Evaluating a Collaborative Defense Architecture 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 susceptible to both insider and outsider attacks more than wired and base station-based wireless networks. This is because of the lack of a well-defined defense perimeter in MANETs, preventing the use of defenses including firewalls or intrusion detection systems. This lack of perimeter calls for implementation of security in a distributed, collaborative manner.**

We recently introduced a novel deny-by-default distributed security policy enforcement architecture for MANETs by harnessing and extending the concept of *network capabilities*. The *deny-by-default* principle allows compromised nodes to access only authorized services, limiting their ability to disrupt or even interfere with end-to-end connectivity and nodes beyond their local communication radius. The enforcement of policies is done hop-by-hop, in a distributed manner. In this paper we present preliminary results evaluating our architecture. Through simulation, we show that our solution incurs minimal overhead in terms of network bandwidth and latency even in the presence of cryptographic operations. Furthermore, we show that the protection remains effective even in the presence of misbehaving nodes and routing changes due to mobility. While further work is needed to fully evaluate our scheme, we believe that the notion of collaborative security in MANETs is a promising direction for future research.

Keywords: MANETs, Capabilities, Distributed firewalls

I. Introduction

Mobile Ad-hoc Networks (MANETs) are increasingly employed in both military and commercial network situations where fixed infrastructure is too costly or dangerous to deploy, or has been rendered inoperable. MANETs are fundamentally different from the Internet because all peers act as both sources and routers using the other participants to relay packets to their final destination. MANETs are susceptible to both insider and outsider attacks. Even a small number of misbehaving nodes can successfully render the entire MANET inoperable: malicious peers can abuse the network exhausting all network and power resources.

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. 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 address this, recently, we proposed a deny-by-default architecture [3] that enforces trust relationships and traffic accountability between mobile nodes through a distributed policy enforcement scheme for MANETs. In that architecture, we extended the network capability framework [4] and tailored it to the resource-constrained MANET environment. A capability is a token of authority that has associated rights. The capabilities propagate both access control rules and traffic-shaping parameters that should govern a node's traffic. In the deny-by-default, model nodes can only access the services and hosts they are authorized for by the capabilities given to them. The enforcement of the capability is done in a distributed manner by all the nodes in the path from the source to the destination. Compromised or malicious nodes cannot exceed their authority and expose the whole network to an adversary. Upon detection, we can prevent a compromised node from further attacking the network simply by revoking its capabilities. Moreover, that architecture helps 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.

In this paper, we provide a preliminary evaluation of such a deny-by-default system using the GloMoSim [1] simulator. Because GloMoSim does not include any packet checking functionality, we added another layer between IP and the AODV routing processing, where we implemented our protocols. Our primary concern in the evaluation is 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 show that the collaborative effort of enforcing the policies provides strong security benefits without incurring much performance overhead. We discovered that our scheme imposes an 8% overhead on the end-to-end latency and a 5% reduction on available bandwidth. We believe that this is not a high price to pay given that there are scenarios where a MANET becomes completely unusable even when a single node misbehaves. We also show that our system allocates the network resources in a fair manner, even in the presence of misbehaving nodes.

The rest of the paper is organized as follows. We briefly present the high level system architecture in Section II. We evaluate our architecture through simulation, with results given in Section III. Related work is discussed in Section IV.

978-1-4244-4793-0/09/$25.00

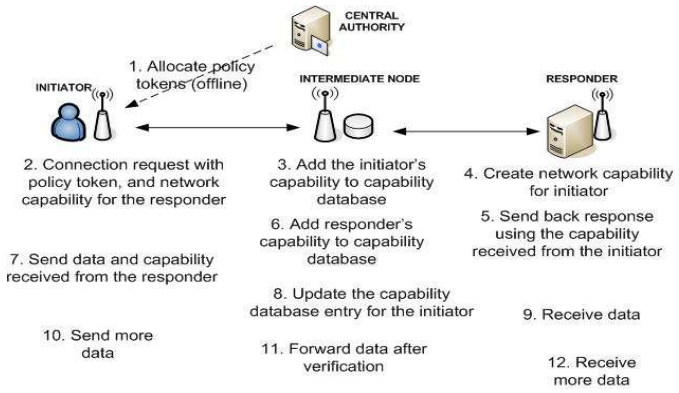

Fig. 1. System overview

II. System Architecture

In our architecture, one or more pre-defined nodes act as a *group controller* (GC), which is trusted by all the group nodes. A GC has authority to assign resources to the nodes in MANET. This resource allocation is represented as a KeyNote-style credential [5] (capability) called *policy token*, and it can be used to express the services and the bandwidth a node is allowed to access. They are cryptographically signed by the GC, which can be verified any node in the MANET.

When a node (initiator) requests a service from another MANET node (responder) using the policy token assigned to the initiator, the responder can provide a capability back to the initiator. This is called a *network capability*, and it is generated based on the resource policy assigned to the responder and its dynamic conditions (*e.g.*, level of utilization).

Figure 1 gives a brief overview of our system. All nodes in the path between an initiator to a responder (*i.e.*, nodes relaying the packets) enforce and abide by the resource allocation encoded by the GC in the policy token and the responder in the network capability. The enforcement involves both accessibility and bandwidth allocation. A responder accepts packets (except for the first one) from an initiator only if the initiator has authorization to send, in the form of a valid network capability. It accepts the first packet only if the initiator's policy token is included. An intermediate node will forward the packets from a node only if the packets have an associated policy token or network capability, and if they do not violate the conditions contained therein. Possession of a network capability does not imply resource reservation; they are the maximum limits a node can use. Available resources are allocated by the intermediate nodes in a fair manner, in proportion to the allocations defined in the policy token and network capability.

The capability need not be contained in all packets. The first packet carries the capability, along with a transaction identifier (TXI) and a public key. Subsequent packets contain only the TXI and a packet signature based on that public key. Intermediate nodes cache policy tokens and network capabilities in a *capability database*, treating them as soft state. A capability database entry contains the source and destination addresses, TXI, the capability, public key for the packet signature and packet statistics. Capability retransmissions update the soft state of intermediate nodes when the route changes due to node mobility. The soft state after a route change is also updated using an on-demand query for the capability database entry from the up stream nodes.

III. Simulation Results

We implemented our scheme in the GloMoSim simulator [1]. To that end, we extended GloMoSim by developing an additional layer between the IP and AODV routing layer [10]. Here, we compare the performance of capability-based MANETs (referred to as *caprt*) with a system that does not use capabilities (referred to as *original*). Note that *original* is inherently vulnerable to a number of attacks, including DoS and unauthorized access, which are not feasible in *caprt*. However, our experiments are aimed at quantifying the performance impact of using our scheme, relative to an unsecure MANET.

We conduct a number of experiments, of increasing complexity in terms of topology and MANET parameters, in order to build up our understanding of the system behavior. Initially, we use a simple "line" topology, where seven nodes (numbered 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. We then measure our system using more complex and realistic networks.

In our experiments, we keep 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. 256-bits are sufficient for very-short-lifetime data packet signatures as we change this key periodically [3]. Headers related to our scheme introduce an additional 36 bytes per data packet; 4 bytes for the transaction identifier and 32 bytes for the signature. Packets containing policy tokens are always verified by the intermediate nodes since they constitute 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 and isolate the misbehaving node. 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 costs 0.005 milliseconds. In addition, generating a signature requires 0.168 milliseconds, while verification takes 0.0275 milliseconds for data packets. Network capabilities require 3.159ms for signature generation and 0.140ms for signature verification. A capability refresh

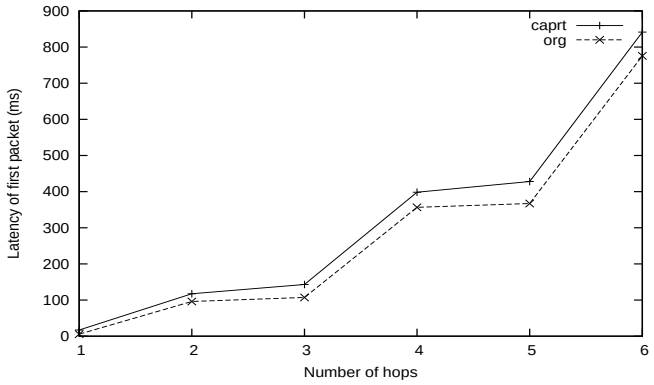

Fig. 2. Latency of the first CBR packet of size 512 bytes

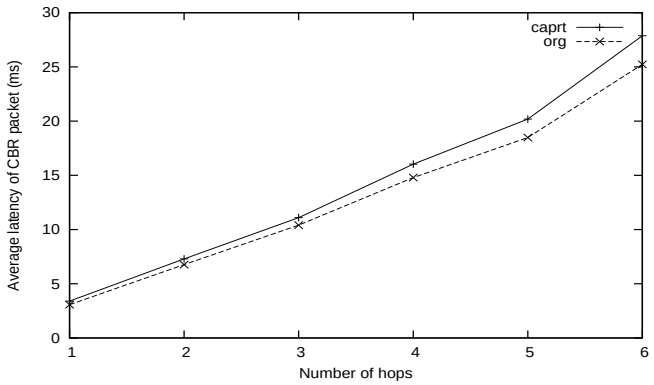

Fig. 3. Average latency of 1000 CBR packets of size 512 bytes

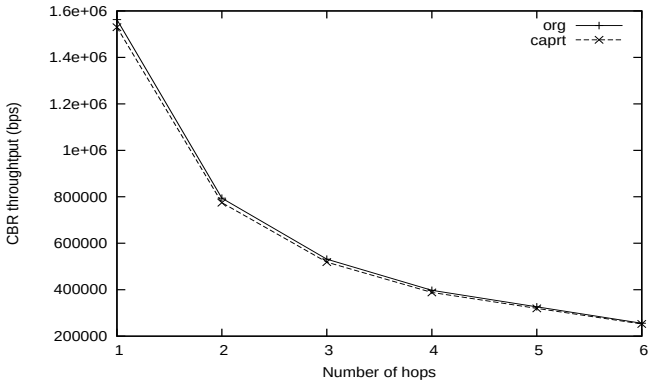

Fig. 4. CBR throughput

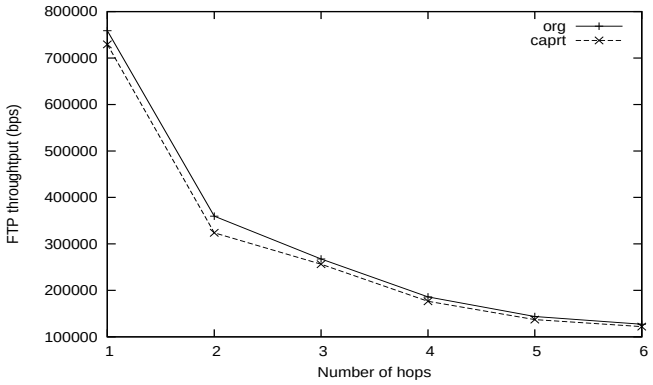

Fig. 5. FTP throughput

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)). The performance overhead for a system in which the nodes verified the signature of all the packets were also similar, since the signature verification did not incur high overhead.

We implemented a token bucket algorithm to enforce the bandwidth limitation at the intermediate nodes. This enables us to limit both burst and average rate of the flow. Each of the experiments was run 20 times with different seed values, and the average of the parameter of interest was taken.

A. Packet latency

We compare the latency of packet processing in *caprt* with that of *original* (also shown as *org* in the figures)). We send 1000 packets of size 512 bytes at 100 ms intervals 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 the time it reaches the final destination.

The latency of the first packet is larger than the rest of the packets. This is because the route needs to be discovered in both schemes (*caprt* and *original*), and credentials need to be established in our scheme. The packet processing for the our scheme also includes capability database lookup and probabilistic verification of packet signatures.

Figure 2 shows the latency for the first packet to reach the different destination nodes. The higher latency in our scheme is due to the credential establishment, capability database lookup and signature verification, as well as the size 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 increases 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 schemes. This is an artifact of using AODV as the underlying routing protocol, 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 3 shows the average latency for all 1000 packets to reach their destination node, in each of the different measurements (transmission to nodes n hops away, varying n in each experiment). The effect of the high latency for the first packet is amortized over a large number of packets. 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.

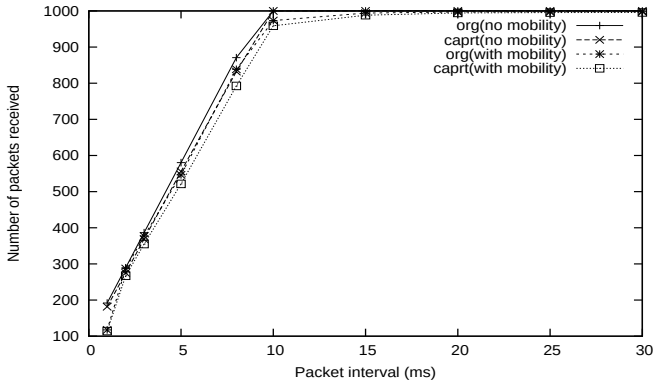

Fig. 6. Number of packets received after a route change

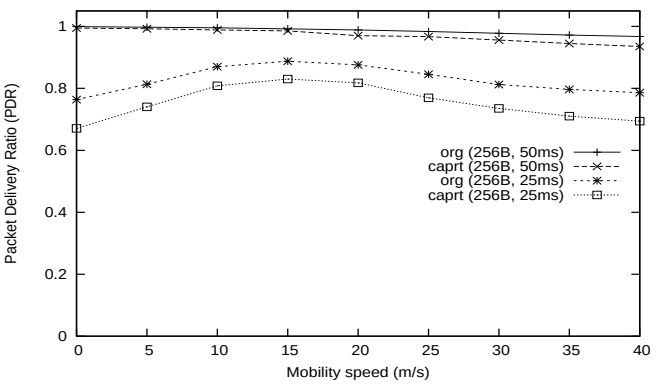

Fig. 7. Packet delivery ratio for unicast CBR traffic at various mobility speeds

B. Throughput

UDP throughput: We now compare the throughput of *caprt* with that of *original* on an 802.11-based MANET. We use the line topology and pump large packets (1400 bytes) at high rate (every 1ms). We set 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 4. As expected, the throughput in both schemes decreases as the number of hops in the path increases. The throughput of our scheme is only 2% lower than the original (insecure) scheme.

TCP throughput: To measure the performance of TCP in our scheme, we compare the throughput of FTP on both schemes on a line topology. 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 the same for both schemes in the same experiment. The results are plotted in Figure 5. 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 our scheme is 5.3% lower than the original (insecure) scheme.

C. Resilience to mobility

To verify the validity of our approach in a MANET environment, we evaluate the effects of mobility on the capability scheme. Since the nodes keep only soft state about the capabilities, when the route changes due to node mobility, the new node needs to receive the credentials (policy token and network capability) for existing sessions.

Figure 6 shows the effect of mobility on the number of packets received for various inter-packet intervals. In this experiment, 1000 packets of 512 bytes were sent at a constant rate to a node 3 hops away, starting at time 0. At time 0.5 seconds, the node 2 hops away was removed and a new node introduced. Figure 6 shows the number of packets received at the destination for both schemes (*caprt* and *original*), with and without this mobility. As expected, the number of packets lost due to mobility increases at lower inter-packet interval. On average, our scheme drops 155ms worth of traffic, whereas the original scheme drops 108ms worth of traffic. This higher loss is due to the need for propagating the network capability to the new node.

		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

D. Larger topology

Next, we evalute our system in the context of a larger and more complex topology, and in the presence of mobility.

Grid Topology: We use a grid topology containing 100 nodes (10x10 grid), each node 300m apart. We ran four FTP sessions, two of them from nodes on the top of the grid to nodes on the bottom of the grid; specifically, between node 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 traffic of 1400 bytes with 10ms inter-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. The average throughput of *caprt* and *original* is comparable. Our scheme's throughput is only 0.5% lower for FTP and 11.8% lower for CBR.

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

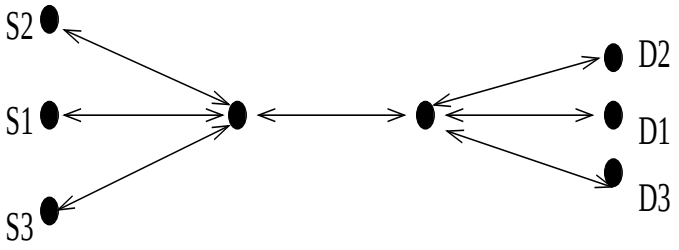

Fig. 8. Topology to study the misbehaving nodes

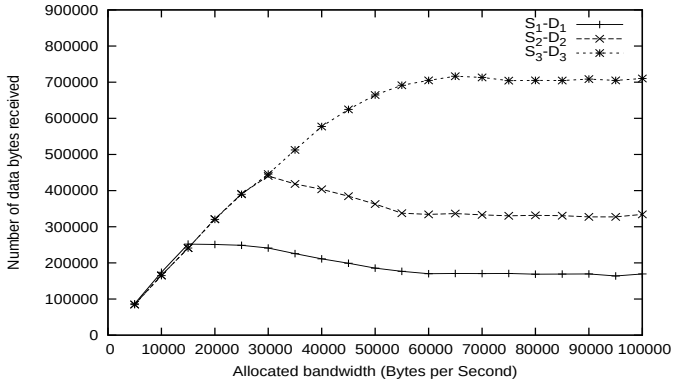

Fig. 9. Limiting bandwidth of misbehaving nodes

The last set of rows shows the effect of mobility in both schemes. In this set of experiments, the second node in the route from source to destination of all the traffic pairs was removed 2 seconds after the experiment began. The average throughput of *caprt* was 16.1% less than the original scheme. This reduction is more than the CBR traffic without mobility. This is because the our scheme needs more time to recover, due to the 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 *caprt* without mobility.

Random Topology: We placed 50 nodes at random on a terrain of $1200s \times 1200$ meters. There were five random source-destination pairs that were sending CBR traffic of 256 bytes at a packet interval of either $50ms$ or $25ms$ (*i.e.,* data rate of $40kbps$ and $80kbps$ respectively). In each experiment, all the nodes were mobile with a constant speed using the *Random Waypoint* model. Each of the experiments was conducted 20 times using different seeds and the average was taken. Experiments where the topology was partitioned were discarded.

Figure 7 shows the packet delivery ratio (PDR) for both schemes at various mobility speeds. On average, the PDR for *caprt* was only 1.6% and 9.14% lower than for *original*, for $50ms$ and $25ms$ inter-packet interval respectively.

E. Resilience against misbehaving nodes

Another important characteristic is the system's behavior in the presence of malicious or misbehaving nodes. To that end, we study the attack resilience of the our protocol. The topology for this experiment is shown in Figure 8. There are three source nodes S_1, S_2 and S_3 sending traffic to respective destination nodes D_1, D_2 and D_3. The traffic is CBR with packets of size 512 bytes, sent at packet intervals 40ms, 20ms and 10ms, originating respectively from S_1, S_2 and S_3. All the nodes are allocated the same bandwidth. Even though all the source nodes have the ability to send to their destination, they try to send more than they are allowed. The node S_3 is the most misbehaving and the node S_1 is the least. The network tries to limit the flow to their capability.

The results are shown in Figure 9. Even though $S_3 - D_3$ traffic is four times the $S_1 - D_1$ traffic and two times the $S_2 - D_2$ traffic, each of them gets the same bandwidth initially. Any increase in the allocated bandwidth after reaching the rate of $S_1 - D_1$ gives the same increase for the other two flows. Once the allocated bandwidth reaches the rate of $S_2 - D_2$ traffic, $S_3 - D_3$ bandwidth increases to its full rate. We see that the number of bytes received is slightly less than the allocated bandwidth since the IP, UDP and *caprt* headers are not counted as part of the bytes received.

IV. Related Work

The concept of capabilities was used in operating system for securing resources [11]. Follow-on work investigated the controlled exposure of resources at the network layer using the concept of "visas" for packets [6], which is similar to network capabilities. More recently, network capabilities were proposed to prevent DoS in wired networks [4]. We extend the concept to MANET and use it for both access control rules and traffic shaping parameters [3]. Previous work on distributed firewalls [7] focused on wired fixed-network environments, attempting to protect only end-hosts, using a host-based solution.

Signing and verification of packets between a sender and a receiver were commercially available in early 1990s. Novell's Netware 3.11 and 4.x supported *NCP Packet Signature Option*, where a unique signature was appended to each packet sent between the client and the server [9]. Mitigating DoS attacks by including a message authentication code and the certificate of the sender for each packet has been previously proposed [12]. That work does not study the high overhead associated with sending a large signature or a large certificate on each packet. The authors use game theory to study the problem of dealing with selfish nodes that do not verify the packet signatures, using incentives and punishments. This mechanism or any other reputation based mechanism [8] can also be used in our scheme to deal with selfish nodes.

HEAP [2] mitigates various MANET attacks from outsider nodes by doing a hop-by-hop packet authentication using HMAC. MACs (end-to-end or hop-by-hop) cannot deal with insider attacks. They also cannot provide access control unless different MAC keys are used for different policies. MACs allow rogue nodes to "hide" since MACs are repudiable as all the intermediate nodes in the path between a sender and a receiver need to know the key. Only public key mechanisms allow packet source validation by all intermediate nodes.

V. Conclusions and Future Work

We evaluated a novel deny-by-default architecture for enforcing security policies in MANETs. The architecture based on the concept of network capabilities, can protect both end-host resources and network bandwidth from denial of service attacks, as well as limit the exposure of the MANET to compromised and malicious nodes. We show that the impact of the scheme is minimal on throughput and latency, in spite of using cryptographic operations, and the scheme can recover well on route changes due to mobility. We also show that the scheme allocates resources in a fair manner even in the presence of misbehaving nodes. For our future work, we plan to implement and deploy on MANET test-beds with real traffic. We plan to measure the performance impact and the user experience with and without the presence of attackers.

Acknowledgements

This material is based on work supported by ONR grant N00014-09-10757 and NSF CNS-07-14277. Any opinions, findings and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of ONR, the National Science Foundation, or the US Government.

Mansoor Alicherry's research is supported by Alcatel-Lucent, Murray Hill, NJ.

References

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

[2] R. Akbania, T. Korkmaz, and G. Raju. HEAP: A packet authentication scheme for mobile ad hoc networks. *Communications and Networking Simulation Symposium*, 2007.

[3] M. Alicherry, A. D. Keromytis, and A. Stavrou. Deny-by-Default Distributed Security Policy Enforcement in Mobile Ad Hoc Networks. *SecureComm*, September 2009.

[4] T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet Denial-of-Service with Capabilities. *Proc. of Hotnets-II*, 2003.

[5] M. Blaze, J. Ioannidis, and A. Keromytis. Trust Management for IPSec. In *Symposium on Network and Distributed Systems Security (SNDSS)*, 2001.

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

[7] S. Ioannidis, A. D. Keromytis, S. M. Bellovin, and J. M. Smith. Implementing a Distributed Firewall. In *Proceedings of Computer and Communications Security (CCS)*, pages 190–199, November 2000.

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

[9] R. Lee. Netware 4.x performance tuning and optimization: Part 3. *http://support.novell.com/techcenter/articles/ana19931001.html*, October 1993.

[10] C. Perkins, E. Belding-Royer, and S. Das. Ad Hoc On Demand Distance Vector (AODV) Routing. *IETF RFC 3561*.

[11] E. Wobber, M. Abadi, M. Burrows, and B. Lampson. Authentication in the Taos Operating System. *ACM Transactions on Computer Systems*, 12, February 1994.

[12] X. Wu and D. K. Y. Yau. Mitigating Denial-of-Service Attacks in MANET by Distributed Packet Filtering: A Game-theoretic Approach. *ASIACCS*, March 2007.