbors E and F (who would ordinarily never see this capability). This allows attackers to disrupt the normal operation of the network regardless of whether the inject packets ever reach the destination or not. As a potential solution to this problem, partial route dependence could be employed. Since the vast majority of route repairs occur because of the loss or replacement of a single node, introducing partial route dependence could prevent disjoint routes from being used by mandating that a certain percentage of nodes involved in the original route be present. While this necessarily requires greater overhead as capabilities would need to be renewed more often, it mitigates multiple types of capability-enabled packet injection attacks.

An adversary able to successfully impersonate the legitimate sender could fool the receiver into accepting its hijacked packets rather than the legitimate packets. As mentioned before, this attack is detectable. Unless attacking nodes control an entire alternate route and are able to modify packets without detection, the destination will be able to analyze packet routing information and recognize that it is receiving packets from the same source on two simultaneous different routes. However, mitigation can be complicated as it is difficult to identify which routes are legitimate. Furthermore, it is difficult to determine whether a source is a malicious node deliberately exploiting multiple paths or it is a legitimate node under attack. Addressing the issue at the routing level is an attractive (and potentially necessary) option if the underlying routing protocol is vulnerable to such attacks. While wormhole attacks are possible, they are difficult to engineer and represent a fairly complex and impractical attack, and we note that integrating EPIC with secure routing protocols would make detection and mitigation of wormhole attacks more efficient. We note, however, that wormhole attacks represent an attack against the routing protocol and not nec-

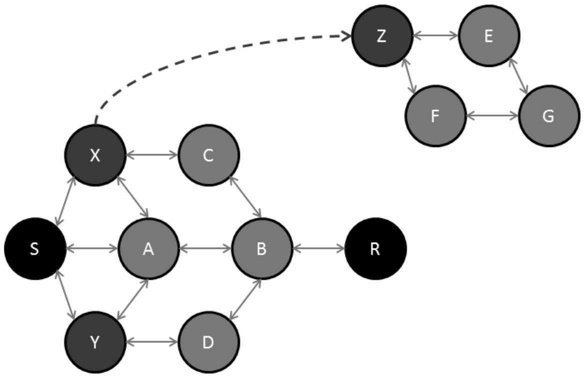

Figure 2: Flooding Attacks: Local and Distant Packet Injection

essarily against EPIC. Regardless, mechanisms such as temporal or geographical packet leashes can be used to detect and mitigate such attacks [17].

4.3 Attacks on Capability Establishment

In ad hoc networks that employ a deny-by-default access policy, preventing legitimate nodes from obtaining authorization effectively creates an absolute denial-of-service attack. We consider neither such denial-of-capability attacks nor their countermeasures in this paper, instead noting that existing measures (such as Portcullis) are adaptable to MANETs [8].

4.4 Colluding Senders and Receivers

A particularly difficult type of attack involves two nodes working together within the established rules of the network to dominate the network. Since there is no centralized authority for issuing and verifying capabilities, it is possible that malicious nodes can issue an unlimited number of capabilities to other malicious nodes. As the number of malicious nodes increases, the overall traffic in the network increases and, in the absence of some mitigation method, legitimate traffic can be delayed or even prevented altogether. While the details of mitigation methods are beyond the scope of this paper, we note that a distributed approach based on per-source fair queuing and congestion control can help limit the impact of such an attack.

5 Performance Results

5.1 Simulation Methodology

We use QualNet version 5.0 to compare the operation of EPIC with a more traditional route-dependent capability system. QualNet is a commercially available network simulator based on UCLAs GloMoSim project, a scalable Parsec-based parallel discrete-event simulator suitable for mobile ad hoc networks. QualNet provides a wide range of statistics based on the OSI model and we make additional modifications to produce other relevant statistics about our simulation. [18] We have two main goals for simulation: first, to show that EPICs performance with regards to denial-of-service prevention is at least statistically equivalent to that of traditional route-dependent capabilities and second, to evaluate whether the theoretical advantages of EPIC result in statistically significant improvements in application performance and efficiency. For purposes of comparison, route-dependent capabilities are implemented by including route information in the packet headers and comparing them with the established route, which is included in the capability. When a route breaks, the capability is considered broken and a new request must be issued by the source. We construct our experiments using the *t*-distribution and the sample standard deviation to give results within a 95% confidence interval and validate our results. [19]

Table 1: Summary of Simulation Scenarios

Parameter	Values
Routing Protocols	AODV, OLSR, LAR1
Mobility Models	Manhattan, MMTS
MMTS Models	City, Rural, Urban
Number of Nodes	50 (Manhattan)
	158 (MMTS Rural)
	250 (MMTS Urban / City)
Simulation Area	1 km^2 (Manhattan)
	9 km^2 (MMTS)
Applications	CBR, FTP, VoIP
Simulation Time	1800 s
Number of Runs	20
Confidence Interval	95%

5.2 Simulation Variables

To evaluate the potential performance and efficiency advantages of EPIC, we simulate a range of routing protocols, mobility models, and traffic patterns. Details are listed in Table 1.

5.3 Routing Protocols

- *AODV:* The Ad Hoc On-Demand Distance Vector routing protocol is a reactive protocol that only establishes routes when it needs to support communication between endpoints. Capabilities are implemented within the routing protocol itself, providing fine-grained integration with the operation of the routing protocol. In our simulations, local route repair is an option, but we note this provides substantially more benefit to EPIC. With route-dependent capabilities, local repair only provides any benefit if the same route can be established, which is highly unlikely, When underlying routes change, RDC needs to establish a new capability while EPIC does not. Since local repair can result in less optimal routes, we expect that the average hop count for EPIC will be slightly higher than that of RDC. However, we do expect that RDC will have increased overhead as a direct result of the increased capability maintenance requirements. [20]

- *OLSR:* The Optimized Link State Routing protocol is a proactive protocol that maintains a complete picture of the network. As a proactive routing protocol, we expect that OLSR will incur performance penalties as the routing protocol overhead interferes with the operation of the application. If a route can be found and maintained, then OLSR will do so. As a result of this, capability overhead is likely to be similar between EPIC and RDC. Idle networks continue to transmit routing data even if no application-layer communication is required. [21]

- *LAR1:* The Location-Aided Routing protocol is an on-demand protocol that uses prior known location to route future packets. If no location information is available (as in the case of an initial request) a packet is flooded. Once a route is established, LAR1 will maintain that route and subsequent packets will be source routed. We include LAR1 as an interesting comparison because the source routing aspect is acting against the inherent advantages of path-independent capabilities. Because routes are fairly short, however, this provides some mitigation against the inherent drawbacks of RDC. [22]

5.4 Mobility Models

- *Manhattan:* Designed to simulate vehicle and pedestrian traffic in a city setting, nodes move along a restricted grid-like pattern at randomly determined speeds and either turn only at the intersection of horizontal and vertical grid lines or continue straight. Manhattan mobility simulations are conducted with 50 nodes in a 1000m x 1000m (1 km^2) area. [23]

- *MMTS:* The Microscopic Multi-Agent Traffic Simulator (MMTS) is designed to simulate realistic vehicular movement patterns over real-world regional road maps in Switzerland. Three different models are generated from the Generic Mobility Simulation Framework (GMSF): City, which represents relatively high density and lower velocity over a smaller portion of the total map; Rural, which represents lower density and higher velocity; and Urban, which represents an approximate middle ground between the previous two. MMTS models are conducted with either 250 nodes (City or Urban) or 158 (Rural) over a 3000m x 3000m (9 km^2) area. [24]

5.5 Applications

- *CBR:* One node acts as the CBR client and one node acts as the CBR server. For IP-integrated capabilities, two identical sessions are established to facilitate easier capability establishment and maintenance (two nodes participate, with each node acting as both client and server). The client sends packets for a 10 minute interval with a 250 ms inter-packet delay for a total of 2400 packets. This represents relatively low channel utilization over a long period of time, allowing the effects of mobility greater significance.

- *FTP:* One node acts as the FTP server and one node acts as the FTP client. The server sends 1000 packets, each 1000 bytes in length, to the client continuously until the session is complete. This represents maximal channel utilization over a potentially long period of time, allowing varying degrees of significance to both channel contention and mobility.

- *VoIP:* One node acts as a call initiator and one node acts as the call receiver. Calls last for 10 minutes, with each node talking for an interval determined by an exponential distribution with a 20-second mean. This represents moderate but relatively bursty channel utilization over a fixed period of time, again allowing varying degrees of significance to both channel contention and mobility.

5.6 Simulation Results

We are primarily concerned with two broad categories: efficiency and performance. For efficiency, we define the capability efficiency as the percentage of total data (application data in addition to capability-related overhead) transmitted by a client or server that represents actual application-layer data. For performance, we use one key metric for each application: average per-packet delay for CBR traffic, throughput for FTP traffic, and quality of service for VoIP traffic. While the metrics for CBR and FTP are straightforward, most VoIP applications use a statistic called Mean Opinion Score (MOS) to calculate voice quality. The MOS is calculated from the available simulation statistics and is based on the ITU-T P.563 standard, defined in [25]. In particular, we create a three-factor metric based on packet delivery ratio, average end-to-end delay, and capability overhead. In voice quality terms, this correlates with hearing everything the other party says as quickly as possible. We note that samples are weighted based on how much data they transmit with respect to the overall amount of data transmitted.

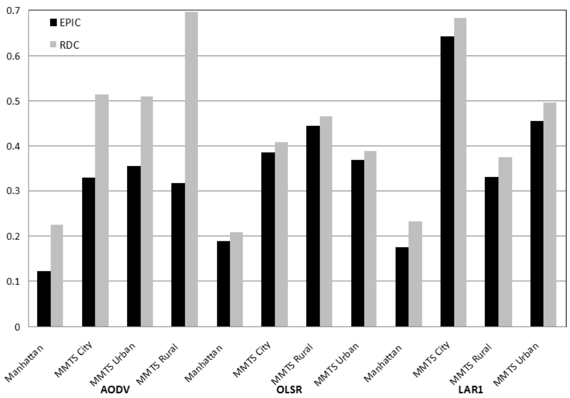

Figure 3: Average CBR Per-Packet Delay (s)

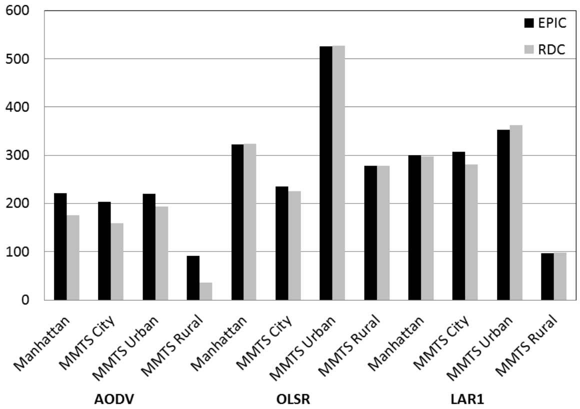

Figure 4: Average FTP Throughput (kbps)

5.6.1 Performance

EPIC exhibits an average performance increase of 21.3% across the three applications. For CBR applications (Figure 3), the average per-packet delay increases from 0.34 seconds for EPIC to 0.43 seconds to RDC, an increase of 20.9%. For FTP applications (Figure 4), EPIC's average throughput is 262.8 kbps compared to 246.5 kbps for RDC, an increase of 6.6%. For VoIP applications (Figure 5), EPIC maintains an MOS of 2.62 to 1.92 for RDC, an increase of 36.4%. Within the results, the highest performance is generally achieved within the Manhattan model, likely due to the smaller size and more regular node mobility. MMTS models exhibit greater variability and lower performance overall, due to a wide range of conditions (very high movement speeds and both very dense and very sparse areas). We note that all calculated performance advantages for EPIC, even the relatively small increase seen with FTP, are statistically significant at the 95% confidence level. Thus, the general repeatability and consistency of the results give us confidence that EPIC would likely maintain a statistically significant performance advantage across almost any application or mobility model.

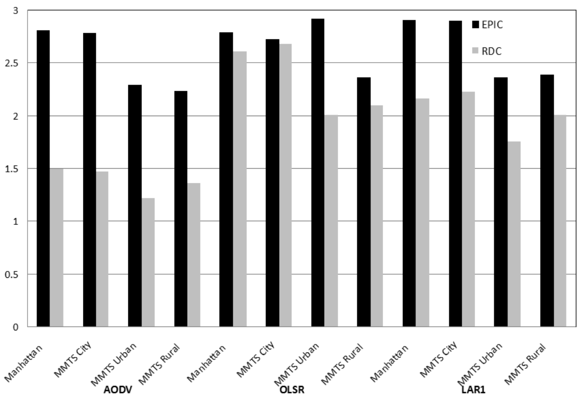

Figure 5: Average VoIP Mean Opinion Score (MOS)

5.6.2 Efficiency

As both mobility and contention increase, transmission errors occur and routes break. This leads to increased capability-associated overhead, particularly with the RDC model since the capability needs to be reestablished if the route changes. The overall efficiency numbers are illustrated in Figure 6. On the whole, EPIC achieves an 83.0% capability efficiency while RDC achieves a 60.0% capability efficiency, an improvement of 38.3%. Within these results, we note some important findings. First, despite both being on-demand routing protocols, AODV has the lowest efficiency while LAR1 has the highest. Since AODV and LAR1 only establish and maintain routes on demand, they are likely to encounter more maintenance overhead. However, the location-aware aspect of LAR1 allows it to more efficiently operate in large and high-mobility environments. Since OLSR maintains routes even when no application-layer communication is required, the overall routing overhead greatly increases while capability overhead remains relatively low. Also, efficiency decreases as the inter-packet interval increases. CBR applications send packets regularly but at a much lower rate than FTP. VoIP sends packets in bursts, but with potentially long pauses between bursts. This allows mobility effects to become more significant.

5.6.3 Case Study: Effects of Route Dependence on FTP

We have illustrated statistically significant advantages for EPIC over RDC in terms of both performance and efficiency. Recall that one of the key requirements for capabilities was non-permanence. This can be effectively achieved in one of two ways - deterministically (via time constraints) or probabilistically (via route dependence). Figure 7 illustrates the average capability lifetime for the route-dependent model. For our simulations, the mean value was 22.08 seconds and the distribution appears to be exponential. If we assume a

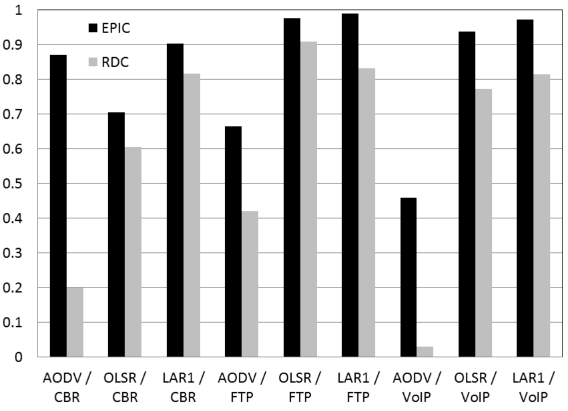

Figure 6: Average Capability Efficiency Percentage

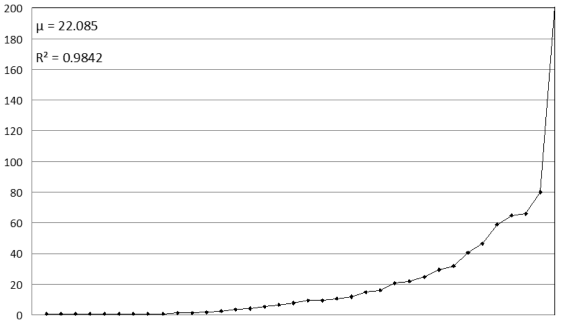

Figure 7: Average RDC Capability Lifetime

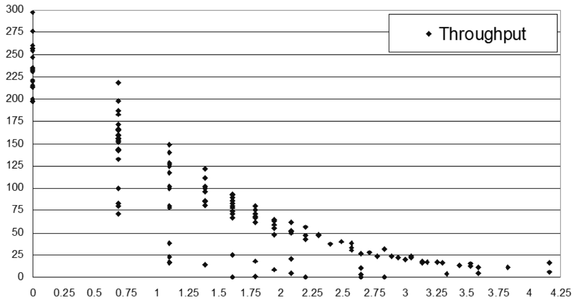

Figure 8: Throughput (kbps) vs. Capability Breaks

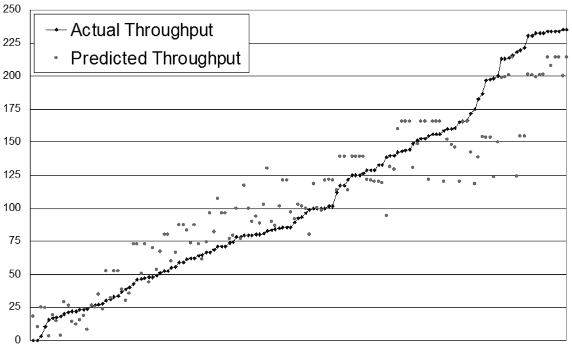

Figure 9: Actual and Predicted FTP Throughput (kbps)

capability lifetime above 30 seconds, then a significant majority of route-dependent capabilities would break prior to expiration. In our simulations, we do not assume that the capabilities expire during the simulation (the validity period is at least 1800 seconds). Under these constraints, the relatively short lifespan of route-dependent capabilities results in substantially higher overhead (both in terms of signature-containing capability packets and total capability overhead size). We note, however, that there does not appear to be a strong correlation between the average lifetime of a capability and the associated increase in overhead. Route dependence thus impacts both performance and efficiency by prematurely terminating a capability and subsequently requiring a substantial amount of additional overhead to correct the problem, which is reflected in EPIC's improvements in both performance and efficiency. Route-dependent models incur on average more than 11,451 additional overhead packets per trial, a 3,481% increase in the number of capability-enabled overhead packets and a corresponding increase in transmitted overhead data.

Intuitively, we can see that the performance of EPIC is increased relative to RDC because route dependence introduces additional delays as capability re-establishment takes place. Ideally, both the lifetime and efficiency of a capability should be maximized - that is, the amount of time a capability is valid and the amount of data it can transmit while it is active. A long-lived capability does not achieve much if the throughput is low, while a short-lived capability is less problematic if the throughput is high. Thus, a capability's lifetime is most accurately measured in bytes. Subsequently, we can assume based on what we know that throughput should decrease based on these two factors. When we look at the simulation results for FTP, we can see a correlation between throughput and the number of capability breaks per session. While throughput varies for a given number of breaks, it does decrease as the number of breaks increases. This is illustrated graphically in Figure 8. Based on this, we can also use the two aforementioned factors to develop a predictive model for throughput. Regression testing illustrates a statistically significant correlation between throughput and both factors as illustrated in Figure 9.

5.6.4 Authorized Flooding: FTP / Manhattan / AODV

For this simulation, we start with the Manhattan mobility model using the AODV routing protocol. We use FTP as the base application, but instead use a highly structured session that sends exactly 1000 items of 1000 bytes in size. For the attack model, we assume that malicious nodes are only able to flood unmodified capability-enabled messages. For each scenario, we define the tuple $\{x, W\}$ where x is the total number of malicious nodes in the network and W is a global parameter representing the maximum number of times a given capability value will be accepted (the forwarding limit). In general, a given capability is accepted as many as W times per node unless the next value in the chain is accepted. Thus malicious nodes are limited by the global parameter W as well as the legitimate sending activity of the source. Figure 10 illustrates the performance of EPIC under various attack scenarios. Upper and lower bounds for the confidence interval are also given, indicating a relatively straightforward relation between both the forwarding limit and the number of attackers and throughput loss. Throughput degrades as the legitimate source contends with malicious nodes for limited resources, with a stronger correlation seen between the number of attack nodes and the throughput as compared to the forwarding limit. With a limited degree of malicious nodes and a limited forwarding limit, EPIC can retain reasonable performance even during an authorized flooding attack. At the maximum level simulated - $\{16, 16\}$ - EPIC retains 11.8% of its base legitimate throughput even though it represents as little as 0.39% of the overall authorized traffic. Figure 11 illustrates a similar pattern of degradation for efficiency, with EPIC retaining 27.3% of its baseline efficiency even under a $\{16, 16\}$ attack pattern. Again, upper and lower bounds are given, indicating that although there is some variability attacks definitely do have a significant effect on efficiency. Figure 12 illustrates the percentage of malicious packets blocked, overall as well as the percentage that reaches the destination. On average, 98.1% of all malicious packets are blocked or rejected. Of these, 45.9% are blocked by intermediate nodes, while 54.1% of the malicious packets reach the destination before being rejected. (In absolute terms, this means that 1.9% of flooded packets are accepted as legitimate, 45.0% are blocked by intermediate nodes, and 53.1% reach the destination before being rejected as duplicates.) As the number of attackers increases, more packets reach the destination before being blocked. This is because the packets are likely to find a disjoint route (and thus a node that has not seen or will not see the original capability before reaching the forwarding limit).

6 Future Work and Conclusions

In this paper, we have presented EPIC, Efficient Path-Independent Capabilities, which represents a significant improvement in efficiency and performance on the existing unicast capability methods. EPIC is based on reverse-disclosure hash chains, which allow two communicating entities to efficiently maintain secure communications over multiple time periods on a unique per-packet basis. Each packet includes a route-independent capability that can be verified by any node in the network without requiring additional communication, mitigating the negative effects of mobility in ad hoc networks. In addition, EPIC also avoids the problem of public key exchange by having digital signatures and authentication methods dependent on a unique identity (such as a network address). These factors allow EPIC to achieve significant efficiency improvements by limiting the negative effects of mobility and providing for per-packet authorization while minimizing digital signature verification as only the first received packet in a capability-enabled flow must be verified. Results indicate a statistically significant increase in performance and a significant reduction in routing-associated and capability-associated overhead.

With results thus far being encouraging, future work will be focused on three primary areas. The first is the simulation and evaluation of more complex and tar-

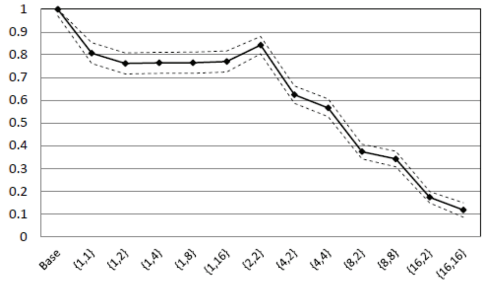

Figure 10: Normalized FTP Throughput under Multiple Attack Scenarios

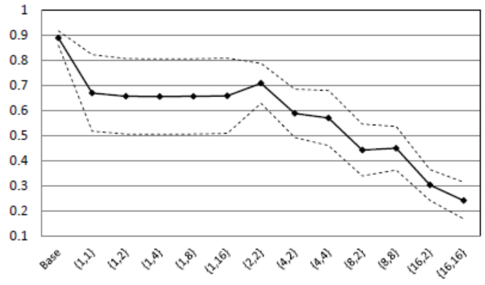

Figure 11: Routing and Capability Efficiency under Multiple Attack Scenarios

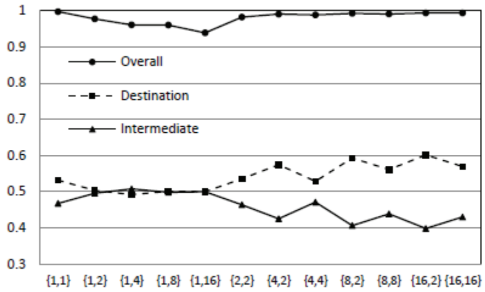

Figure 12: Packet Blocking, Overall / Destination / Intermediate

geted attack scenarios such as colluding attackers, multipath exploitation, and false capability injection. The second area will be the extension of the overall architecture to support mixed-priority traffic ranging from blacklisted traffic to multiple levels of authorized traffic. The third area will be the extension of EPIC capabilities to multicast applications. Since unicast routing is markedly different from multicast routing, we expect additional challenges in extending path-independent capabilities to multicast environments As a general goal, we also plan to expand EPIC into a more complete deny-by-default architecture rather than a simple expansion of existing routing protocols to support network capabilities.

References

[1] D. Djenouri, L. Khelladi, and A. Badache, "A Survey of Security Issues in Mobile Ad Hoc and Sensor Networks," *IEEE Journal of Communications Surveys*, vol. 7, no. 4, Q4 2005.

[2] B. Wu, J. Chen, J. Wu, and M. Cardei, "A Survey of Attacks and Countermeasures in Mobile Ad Hoc Networks," in *Wireless Network Security*, Y. Xiao, X. Shen, and D. Du, Eds. Springer, 2007.

[3] S. Zhu, S. Xu, S. Setia, and S. Jajodia, "LHAP: A Lightweight Hop-by-Hop Authentication Protocol for Ad-Hoc Networks," in *Proceedings of the 23rd International Conference on Distributed Computing Systems*, 2003.

[4] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks," *ACM Transactions on Sensor Networks*, vol. 2, no. 4, November 2006.

[5] T. Anderson, T. Roscoe, and D. Wetherall, "Preventing Internet Denial-of-Service with Capabilities," in *Proceedings of the 2nd Workshop on Hot Topics in Networks*, November 2003.

[6] T. Wolf and K. Vasudevan, "A High-Performance Capabilities-Based Network Protocol," in *Proceedings of the 5th IEEE Workshop on Secure Network Protocols*, October 2009.

[7] X. Yang, D. Wetherall, and T. Anderson, "A DoS-Limiting Network Architecture," in *Proceedings of the 11th ACM Special Interest Group on Data Communications Conference*, August 2005.

[8] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y. Hu, "Portcullis: Protecting Connection Setup from Denial-of-Capability Attacks," in *Proceedings of the 2007 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, October 2007.

[9] K. Argyraki and D. Cheriton, "Network Capabilities: The Good, the Bad, and the Ugly," in *Proceedings of the 4th Workshop on Hot Topics in Networks*, November 2005.

[10] A. Yaar, A. Perrig, and D. Song, "SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks," in *Proceedings of the 2004 IEEE Symposium on Security and Privacy*, May 2004.

[11] M. Alicherry, A. Keromytis, and A. Stavrou, "Deny-by-Default Distributed Security Policy Enforcement," in *Proceedings of the 5th International ICST Conference on Security and Privacy in Communication Networks*, September 2009.

[12] ——, "Evaluating a Collaborative Defense Architecture for MANETs," in *Proceedings of the IEEE Workshop on Collaborative Security Technologies*, December 2009.

[13] M. Alicherry and A. Keromytis, "DIPLOMA: Distributed Policy Enforcement Architecture for MANETs," in *Proceedings of the 4th International Conference on Network and System Security*, September 2010.

[14] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing*, 1996.

[15] J. Baek, J. Newmarch, R. Safavi-Naini, and W. Susilo, "A Survey of Identity-Based Cryptography," in *Proceedings of the 2004 Australian Unix Users Group Annual Conference*, September 2004.

[16] P. Michiardi and R. Molva, "IDHC: ID-Based Hash Chains for Broadcast Authentication in Wireless Networks," Institut Eurecom, Tech. Rep., July 2004.

[17] Y. Hu, A. Perrig, and D. Johnson, "Packet Leashes: A Defense against Wormhole Attacks in Wireless Networks," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, April 2003.

[18] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks," in *Proceedings of the 12th Workshop on Parallel and Distributed Simulation*, May 1998.

[19] R. Jain, *The Art of Computer Systems Performance Analysis - Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley, 1991.

[20] C. Perkins and E. Royer, "Ad Hoc On Demand Distance Vector Routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, February 1999.

[21] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, “Optimized Link State Routing Protocol for Ad Hoc Networks,” in *Proceedings of the 2001 IEEE International Multi-Topic Conference*, August 2001.

[22] Y. Ko and N. Vaidya, “Location-Aided Routing (LAR) in Mobile Ad Hoc Networks,” *Wireless Networks*, vol. 6, no. 4, July 2000.

[23] F. Bai, N. Sadagopan, and A. Helmy, “IMPORTANT: A Framework to Systematically Analyze the Impact of Mobility on Performance of Routing Protocols for Ad Hoc Networks,” in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, March 2003.

[24] R. Baumann, F. Legendre, and P. Sommer, “Generic Mobility Simulation Framework (GMSF),” in *Proceedings of the 1st ACM SIGMOBILE Workshop on Mobility models*, May 2008.

[25] L. Malfait, J. Berger, and M. Kastner, “P.563 - The ITU-T Standard for Single-Ended Speech Quality Assessment,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 6, November 2006.