Distributed Firewall For MANETs

Mansoor Alicherry Angelos D. Keromytis
Department of Computer Science
Columbia University

Angelos Stavrou
Department of Computer Science
George Mason University

***Abstract*— Mobile Ad-hoc Networks (MANETs) are increasingly used in military tactical situations and in civil rapid-deployment networks, including emergency rescue operations and *ad hoc* disaster-relief networks. The flexibility of MANETs comes at a price, when compared to wired and basestation-based wireless networks: MANETs are susceptible to both insider (compromised node) and outsider attacks due to the lack of a well-defined perimeter in which to deploy firewalls, intrusion detection systems, and other mechanisms commonly used for network access and admission control.**

In this paper, we define a distributed firewall architecture that is designed specifically for MANETs. Our approach harnesses and extends the concept of a *network capability*, and is especially suited for environments where the communicating nodes have different roles and hence different communication requirements, such as in tactical networks. Our model enforces communication restrictions among MANET nodes and services, allowing hop-by-hop policy enforcement in a distributed manner. We use a "deny-by-default" model where compromised nodes have access only to authorized services, without the ability to disrupt or interfere with end-to-end service connectivity and nodes beyond their local communication radius. Our simulations show that our solution has minimal overhead in terms of bandwidth and latency, works well even in the presence of routing changes due to mobile nodes, and is effective in containing misbehaving nodes.

Keywords: MANETs, Capabilities, Distributed Firewall

I. Introduction

Recent advances in low power computing and communications have led to the proliferation of handheld and portable devices equipped with wireless connectivity. These mobile wireless devices appear to be ideal for situations where fixed infrastructure would be too costly or dangerous to deploy, or has been rendered inoperable. However, because of radio power consumption, physical obstacles, and channel capacity, a mobile node may not be able to reach all other nodes within a single broadcast. Therefore, to achieve end-to-end connectivity, nodes have to form mobile ad-hoc wireless networks (MANETs) which allow data to be routed through intermediate nodes. MANETs are fundamentally different than Internet because all peers act as both sources and routers using the other participants to relay packets to their final destination. Due to their flexibility, MANETs are currently employed in both military and commercial applications.

Unfortunately, not all MANET nodes are equally capable, nor can all users be equally trusted. Worse yet, mobile nodes in tactical environments run the danger of being captured or malfunctioning. In current systems, even a small number of misbehaving nodes can successfully compromise the entire MANET: malicious peers can transmit excessive packets, exhausting all network and power resources of other participants through a Denial of Services (DoS) attack.

In traditional networks, malicious nodes and traffic are kept away from a set of nodes belonging to an organization or a group using *firewalls*. This is feasible because of the existence of a well defined network perimeter. All incoming and outgoing traffic needs to transit through these firewall nodes, which enforce the policies at the perimeter. Even within the perimeter smaller sub-groups can have more stringent policies by deploying their own firewalls. Unfortunately, the concept of a network perimeter does not exist in MANETs, and policies need to be enforced in a distributed manner while taking into consideration node mobility.

To amend this, we enforce trust relationships and traffic accountability between mobile nodes by introducing a novel policy enforcement architecture that is designed specifically for MANETs. We extend the network capability framework [23], [3] and we tailor it for an ad-hoc resource-constrained mobile environment. Capability is a token of authority that has associated rights. In our model, capabilities propagate both access control rules and traffic-shaping parameters that should govern a node's traffic. We define a protocol for transmitting capabilities (which are treated as soft state) across the MANET.

Our architecture enables the enforcement of adaptive bandwidth constraints inside the network denying by default unauthorized traffic. Such policy helps to mitigate the impact of denial of service (DoS) attacks because excess or unauthorized packets are dropped closer to the attack source. Thus, we avoid unnecessary data processing and forwarding at the target node and the network itself. Moreover, nodes can only access the services and hosts they are authorized for by the capabilities given to them. Thus, compromised or malicious nodes cannot exceed their authority and expose the whole network to an adversary. Upon detection, a compromised node can be prevented from further attacking the network simply by revoking its capabilities.

To evaluate the performance of our scheme, we performed extensive simulations using GloMoSim [1]. Because GloMoSim did not include any packet checking functionality, we extended its models to adding another layer between the IP and the AODV routing where we implemented the capabilities. Since our enforcement mechanism has provable protection properties, our primary concern was the network overhead of our scheme given the cryptographic operations required. Therefore, we focused our measurements on comparing the packet latency and bandwidth with and without our system in a variety of mobility scenarios and topologies. We discovered

that our scheme imposes an 8% overhead on the end-to-end latency and a 5% drop on available bandwidth. We believe that this is not a high price to pay given that there are scenarios that the MANET becomes completely unusable even when a single node misbehaves.

The rest of the paper is organized as follows. We begin by describing the threat model in Section II. We then present the system architecture and a high-level overview of our scheme, including the security analysis, in Section III, and the details of the protocol in Section IV. We evaluate our architecture through simulation, with results given in Section V. Related work is discussed in Section VI.

II. Threat Model

In our threat model, we assume MANET environments where an adversary may be an existing node that is compromised (insider) or a malicious external node that might want to participate in the MANET. Furthermore, there may be multiple adversaries, and they can cooperate. In addition, compromised nodes may not be detected as such immediately, or ever (depending on their actions).

Our goal is to protect network resources and end-node services from denial of service attacks, and to enforce access control rules in the absence of a fixed topology. We want a node to access only the services it is entitled to; we want to restrict both the type and quantity of the service and filter the unauthorized traffic early on the path to preserve resources.

The resources needed to access a service are allocated by the *group controller(s)* of the MANET. Group controllers are nodes responsible to maintain the group membership for a set of MANET nodes and authorize multicast communications within the group. Without compromising a group server, an external node can participate in a MANET only by stealing the authorization credentials that are bound to the identity of a compromised node.

If a node is compromised, an adversary can only access the services and bandwidth that node is authorized to access. A compromised node does not have ability to disrupt or interfere with end-to-end service connectivity and nodes beyond its local radio communication radius. The nodes providing the services will receive only the traffic that node or the group server has allocated it to be received, unless the adversary is in the communication radius.

III. System Architecture

In our architecture, there is one or more pre-defined nodes that act as a *group controller*. These nodes are trusted by all the group nodes. For the purpose of simplicity and without loss of generality, we will assume that all the MANET nodes are part of a single group. A group controller has authority to assign resources to the nodes in MANET. These resources are expressed in terms of limits on the number of packets or on bandwidth rates that a MANET participant is permitted to transmit destined for a service running on another node. The resource allocation by the group controller to a node is represented using a certificate that all the nodes can verify. This resource allocation is called *forward capability*. When a node (initiator) requests a service from a MANET node (responder) using the forward capability assigned to the initiator, the responder can provide capability back to the initiator. This is called *receiver capability*, and it is generated based on the resource policy of the system.

All the nodes in the path from an initiator to a responder (*i.e.,* nodes relaying the packets) are required to enforce and abide by the resource allocation done by the group authority. The enforcement involves both firewall rules as well as bandwidth allocation of the capability. A responder node accepts the packets from an initiator only if the initiator has authorization to send in the form of a capability. An intermediate node will forward the packets from a node only if the packets have an associated capability, and if they do not violate the capability. Note that the packets do not have to physically contain the capability. The nodes should be able to identify them with the capability, as we explain later in this section.

To enforce that the packets are forwarded only if they have capability associated with it, the intermediate nodes keep track of the capabilities of the data transfer that is transiting through it. These capabilities are maintained in the *capability database*. The details of the capability database are given in Section IV.

Figure 1 gives an overview of the protocol when an initiator (user) wants to get a service from a responder (server). The initiator has a forward capability issued by the group server that authorizes the communication with the responder. The initiator sends a communication request and an optional initial data, along with its forward capability to the responder. This packet also contains a *transaction id* that the initiator will use for subsequent communication. The packet may also contain a receiver capability the initiator generates that can be used by the responder to communicate back to the initiator. Here we assume that the initiator has a routing table entry for the responder. Otherwise the underlying routing protocol will be invoked to get the route. When the packet reaches the intermediate node, it will forward the packet after validating the packet. The validation involves cryptographic verification of the capability, and verification of the constraints (*e.g. bandwidth usage, service and destination address*) specified in the capability. If the validation is successful, the intermediate node also records the capability in its capability database, along with other attributes of the packets like source and destination node address and the transaction id.

The responder, on receiving the packet verifies the capability and creates a receiver capability for the initiator. The responder sends the response to the request as well as the newly created receiver capability for the initiator. The responder also creates a transaction id for the communication, and includes it in the response. The responder also needs to include a capability, which authorizes it to communicate with the initiator. This capability can be either the receiver capability for the responder the initiator had sent in the first packet, or the forward capability issued by the group server authorizing the communication with the initiator. The intermediate node,

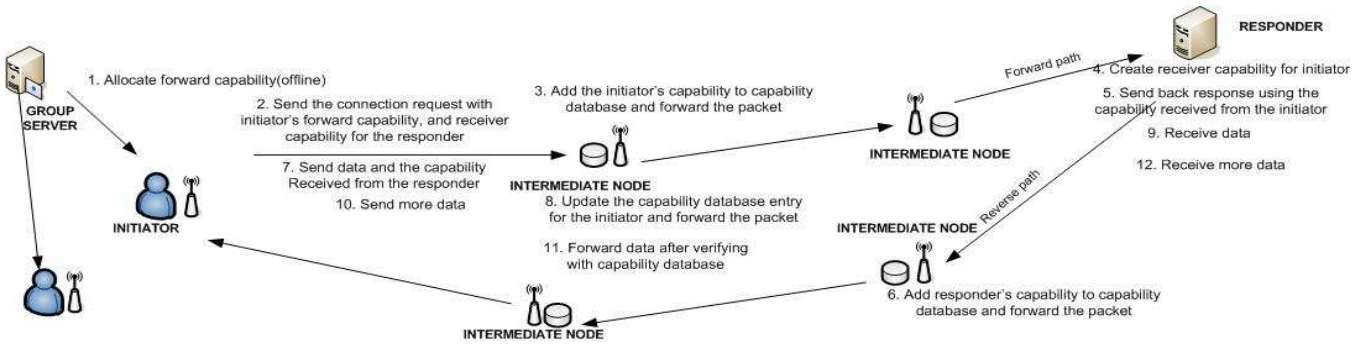

Fig. 1. System overview

on receiving this packet from the responder, validates the packet and adds the capability to its capability database. In the diagram, the reverse path is shown to be different from the forward path. The paths can also be same in-practice. Initiator can optionally send the receiver capability it received from the responder in the forward path. In that case, the intermediate nodes will update its capability database with the new capability.

Any further data traffic between the initiator and the responder does not contain the capability, but it contains only the transaction id that was included in the initial handshake as well as a signature that can be verified by the intermediate nodes. The intermediate nodes can validate the packets by looking at the capability contained in the capability database corresponding to the transaction id in the packet. This validation involves making sure that the packet does not exceed the resource limit allowed in the capability as well as probabilistically verifying the packet signature. For this validation, the intermediate node also maintains the resource usage against each capability in its capability database. The only time the initiator or responder need to re-send the capability is when the path between them changes due to mobility of the nodes, or for refreshing the capability periodically.

We note that our solution can be used to protect multicast traffic and routing control packets. We omit the details due to lack of space.

A. Memory and CPU available per packets in MANETS

We argue that the proposed solution is feasible for MANETs, even though the memory and processing power are lower in MANET nodes compared to routers in wired networks. And our scheme requires memory to store the information about the traffic sessions and processing power to do public key operations. The feasibility comes from the fact that the bandwidth of MANET is significantly less compared to that of wired network. The available memory or processing power per packet is higher in MANET compared to that of wired network. It is possible to do public key operations on one in every few packets on MANETs today. The processing power per packet for MANET nodes are increasing everyday with the advent of faster but less power hungry processors for portable devices.

The public key operations for verifying that a packet belongs to a capability can be achieved with very small key sizes. This is because, unlike traditional use of public keys, the keys are useful only for the short duration of the session. For longer sessions, new keys can be generated and old ones discarded.

B. Capability definition

Each node has authority to send traffic to its peers at certain rates. These are represented by KeyNote-style credentials [4]. These capabilities are assigned by a central authority. The credential contains

1. Identity of the node (principal)
2. (Optional) identity of the destination node.
3. Type of service and amount of data the principal is allowed to send.
4. Signature of the group server

All the nodes in the MANET know the public key of the group servers, so that they can verify the capabilities. Destination node can be a host or a subnet. Type of service is the protocols this capability can access. The amount of data that can be sent can be represented as various constraints on the packet rates that can be verified by a node.

Typically the bandwidth available to a node on a receiver capability is more than that of its forward capability. This is because receiver capabilities are negotiated by the communication parties and the communication parties are free to assign as much as it like within the policy, provided they can process them. Where as forward capabilities are assigned by the central authority and it does not know about the load on the node when communication takes place. Hence the central authority will consider the worst case scenario while assigning the forward capability. Also if the forward capabilities were larger than the receiver capabilities received by the sender, then the sender has no motivation for using the receiver capability.

Forward and receiver capabilities have the same syntactic representation. Following is an example:

```
serial: 130745
owner: unit01.nj.army.mil
destination: *.nj.army.mil
service: https
bandwidth: 50kbps
expiration: 2010-12-31 23:59:59
issuer: captain.nj.army.mil
signature: sig-rsa 23455656767543566678
```

The above represents a forward capability assigned by the captain.nj.army.mil to the unit number 1. The unit can use this capability to send the traffic to any node in the domain *nj.army.mil*. The instantaneous data rate using this capability cannot exceed 50kbps.

If unit 1 wants to communicate with unit 2, it will send a message to unit 2 using this forward capability. The unit 2 will issue a receiver capability for the unit 1, if the communication needs more bandwidth than available in the forward capability.

```
serial: 1567
owner: unit01.nj.army.mil
destination: unit02.nj.army.mil
bandwidth: 150kbps
expiration: 2007:10:21 13:05:35
issuer: unit02.nj.army.mil
comment: Policy allowing the receiver
    to issue this capability.
signature: sig-rsa 238769789789898
```

This capability is restricted to be used only by unit 1 for communication with unit 2. This has a higher bandwidth, but a shorter expiration date. The issuer of the capability is the same as the destination of the capability.

After receiving this capability, unit 1 will use this capability for communication with unit 2. Since the new capability has higher bandwidth, unit 1 will be able to do the intended communication. The more general forward capability can now be used by unit 1 for communicating with other nodes.

If the communication from unit 1 to unit 2 was short and low bandwidth, unit 1 could have used its forward capability for the entire duration of the communication, with out requesting for a receiver capability from the unit 2. This will be faster for short communication as there is no capability request/reply and unit 2 does not have to issue any capabilities. If unit 1 expects some messages from unit 2 that requires more capabilities than the one that is available with unit 2 in form of forward capability, then unit 1 could issue a receiver capability for unit 2.

C. Security Analysis

We now discuss how our architecture relates to the threat model described in Section II.

Since the capabilities are signed by a group server and are verifiable by the nodes, adversaries cannot generate their own valid capabilities. Adversaries can create valid capabilities only if the group server is compromised. Since the individual packets are signed, an adversary cannot use a transaction ID that does not belong to it to send the packets.

A compromised node that does not enforce the capability protocol can only have impact within its communication radius. A compromised node can only access the services it is authorized to. Packets of nodes trying to use more bandwidth than allocated will be rejected. A malicious node frequently doing this can be detected and isolated.

A receiver can protect against DoS attacks by controlling the issuance of receiver capabilities to its communicating peers. A malicious node can use its forward/receiver capabilities to send duplicate packets in multiple disjoint paths; we do not currently protect against this attack, which allows a node to transmit more traffic that it is authorized to. We note, however, that local nodes in the radio perimeter of the misbehaving node can detect this scenario.

IV. Protocol details

Depending on the stage of communication, a MANET source node can transmit either data packets or control packets to a destination node. These packets are routed by the underlying MANET routing protocol, which could be AODV, DSR, or any other mobile routing protocol. However, for those packets to be routed, the source node should have in its possession a capability token in the form of forward or receiver capability. Thus, an intermediate node forwards a packet or the receiving node accepts a packet only if there is a capability associated with the packet. To prevent excessive communication overhead by sending capability inside each packet, we establish a state at the intermediate nodes using the concept of *transaction identifier*.

A node sending the packet creates a unique identifier and assigns one of its capabilities for each communication session it participates in. The node informs about the association between the transaction identifier and the associated capability to the intermediate nodes. This mapping is kept at the intermediate nodes in its *capability database*. All the subsequent packets contain only the transaction identifier. Intermediate nodes verify the packet against the capability using this id and drop the packet if there is a violation.

The capability database, which a table stored in memory at the intermediate nodes, has the following entries.
Source node: The address of owner of the capability.
Transaction id: The unique id generated by the source node for this communication.
Destination node: The address of the destination node.
Capability: A copy of the capability.
Statistics: A structure maintained to enforce the bandwidth limitation of the capability.

Each source node also maintains a table called *transaction table* that maps the traffic session with transaction identifier, capability and usage. This table is consulted before the packets are sent for a traffic session.

A. Control packets

Control packets are used for exchange of capability information, as well as error reporting. Data packets can also be piggybacked on control packets. A control packet can have multiple messages embedded in it using type, length, value (TLV) fields. The following control message types are supported:

CAP-REQ message is used for establishing an entry in the capability database of the nodes along the path from a source to a destination. The message contains the source node address (ID_i), transaction id (TX_{ir}), destination node address (ID_r), flags and the capability (C). This message is signed with the public key of the sender. The message also contains a smaller

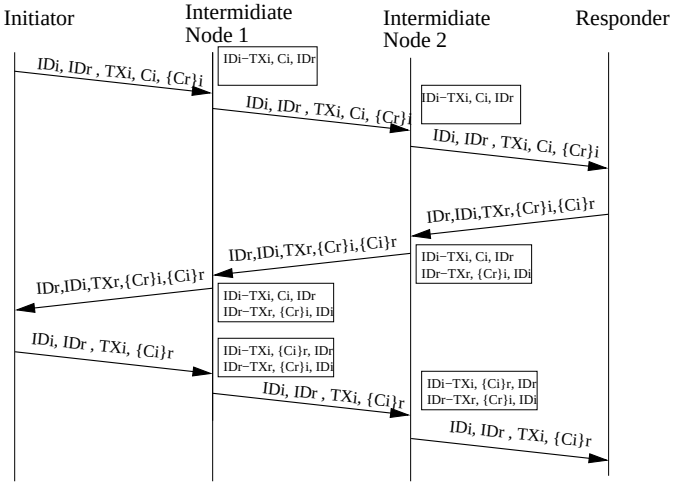

Fig. 2. Connection Establishment

public key that will be used to sign the subsequent packets. Smaller keys are used to reduce the length of the signatures in the packets as well as the processing time.

RECV-CAP message is used by a receiver (responder) for sending a receiver capability back to the sender (initiator). CAP-ERROR message is used for sending various error messages. The value in the message contains the error type. PUB-KEY messages are used to advertise the public key of the sender. DATA message is used for piggybacking the data packets in the control messages. CAP-INFO messages are used for retrieving the capability information after a route change.

B. Connection establishment

When an initiator node wants to enter a communication session with a responder node, the former creates a transaction identifier, associates a capability with it and adds it to the transaction table. Then it sends the following control packet to the responder: CAP-REQ[ID_i, ID_r, TX_i, C_i], RECV-CAP[$\{C_r\}_i$]. Here ID_i is the address of the initiator, ID_r is the address of the responder, TX_i is the transaction identifier, C_i is the forward capability of the initiator, $\{C_r\}_i$ is the receiver capability assigned by the initiator to the responder

This packet consists of two message types. One is a CAP-REQ message for establishing the capability entry at the intermediate nodes. The optional second message type is the receiver capability (RECV-CAP) for the responder to communicate back with the initiator. If the responder already have enough capability to communicate back with the initiator, then RECV-CAP message need not be included. The control packet may also contain a PUB-KEY message if the initiator public key is not known to the nodes. The packet may also piggyback the data using DATA message type.

The packet is send to the next neighbor (intermediate node) using the underlying routing protocol. Each intermediate node verifies the capability C_i and adds it to its capability database. This entry in the capability database indicates that the packets received from the initiator with transaction id TX_i is to be validated with capability C_i and they are destined for ID_r.

The responder node, on receiving the connection request, creates a transaction identifier for the communication and associates it with either the received capability or another capability it already possess. It also creates a receiver capability $\{C_i\}_r$ for the initiator if the initiator has requested one. Consider the case where the initiator had sent a receiver capability for the responder and had requested for a receiver capability from the responder. The responder sends the following control packet to the initiator: CAP-REQ[ID_r, ID_i, TX_r, $\{C_r\}_i$], RECV-CAP[$\{C_i\}_r$]. Each intermediate node (not necessarily the same as in the forward path) creates an entry in its capability database from CAP-REQ message.

On receiving the response packet, the initiator forwards the packet CAP-REQ[ID_i, ID_r, TX_i, $\{C_i\}_r$] to responder, so that the capabilities are updated with the new capability $\{C_i\}_r$ at the intermediate node. The intermediate nodes update the capability database with the new capability.

The connection initiation sequence is shown in Figure 2. For ease of representation, message types are omitted from the figure. The boxes in the figure show the contents of capability database.

C. Data transfer

The data packets are of following format.
$ID_i, ID_r, TX_i, < data >, < signature >$

An intermediate node verifies the packet against the associated capability before forwarding it. It also probabilistically verifies the packet signature to prevent spoofing attacks.

D. Routing Changes

The route between two nodes can change, when the intermediate nodes move in MANET. The new route may contain new intermediate nodes that were not part of the initial connection setup. After a route change, when an intermediate node receives a data packet for which there is no capability database entry, then the node requests for that entry from the upstream node. The entry need to be verifiable by the intermediate node to avoid spoofing attacks. This is illustrated in Figure 3.

New intermediate node 3 is added to the path from the initiator to the responder. When the data packet $ID_i, ID_r, TX_i, < data >$ is received by node 3, it checks ID_i, TX_i in its capability database. Since that entry is not present in its database, node 3 drops the packets and sends a CAP-INFO message requesting information about the capability associated with that transaction id to the upstream node. If the upstream node has the information in its capability database, it forwards the corresponding entry to node 3 using a CAP-INFO response message. Node 3 verifies the capability and installs the capability in its capability database. Hence the effect of routing change is localized only to the neighborhood of the change and does not affect the whole route. Future communication using that transaction id can go uninterrupted.

If the upstream node does not have information about the transaction id in its capability database when it receives the CAP-INFO message, then it sends a CAP-NOT-FOUND

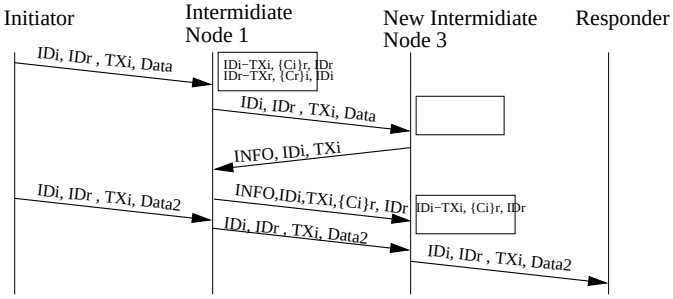

Fig. 3. Route change

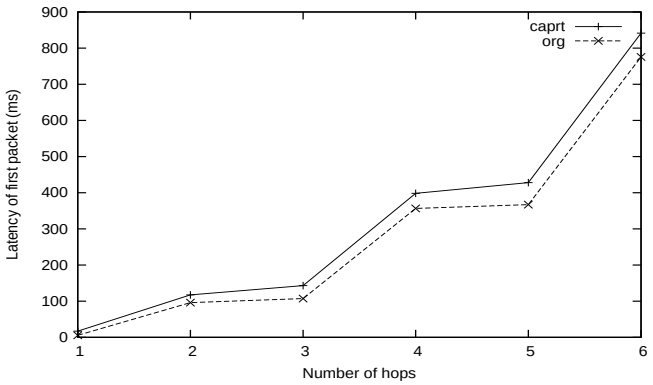

Fig. 4. Latency of the first CBR packet of size 512 bytes

message to the source node. This can happen if multiple consecutive nodes are involved in the route change. The source node, on receiving that message, sends a new CAP-REQ message to the destination node. All the intermediate nodes in the route update their capability database on receiving this message. The data transfer continues normally from this point.

Proactive approaches: Another approach for updating node3's capability database when a routing change happens is for node 1 to proactively send the entry to node 3. This can be done either when the node 1 detects the routing change, or when the node 1 finds the first packet for the transaction id after the routing change. Another approach could be that the node 3 allows the data packets to pass through for a short duration after a routing change, and gets the capability information from the source node or an upstream node during that time.

E. Capability refresh

Periodically, or when the capability is about to expire, the end nodes (both initiator and responder) create new receiver capabilities and send them to the other end. The nodes forward the new capabilities along the forward path, so that the intermediate nodes update their capability database with the new capabilities. Capability refresh packets are also used for updating the keys used for signing the data packets if the communication sessions are of long duration, since we use short keys for data packets.

V. Simulation Results

We implemented our approach in the GloMoSim simulator [1]. To that end, we extended GloMoSim by developing an additional layer between the IP and AODV routing layer [24]. In this section, we compare the performance of capability-based MANETs (referred as *caprt*) with a system that does not use capabilities (referred as *original*). First, we use a simple line topology, where seven nodes 0 through 6 are arranged in a line 200 meters apart. We use this simple topology for computing the basic overhead of our scheme, since it is easy to analyze the results. Then, measure our system using more complex and realistic networks.

In our experiments, we use the default radio parameters of GloMoSim: radio range 376.782m and link bandwidth 2 Mbps. We use 802.11 as the MAC protocol. We introduce a packet processing delay in both models. This is set to 0.01 milliseconds. This is the time required to process 128-byte packets on a 100 Mbps link. To protect the integrity of the capability tokens and verify the identity of the sender and the receiver, we employ 256-bit RSA for signing the individual data packets and 1024-bit RSA for signing the capability itself. This incurs an additional 36 bytes per data packet; 4 bytes for the transaction identifier and 32 bytes for the signature. Requests to issue a capability packets are always verified by the intermediate nodes (relatively low traffic). However, to improve the latency performance of our system, we chose to verify data packets probabilistically (this can depend on the path length). Upon detection of an unauthorized packet, we can revert back to deterministic packet checking. The cost of all packet operations are (per packet): inserting a capability token (identifier) in the capability database costs an average of 0.01 milliseconds and the record lookup operation 0.005 milliseconds. In addition, generating a signature requires 0.168 milliseconds and verification 0.0275 milliseconds for data packets. Capability protocol packets requires 3.159ms for signature generation and 0.140ms for signature verification. A capability refresh packet is sent every 8 seconds. Simulations were run on a Pentium-4 3.20GHz CPU with 1GB memory.

Each intermediate node verifies the signature of a packet with probability 0.2063. Since this verification decision is taken independently by each node, a signature of a packet is verified by at least one node in a 3 hop path with probability 0.50 (*i.e.,* 1-(1-0.2063)(1-0.2063)(1-0.2063)).

We implemented a token bucket algorithm to enforce the bandwidth limitation at the intermediate nodes. Each of the experiments below was run 20 times with different seed values, and the average of the parameter of interest was taken.

A. Packet latency

First we compare the latency of the packets in our scheme (denoted as *caprt*) with that of the one does not use the capability based protection (denoted as *original* or *org*). We send 1000 constant bit rate (CBR) packets of size 512 bytes at 100 ms interval from a source node to a destination node n hops away, where $n = 1, \ldots 6$ in the line topology. We measure the latency of the packet as the time from the creation of packet in the source node to time it reaches the destination.

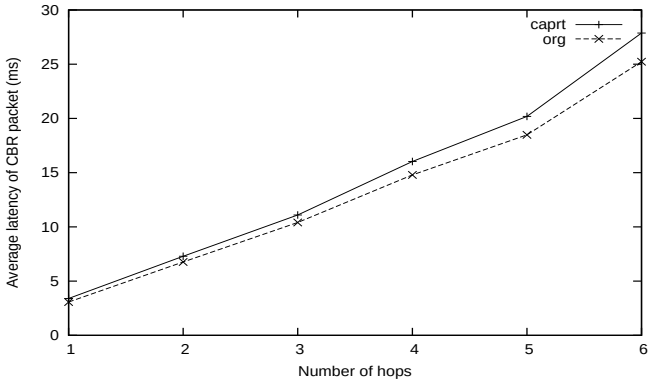

Fig. 5. Average latency of 1000 CBR packets of size 512 bytes

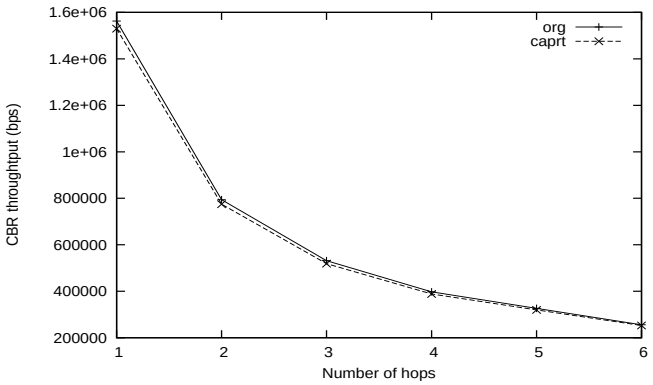

Fig. 6. CBR throughput

The latency of the first packet is larger than the rest of the packets. This is because route needs to be discovered in both the schemes and capability needs to be established in the capability scheme. The packet processing for the capability scheme also includes capability database lookup and probabilistic verification of packet signatures.

Figure 4 shows the latency for the first packet to reach the different destination nodes. The higher latency in capability scheme is due to the capability establishment, capability database lookup and signature verification as well as additional overhead (36 bytes) in the packet. This average overhead is 35.8ms, 41.6ms and 60.9ms respectively for nodes 3, 4 and 5 hops away. The average overhead is 20.5%. The overhead increases as hop length increase since the overhead is added at each node. It can also be seen that the latency increases considerably from 3 hops to 4 hops in both the schemes. This is coming from AODV because AODV had to increment the TTL once more and retransmit the RREQ packet while finding the routes to the node that was 4 or 5 hops away. The same is true for 6 nodes.

Figure 5 shows the average latency for all the 1000 packets to reach the different destination nodes. Since there are large number of packets, effect of the high latency of first packet is minimized. Here the average overhead is only 0.6ms, 1.2ms and 1.6ms respectively for nodes 3, 4 and 5 hops away. The average overhead is only 8% for these paths.

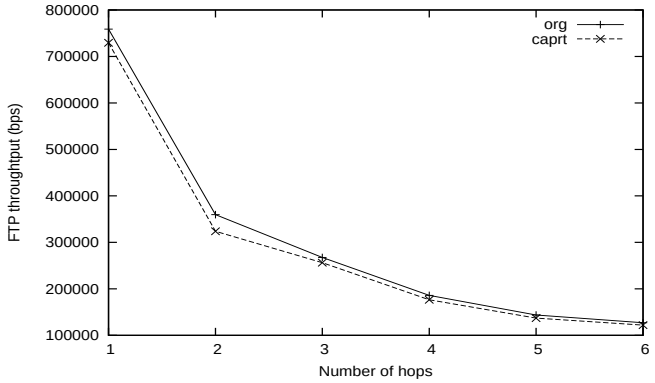

Fig. 7. FTP throughput

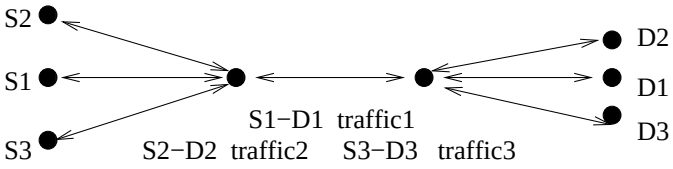

Fig. 8. Topology to study the mis-behaving nodes

B. Throughput

We compare the throughput of the capability scheme with original scheme on an 802.11 network. We use the line topology and pump large CBR packets (1400 bytes) at high rate (every 1ms). We set the node 0 as the source of the CBR traffic and send the traffic to destination nodes at different hops. We measure the number of bytes received within one minute from the start of the data transfer and compute the data throughput. The results are shown in 6.

As expected, throughput of both the schemes decreases as the number of hops in the path increases. The throughput of capability scheme is only 2% lower than the original scheme.

C. TCP performance

To measure the performance of TCP on capability scheme, we compare the throughput of FTP on both the schemes on a line graph. An FTP client at node 0 transfers data to an FTP server at $1, 2, \ldots, 6$ hops away. In each experiment the client sends 10 application layer items of random sizes. The application layer item sent was same for both the schemes in the same experiment. The results are plotted in Figure 7. The behavior of TCP performance is similar to that of CBR, but at lower bandwidth due to TCP congestion control and in-order guaranteed delivery. On average, TCP throughput for the capability scheme is 5.3% lower than the original scheme.

D. Resilience against mis-behaving nodes

In this experiment, we study the attack resilience of the capability based protocol. The topology for this experiment is shown in Figure 8. There are three traffic sessions competing for the bandwidth resources of the network. The traffic is CBR packets of size 512 bytes send at packet intervals 40ms, 20ms and 10ms respectively by $S1$, $S2$ and $S3$. Each of the nodes is given the same capabilities of varying bandwidth.

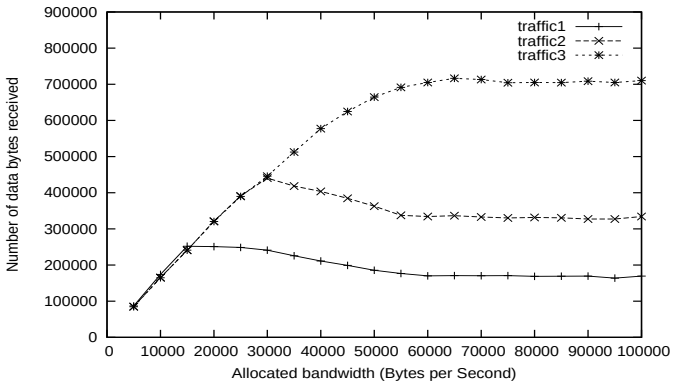

Fig. 9. Limiting bandwidth of misbehaving nodes

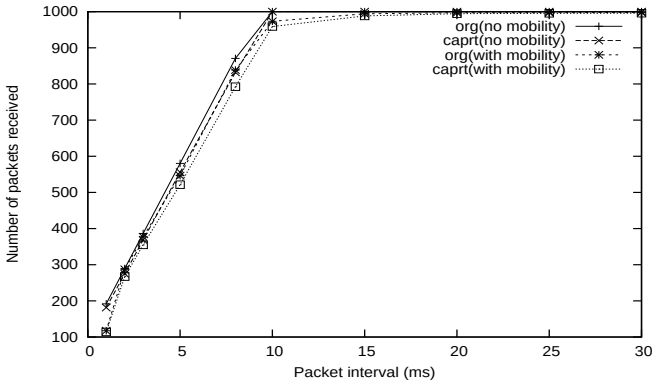

Fig. 10. Number of packets received by the destination after the route change

The results are shown in Figure 9. Even though traffic3 is four times the traffic1 and two times the traffic2, each of them gets the same bandwidth initially. Any increase in the allocated bandwidth after reaching the bandwidth of traffic1, give the same increase for both traffic2 and traffic3. Then traffic2 gets saturated. After that traffic3 bandwidth increases till that also gets saturated. We see that the number of bytes received slightly less than the allocated bandwidth since the UDP/IP and capability headers are not counted as part of the bytes received.

E. Mobility

We study the effect of mobility on capability scheme. Since the nodes keep soft state about the capabilities, when the route changes due to node mobility, the new node needs to receive the capability information about existing sessions.

Figure 10 shows the effect of mobility on number of packets received for various inter packet intervals. In this experiment, 1000 CBR packets of 512 bytes were sent to a node 3 hops away starting at time 0. At time 0.5 seconds, node at 2 hops away was removed and a new node introduced. Figure 10 shows the number of packets received at the destination for both the schemes, with and without this mobility. The number of packets received reduces as the packet interval reduces due to limitation in throughput. As expected, the number of packets lost due to mobility increases at lower inter packet interval. On average, capability scheme loses 155ms worth of traffic; where as original scheme loses 108ms worth of traffic. The higher loss is due to the need for getting the capability related information by the new node in the path from the source.

		3-93	6-96	30-39	60-69
FTP	Original	16395	15546	14759	14694
	Capability	14062	17722	14176	15147
CBR	Original	65711	162293	57535	177303
	Capability	54113	148027	57793	148153
	Ltd bw capability	129164	131437	129844	134230
CBR Mobility	Original	74124	150718	52510	157779
	Capability	59864	117728	57933	129347
	Ltd bw capability	113111	136975	100924	138040

TABLE I

FTP AND CBR THROUGHPUT (BPS) ON A GRID TOPOLOGY

F. Grid Topology

We next use a grid topology containing 100 nodes (10x10 grid), each nodes 300m apart. We ran four ftp sessions, two of them from the nodes on the top of the grid to bottom of the grid; between nodes pairs $(3, 93)$ and $(6, 96)$. The other two ftp session were from left to right between node pairs $(30, 39)$ and $(60, 69)$. We also ran CBR traffic of 1400 bytes with 10ms packet interval for those source destination pairs and computed throughput.

Table I shows the average throughput of the four sessions, for both FTP and CBR. As it can be seen, the average throughput of the schemes are comparable. Capability scheme throughput is only 0.5% lower for FTP and 11.8% lower for CBR compared to the original scheme.

CBR experiment in the table contains 3 rows. The original scheme does not limit the bandwidth a node can use. The capability scheme in the second row allowed capability to use unlimited bandwidth. In both cases two of the sessions get most of the bandwidth. In the third experiment, the capabilities had limited bandwidth. In this case, each of the sessions received a fair share of the available bandwidth.

The last set of rows shows the effect of mobility in both the schemes. In this set of experiments, the second node in the route from source to destination of all the traffic pairs were removed after 2 seconds. The average throughput of the capability scheme was 16.1% less than the original scheme. This reduction is more than the CBR traffic without mobility. This is because the capability scheme needs more time to restore, due to need for restoring the capability database in the new route. The last row shows the results when the capability had a limited bandwidth. Here the average bandwidth dropped by 6.8% compared to capability scheme without mobility.

VI. RELATED WORK

Security for mobile *ad hoc* network is an active area of research. In this section we give a sample of work in this area. Surveys of research in MANETs can be found in [31], [32], [26].

The concept of capabilities was used in operating system for securing resources [30]. There was work on allowing controlled exposure of resources at the network layer using the concept of ”visas” for packets [10], which is similar to

network capabilities. More recently, network capabilities were proposed to prevent DoS in wired networks in [3].

Key management is an important building block of any security system. There are various schemes proposed in literature for establishing symmetric keys between a pair of nodes, in the presence of a central authority by random pre-allocation of key shares [9], [5], [17], [7], [36]. They are particularly useful for sensor networks, which has limited memory and computing [25]. Many papers have addressed public key distribution under the assumption of unavailability or compromised certificate authority [35], [15]. Research also attempted to secure data and global time in the presence of malicious nodes [6], [37], [27]. Securing the routing protocols in MANET is an active area of research [19]. Solutions are proposed to protect the routing protocol like DSR, with the help of cryptographically signing the messages [13]. The solutions are also proposed for preventing specific routing attacks like worm hole, rushing attack etc [12], [33], [11]. Secure network and transport protocol are also proposed in mobile and multi-homing environment [21].

Intrusion detection systems (IDS) for MANETs is also an active area of research [20]. Various distributed IDS architectures are proposed for resource constrained MANETs [2], [34]. Solutions are also proposed to specifically detect the attacks on routing protocol based on the protocol specification [14], [28], [8] and mechanisms to respond to those attacks [29]. There is rich literature on detecting denial of service and node replication attacks on sensor networks [18], [22]. Reputation mechanism is also proposed in MANETs to guard against selfish nodes [16].

VII. Conclusions and Future Work

We presented a novel architecture for enforcing security policies in MANETs. This scheme is based on the concept of network capabilities, and can protect both end-host resources and network bandwidth from denial of service attacks, as well as limit the exposure of the MANET to a compromised node. We showed the details of the capability propagation protocol and discussed the various scenarios of use. Our simulation results show that the impact of the scheme is minimal on throughput and latency, in spite of using cryptographic operations. For our future work, we plan to study the impact of our scheme on larger topologies and classes of traffic. In addition, we would like to quantify the resilience of the scheme in the presence of misbehaving nodes and the performance of multicast traffic on mobility scenarios, and to implement and deploy on MANET testbeds with real traffic.

References

[1] GloMoSim simulator. *http://pcl.cs.ucla.edu/projects/glomosim/*.

[2] P. Albers, O. Camp, B. Jouga, L. Me, and R. Puttini. Security in ad hoc networks: A general id architecture enhancing trust based approaches. *IEEE Network, WIS*, 2002.

[3] T. Anderson, T. Roscoe, and D. Wetherall. Preventing internet denial-of-service with capabilities. *Proc. of Hotnets-II*, 2003.

[4] M. Blaze, J. Ioannidis, and A. Keromytis. Trust management for ipsec. *Symposium on Network and Distributed Systems Security (SNDSS)*, 2001.

[5] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. *IEEE Security and Privacy*, 2003.

[6] H. Chan, A. Perrig, and D. Song. Secure hierarchical in-network aggregation in sensor networks. *ACM CCS*, 2006.

[7] W. Du, J. Deng, Y. Han, and P. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. *ACM CCS*, 2003.

[8] P. Ebinger and T. Bucher. Modelling and analysis of attacks on the manet routing in aodv. *LNCS*, 2006.

[9] L. Eschenauer and V. Gligor. A key-management scheme for distributed sensor networks. *ACM CCS*, 2002.

[10] D. Estrin, J. C. Mogul, and G. Tsudik. Visa protocls for controlling interorganizational datagram flow. *IEEE Journal on Selected Areas in Communications*, May 1989.

[11] L. Hu and D. Evans. Using directional antennas to prevent wormhole attacks. *NDSS*, 2003.

[12] Y. Hu, A. Perig, and D. Johnson. Rushing attacks and defense in wireless ad hoc network routing protocols. *WiSe*, 2003.

[13] Y. Hu, A. Perrig, and D. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. *MOBICOM*, 2002.

[14] Y. Huang and W. Lee. Attack analysis and detection for ad hoc routing protocols. *RAID*, 2004.

[15] J. Hubaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad hoc networks. *MobiHOC*, 2001.

[16] J. Jaramillo and R. Srikant. Darwin: Distributed and adaptive reputation mechanism for wireless ad-hoc networks. *MOBICOM*, 2007.

[17] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. *ACM CCS*, 2003.

[18] J. McCune, E. Shi, A. Perrig, and M. Reiter. Detection of denial-of-message attacks on sensor network broadcasts. *IEEE Security and Privacy*, 2005.

[19] N. Milanovic, M. Malek, A. Davidson, and V. Milutinovic. Routing and security in mobile ad hoc networks. *IEEE Computer Society*, 2004.

[20] A. Mishra, K. Nadkarni, and A. Pactha. Intrusion detection on wireless ad hoc networks. *IEEE Wireless Communications*, 2004.

[21] P. Nikander, J. Ylitalo, and J. Wall. Integrating security and mobility and multi-homing in a hip way. *NDSS*, 2003.

[22] B. Parno, A. Perrig, and V. Gligor. Distributed detection of node replication attacks in sensor networks. *IEEE Security and Privacy*, 2005.

[23] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu. Portcullis: protecting connection setup from denial-of-capability attacks. *SIGCOMM Comput. Commun. Rev.*, 37(4):289–300, 2007.

[24] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on demand distance vector (AODV) routing. *IETF RFC 3561*.

[25] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. Tygar. Spins: Security suite for sensor networks. *MOBICOM*, 2001.

[26] E. Shi and A. Perrig. Designing secure sensor networks. *IEEE Wireless Communications*, 2004.

[27] K. Sun, P. Ning, C. Wang, A. Liu, and Y. Zhou. Tinysersync: Secure and resilient time synchronization in wireless sensor networks. *ACM CCS*, 2006.

[28] C. Tseng, S. Wang, C. Ko, and K. Levitt. Demem: Distributed evidence-driven message exchange intrusion detection model for manet. *RAID*, 2006.

[29] S. Wang, C. H. Tseng, K. Levitt, and M. Bishop. Cost-sensitive intrusion responses for mobile ad hoc networks. *RAID*, 2007).

[30] E. Wobber, M. Abadi, M. Burrows, and B. Lampson. Authentication in the taos operating system. *ACM Transactions on Computer Systems*, 12, February 1994.

[31] B. Wu, J. Chen, J. Wu, and M. Cardei. A survey on attacks and countermeasures in manets. In *Wireless/Mobile Network Security*, chapter 12. Springer, 2006.

[32] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang. Security in mobile ad hoc networks: Challenges and solutions. *IEEE Wireless Communications*, 2004.

[33] P. Yi, Z. Dai, S. Zhang, and Y. Zhong. A new routing attack in mobile ad hoc networks. *International Journal of Information Technology*, 2005.

[34] Y. Zhang and W. Lee. Intrusion detection for wireless ad-hoc networks. *MOBICOM*, 2000.

[35] L. Zhou and Z. Haas. Securing ad hoc networks. *IEEE Network*, 1999.

[36] S. Zhu, S. Setia, and S. Jajodia. Leap: efficient security mechanisms for large-scale distributed sensor networks. *ACM CCS*, 2003.

[37] S. Zhu, S. Setia, S. Jajodia, and P. Ning. An interleaved hop-by-hop authentication scheme for filtering false data injection in sensor networks. *IEEE Security and Privacy*, 2004.