Mobile Networks and Applications 10, 985–995, 2005

DOI: 10.1007/s11036-005-4454-4

Towards Designing a Trusted Routing Solution in Mobile Ad Hoc Networks*

TIRTHANKAR GHOSH and NIKI PISSINOU
Telecommunications and Information Technology Institute, College of Engineering, Florida International University, 10555 W Flagler Street, Miami, FL 33174, USA

KAMI (SAM) MAKKI
Department of Electrical Engineering & Computer Science, College of Engineering, University of Toledo, Toledo, OH 43606, USA

Published online: 24 October 2005

Abstract. Designing a trusted and secure routing solution in an untrustworthy scenario is always a challenging problem. Lack of physical security and low trust levels among nodes in an ad hoc network demands a secure end-to-end route free of any malicious entity. This is particularly challenging when malicious nodes collude with one another to disrupt the network operation. In this paper we have designed a secure routing solution to find an end-to-end route free of malicious nodes with collaborative effort from the neighbors. We have also extended the solution to secure the network against colluding malicious nodes, which, to the best of our knowledge, is the first such solution proposed. We have also proposed a framework for computing and distributing trusts that can be used with out trusted routing protocol. Our proposed framework is unique and different from the other schemes in that it tries to analyze the psychology of the attacker and quantifies the behavior in the computational model. Extensive simulation has been carried out to evaluate the design of our protocol.

Keywords: collaborative trust-based routing, secure communication, colluding malicious nodes, mobile ad hoc networks

1. Introduction

With the advent of mobile and ubiquitous computing over the last decade, another form of infrastructureless networking has emerged. Known as "ad hoc" networking, this form of peer-to-peer (or even multicast), multihop networking is growing more popular in an infrastructureless and untrustworthy environment (like the absence of access points or base stations). These ad hoc networks have given rise to active research issues since their evolution [6,7,10,12,19,22–26,28–30,33–39,41,42,50,51,53,54,56,58]. Most of the research, so far, has been done in the area of routing [7,13,20,23,24,34,35,37,41,52], though, in recent years, security issues have also been addressed [2,6,19,22,26,28,42,48,50,51,53,58]. Although the basic security goals and requirements of an ad hoc network are very similar to those of a wireless network, some inherent characteristics of the former make security issues more challenging.

Ad hoc networks are formed without the aid of any infrastructure. Hence, unlike an infrastructured wireless network, they cannot rely on any central entity for security issues. This lack of infrastructure has posed serious threats so far as routing security is concerned. Secondly, the vulnerability of the nodes to physical compromise gives rise to serious internal threats[1] within the network, which make the issues of authentication, integrity and confidentiality even more challenging than conventional wireless networks. The third most important characteristic of an ad hoc network is that the topology of the network changes dynamically. Thus, any security model, based on a fixed architecture cannot be used in such a scenario. In addition to the above three network centric features, the nodes in an ad hoc network are characterized by their low battery power and limited computational abilities. These are even more prominent in applications like personal computing and small sensor networks. These restrictions seriously limit the ability of the nodes to perform intensive computations.

Worse still, ad hoc nodes are characterized by minimum trust for each other. This requires an efficient protocol for finding secure end-to-end routes, free of non-trusted nodes. Most of the work on routing security focus on the efficient use of digital signatures or shared secret keys to authenticate and confide the data and routing headers. However, they always tend to find the shortest path between source and destination irrespective of the presence of malicious nodes in between. To overcome this problem and in quest for a trusted routing solution, we have designed a protocol capable of finding secure end-to-end paths and can also prevent any attack from colluding malicious nodes.

*Partially funded by Department of Defense Award No. H98230-04-C-0460, Department of Transportation Project No. FL-26-7102-00 and National Science Foundation Grant Nos. ANI-0123950 and CCR-0196557.

[1] We define an internal threat as an active attack by a compromised node or a node that has ended its loyalty to the network. These nodes can actively take part in ongoing communication, as they possess all relevant network secrets.

The rest of the paper is organized as follows. We give an overview of related work in Section 2. More detailed discussion on the related issues may be found in [31]. Section 3 discusses the design of our protocol, followed by a detailed description of trust modeling in Section 4. Section 5 evaluates the protocol with extensive simulation results and discussing different threat scenarios. Finally, Section 6 concludes the paper.

2. Related work

Although security issues in ad hoc networks have drawn considerable attention over the past few years, no solution has been proposed so far to secure the network against an internal attack by colluding malicious nodes by discovering a trusted end-to-end route. Most of the work done so far [18,19,22–27,32,36,39,58], fail in the face of an active internal attack where the adversaries have the network secrets in their possession. The threat is more severe when more malicious nodes act in collusion.

In [39], the authors have proposed an authenticated routing (ARAN) for ad hoc applications. The protocol works under the assumption of the existence of a trusted certificate server. In [24], the authors have proposed a new on-demand secure routing protocol, called Ariadne. The working of the protocol relies on the distribution of shared secret keys between source and destination, which itself is a burning research problem in an ad hoc network without the presence of any trusted entity. While Ariadne uses an end-to-end security solution, a hop by hop approach is proposed in [23]. The security is based on the efficient use of one-way hash function, unlike the use of MAC in Ariadne. The protocol works under the assumption that some secure means of distributing the elements of the hash chain is already there.

In [56], the authors have proposed an extension of the existing AODV [36] routing protocol using digital signature to secure the non-mutable fields of the AODV messages and hash chains to secure the hop count information, which is the only mutable Field in AODV. The protocol works under the assumption of the existence of an efficient key management system, which enables all the ad hoc nodes to obtain public key information of all other nodes. The authors also did not consider the problem of compromised nodes, which they think is not critical in non-military application. The security of the proposed scheme is limited by an attacker who deliberately keeps the information unchanged and can force the source node to select the path. Another extension of AODV is proposed in [50]. The protocol requires each node to carry a token signed with system secret key. The authors have referred to threshold cryptographic scheme to distribute the token secretly among nodes. In [6] the authors have proposed a CONFIDANT protocol based on DSR [26]. The monitoring mechanism is implemented by a neighborhood watch concept where the no-forwarding behavior of the nodes are monitored and reported.

All the protocols discussed so far use a secure way of route discovery, not considering any secure means to discover the topology of the network [34,35]. In [35], the authors have proposed a protocol for securely discovering the network topology in a public key infrastructure. However, the protocol fails in face of a colluding attack, and also does not scale well to frequent topology changes. Another protocol to achieve a similar goal is proposed in [34]. The authors have proposed a secure source-routing protocol that can securely discover correct connecting information in the network. The protocol works under the assumption of an already established shared secret between the source and the destination. However, an internal attacker in the network, who can get hold of the shared secret, can easily place itself on the end-to-end route, thus getting hold of all the data packets. Moreover, the protocol does not secure the network from an attack by colluding malicious nodes.

All the proposed solutions discussed above have used either a symmetric or a public key cryptosystem, intending to use a shared secret key or a digital signature to authenticate and protect the routing information. But they always tend to find the shortest path from source to destination irrespective of some malicious nodes in between. In [52], the authors have proposed a secured routing protocol based upon the trust level of the nodes. They have defined a security metric and embedded it into the RREQ packet. When an intermediate node receives a RREQ with a specified security metric or trust level, it can only process or forward the packet if it meets the required security level. However, a compromised node can effectively place itself into the route by manipulating its trust level. The authors also did not discuss any model for computing and distributing the trust levels.

Establishing security associations based on distributed trust among nodes is an important consideration while designing a secure routing solution. Not much work has been done to develop a trust model to build-up, distribute and manage trust levels among the ad hoc nodes. Most of the proposed schemes talk about the general requirement of trust establishment [7,16,27,47], but do not come up with any specific model or computational framework to do so. None of the models proposed so far have tried to analyze the psychology of the attacker and quantify those behaviors in the computational framework.

Modeling and computing trust for a distributed environment has been actively researched for quite a long time [1,4,59], though not much work has been done to extend the concept in ad hoc networks. Most of these distributed trust models combine direct and recommended trusts to come up with trust computations. Watchdog mechanism [32], based on promiscuous mode operation of the ad hoc nodes, has been the fundamental assumption in any trust computational model. In [49] the authors have proposed a trust evaluation-based secure routing solution. However, the mechanism for collecting the required parameters was not discussed by the authors. Also, some of the parameters suggested by the authors are not realistic in a highly sensitive application. In [38] a similar concept has been proposed. The trust computation is

based only on the success and failure of transmission of different packets and does not take into account different forms of malicious behavior. In [33] the authors have proposed an authentication scheme based on Public Key infrastructure and distributed trust relationship. The trust relationship is established by direct as well as recommended trusts. Composite trust is computed by combining both direct and recommended trust relationships. Some work has also been done to establish trust based on distribution of certificates. In [12] the authors have proposed such a trust management scheme. However, the proposed scheme lacks any specific framework for computing the indices. Another model has been proposed based on subjective logic [30]. This model fails to protect the network from an internal attack, where a malicious node either refuses to forward the packets and duly authenticates itself to the source, or it cooperates with the source node and acts as a black hole. Some mechanisms have been proposed to give incentives to the nodes for acting unselfishly. In [21] authors have proposed a secure reputation-based incentive scheme (SORI) that prevents the nodes from behaving in a selfish way. The scheme, however, does not prevent a malicious node from selectively forwarding packets or from other malicious behavior.

3. Design of the trusted routing protocol

Our trusted routing protocol is based on the following assumptions that we think are justified. First, all the nodes communicate via a shared wireless channel and all communication channels are bi-directional. Second, all the nodes operate in a promiscuous mode. Third, we assume a reliable link layer protocol to be in place. Our main focus is on the network layer and the protocol that we propose here is an extension of the Ad hoc On Demand Distance Vector (AODV) [36] routing protocol which we call it Trust-embedded AODV (T-AODV). Last but not least, we assume that all the nodes are identical in their physical characteristics, i.e., if node A is within the transmission range of B, then B is also within the transmission range of A.

3.1. Protocol description

Essentially all routing protocols in the ad hoc community tend to find the shortest path to the destination irrespective of the presence of any malicious node in that path. We can argue that, as internal threat in the network in the form of a compromised or disloyal node is of significant concern, a path free of malicious node is more important than the shortest path. In the following section we present a detailed description of our T-AODV.

3.1.1. High level description of T-AODV

When a node wants to find a route to another node, it initiates a route discovery. The RREQ packet header contains a *trust_level* field, in addition to the other fields in AODV RREQ. When an intermediate node receives the RREQ packet, it rebroadcasts it after modifying the *trust_level* field to include the trust level of the node that sends it the RREQ. Every node checks back the rebroadcasted RREQ packet from its next node to see whether it has provided the proper information. If not, it immediately broadcasts a warning message questioning the sanctity of that node. Our protocol does not encourage any intermediate node to send a route reply. The final route selection is based upon the *trust_level* metric. *Hop_count* plays a role in deciding the final route only when more than one packet has same *trust_level*. The RREP packet has the next hop information. This is in line with the solution given in [13] to counter the black hole problem. When the source node gets back the first RREP, it waits for a specified amount of time before using that route. If within that time another RREP comes, the source node queries the next hops of the two RREPs. The next hop of the malicious RREP will obviously not have the same route to the destination. Thus, malicious route injection into the network can be prevented. The procedure below shows the action of a node after it receives a route request packet.

```
// when a node receives a Route Request packet
   Receice_RREQ( ) {
   // check whether it is the destination of the route request
   if destination
      compute_highest_trust_level( )
      // in case more than one RREQ has same trust_level
      // decides on the basis of lowest hop_count
      sends_RREP_to_source( )
   else (not destination)
      if duplicate packet
         cross_checks_trust_level( )
         if found ok
           drops the packet
         else
           broadcasts roure_warning message( )
         end if
      else (not duplicate)
         modifies trust_level
         increments hop_count
         rebroadcasts RREQ
      end if
   end if
   }  // end of function Receice_RREQ
```

Procedure to show the action of a node after receiving the RREQ packet

To summarize, a node first checks whether it is the destination of the packet. If it is the destination, it creates a route reply and sends it back along the reverse route. If it is not the destination, it checks whether the packet is duplicate. If found duplicate, the node cross checks its trust level provided by its neighbor and takes action according to the correctness of the information. If the packet is not a duplicate, it appends the necessary information and rebroadcasts it.

The procedure below shows the detailed action of the source node after it receives the first route reply.

```
// when the source node gets back the first Route Reply
 Receive_RREP( ) {
   waits for a specified period
   if receives another RREP
     queries next_hop( )
   else
     sends data( )
   end if
 } // end of function Receive_RREP
```

Procedure to show the action of the source node after receiving the first route reply

The function *cross_checks_trust_level* can be implemented in two ways. When an intermediate node receives a duplicate route request packet, it checks back the *hop_count* field to find out from which node it is receiving the packet. The following algorithm implements this function.

```
if (current ->hop_count = = hop_count - 1)
   cross_checks_trust_level( )
else
   // the node is trying to put malicious information
   // finds out which node is malicious
   broadcasts roure_warning message( )
end if
```

Cross checks trust level function

The above algorithm works under the assumption that all the nodes are identical in their radio range. If they are not, a node can receive a duplicate route request from any other node which it cannot reach directly and wrongly assume that the later is trying to act malicious. This will generate false warning messages in the network.

The second possible implementation takes care of the above assumption. An intermediate node, on receiving a duplicate route request packet, extracts the address stored in the *lastaddr* field (the *lastaddr* field contains the address of the node from which the next node receives a route request packet) and checks from the neighbor table whether it is from any of its neighbor. The algorithm works as follows:

```
if (lastaddr = = neighbortable->addr)
   cross_checks_trust_level( )
else
   drops the packet( )
end if
```

Cross checks trust level function alternative implementation

The above implementation can actually increase the computational overhead in each node. However, the computational overhead can be reduced by efficient searching of the neighbor table. We recognize that, the protocol that we have designed so far, has one possible vulnerability. It fails to secure the network against multiple malicious nodes colluding together. In the following section we discuss a threat model with multiple colluding malicious nodes and design solutions to secure against it.

3.2. Threat model

As we have discussed in the earlier section, our secure routing protocol fails to secure the network against multiple malicious nodes colluding together. In this section we describe the threat model and propopse a secure solution for it. Let us consider the following example shown in figure 1 below.

M1 gets a RREQ from A (it also gets a copy from B which is redundant). It rebroadcasts the RREQ packet to M2 and C. Let us assume that M1 and M2 are malicious and colluding to disrupt the routing operation. M2 appends a high trust level

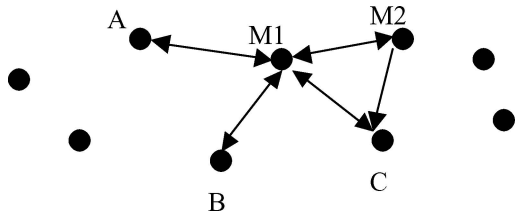

Figure 1. An example of the threat model.

for M1 and rebroadcasts the packet. C, however, appends the right trust level. M1 broadcasts a RWARN message that C is malicious. C subsequently gets isolated from the network and the route through M1 and M2 is selected, as it has a high trust level. Thus, the malicious nodes collude with each other to bring down the entire network. The sheer purpose of finding a secure end-to-end route is defied and trusted nodes are isolated from the network as malicious.

3.3. Preventing colluding attack

The fact that ad hoc nodes are characterized by low level of trust among themselves, motivated us to design a secure algorithm based on internal spying and verification. We recognize the need of developing a mechanism to verify the claim of an accuser accusing another node of malicious behavior. For example, when M1 broadcasts a RWARN message to A and B about C's trustworthiness, there must be a mechanism by which both A and B check back M1's accusation.

Our algorithm to counter the colluding attack assumes the existence of a Public Key infrastructure. Each node has a <Public Key (PK), Private Key (SK)> pair, the generation of which can be done by any existing algorithm. Further assumptions are already discussed in Section 3.1. We extended the T-AODV protocol discussed in Section 3.1.1 to incorporate the security needed to counter the colluding attack. Each node, before broadcasting the RREQ packet, not only computes the *trust_level* field, but also computes a signature and appends it to the RREQ packet header. The signature is computed as follows:

$$Sign_i = (Source\_Address,\ \ Broadcast\ ID,\ trust\_level_{i-1},\ IP\_Address_i : SK_i$$

where, $trust\_level_{i-1}$ = trust level of the node from which node i receives the RREQ,

$$IP\_Address_i = \text{IP address of node } i, \text{ and}$$

$$SK_i = \text{the private key of node } i.$$

After receiving the RREQ packet, a node, besides computing the *trust_level*, concatenates the source address, broadcast ID and its own IP address with the trust level of the node from which it receives the RREQ and signs it with its private key to compute the signature. It then appends the signature in the RREQ packet before broadcasting it to its neighbors. When a node questions about another node's trustworthiness and

broadcasts a route warning (RWARN) message, it not only sends the IP address of the accused node, but also the signature provided by the later. The RWARN message structure is shown below:

Source Address	Broadcast ID	Malicious node IP	Signature	RWARN source IP

The RWARN message structure

The inclusion of source address and broadcast ID into the signature is to prevent any replay attack that a malicious node can carry out. A malicious node can copy the signature field from the RREQ packet and use it at a later time to falsely accuse another node after its own trust level has changed. The inclusion of the source address and broadcast ID fields in the signature generation can successfully prevent such a replay attack.

When a node receives a RWARN message, it verifies the accusation by the sender of the message. It decrypts the signature with the public key of the accused node and checks whether the trust level matches the trust level of the accuser. If the two trust levels match, the node concludes that the accused node has provided right information, and hence cannot be malicious. Thus the trustworthiness of the accuser is in question. However, if the trust levels are different, then it concludes that the accused node is malicious. The procedure is shown below:

```
receive RWARN ( ) {
decrypt_MAC ( ) {
if (decrypt = yes)
     Check trust level provided by the accused node
     if (trust_level provided = trust_level of the accuser)
       the accuser is malicious
     else
       the accused node is malicious
     end if
else (decrypt = no)
   the accuser is malicious
end if
    } //end of function decrypt_MAC
} //end of function receive RWARN
```

The receive RWARN function

4. Trust model

Our motivation for developing the trust model is to form a true and honest impression about the trustworthiness of the nodes and to punish the nodes with the slightest malicious intention. To do this we need to understand clearly the ways a node can engage itself in different malicious acts. Below we highlight three different malicious behavior.

1. A node engaging in selfish behavior by not forwarding packets meant for other nodes, or selectively forwarding smaller packets while discarding larger ones.
2. A node falsely accusing another node for not forwarding its packets, thus isolating the node from normal network operation.
3. A node placing itself in active route and then coming out to break the route, thus forcing more route request packets to be injected into the network. By repeating this malicious act, a large number of routing overhead is forcefully generated wasting valuable bandwidth and disrupting normal network operation.

4.1. The model

Our model has been developed with a view to form a true and honest opinion about the trustworthiness of the nodes with collaborative effort from their neighbors. Our trust model is not transitive, i.e., we do not consider the notion "if A trusts B and B trusts C, then A trusts C". This is to prevent any colluding malicious behavior among nodes where two or more nodes can conspire to claim themselves trustworthy. In the following section we analyze different malicious behavior and quantify them to gradually develop the model.

4.1.1. Trust model against selfish behavior

The development of the model to punish a node for selfish behavior is based on the Secure and Objective Reputation-based Incentive (SORI) scheme proposed in [21] with several modifications. We will elaborate more on these modifications as we describe the trust model. The parameters used for the model are described below:

NNL_N = neighbor node list (each node maintains a list of its neighbors, either by receiving *Hello* messages, or by learning from overhearing).

$RF_N(X)$ (Request for forwarding) = total number of packets node N has forwarded to node X for further forwarding.

$HF_N(X)$ (Has forwarded) = total number of packets that have been forwarded by X and noticed by N.

We are not discussing the details of updating these parameters, which can be found in [50]. With the above parameters, node N can create a *local evaluation record* (denoted by $LER_N(X)$) about X. The record $LER_N(X)$ consists of two parameters shown below:

$$LER\_N(X) = \{G_N(X),\ C_N(X)\}$$

where, $G_N(X)$ is the forwarding ratio given by $G_N(X) = (HF_N(X) / RF_N(X))$

$C_N(X)$ is the confidence level of N on X

In [21] the authors have set $C_N(X) = RF_N(X)$. This gives quite an accurate estimation about the trustworthiness of a node when weighted by the confidence level. But the trust computation does not take into account a node's "selective forwarding" behavior, where it only forwards small packets while selectively discarding larger ones. To reflect this kind of malicious behavior in our trust model, we compute the confidence level $C_N(X)$ as given below:

$$C_N(X) = \frac{\Sigma(HF_N(X)/RF_N(X))\, x\ Pkt\_size}{\Sigma Pkt\_size} \qquad (1)$$

Node N computes its confidence level on X after sending a specified number of packets to X. The computation is

weighted by the packet size to reflect the "selective forwarding" behavior of a node.

We propose a similar propagation model proposed in SORI [21]. Each node updates its local evaluation record (LER) and sends it to its neighbors. When a node N receives the $LER_i(X)$ from node I, it computes the *overall evaluation record* of X (denoted by $OER_N(X)$), as given below:

$$OER_N(X) = \frac{\Sigma_{i \in NNL,\, i \neq X}\, C_N(i)x\, C_i(X)xG_i(X)}{\Sigma_{i \in NNL,\, i \neq X}\, C_N(i)x\, C_i(X)} \quad (2)$$

where, $C_N(i)$ = confidence level of node N on node i from which it receives$LER_i(X)$

$$C_i(X) = \text{confidence level of node } i \text{ on node } X$$

$$G_i(X) = \text{forwarding ration of node } i \text{ on } X$$

$LER_i(X)$, $C_i(X)$ = confidence level of node i on node X, $G_i(X)$ = forwarding ratio of node i on X

4.1.2. Trust model against malicious accuser

In this section we extend the above trust model against selfish behavior to take into account the malicious accusation of a node about another node. We foresee a threat where a node falsely accuses another node of not forwarding its packets, eventually to isolate that node as an untrustworthy one. This malicious act should also be reflected in the trust computation, where every node should be given a chance to defend itself. We have modified equation (1) above to reflect such a malicious act in the computation of the confidence level. The modified equation is shown below:

$$C_N(X) = \frac{\Sigma(HF_N(X)/RF_N(X))\, x\, Pkt\_size}{\Sigma\, Pkt\_size}\, x\, \alpha_x(N) \quad (3)$$

$$\text{where, } \alpha_x(N) = \text{accusation index of } N \text{ by } X$$

$$= \begin{cases} 0; & \text{if } X \text{ falsely accuses } N \\ 1; & \text{otherwise} \end{cases}$$

Node N keeps a track of the packets it received from X and packets it forwarded. If N finds out that X is falsely accusing it for non-cooperation, it recomputes its confidence level on X by taking into account the accusation index. It then broadcasts the new $LER_N(X)$ with new $C_N(X)$, thus resulting in computation of a new $OER_N(X)$, which is low enough to punish X. Thus, any sort of malicious behavior of X by falsely accusing other nodes gets punished eventually.

4.1.3. Trust model against malicious topology change

In this section our proposed model is extended to reflect the malicious behavior of a node where it forces the network topology to change frequently, eventually generating a large overhead. If such a behavior is detected, the confidence level must be changed in order to punish the malicious node. However, detection of such a behavior is not easy, as any such topology change can be viewed as a normal characteristic of an ad hoc network. We have tried to capture such a malicious act by statistically modeling the action and reflecting it in the computation of trust.

Table 1
Snapshot of neighbor temove table.

Node address	Time of leaving	Time difference
X	T1	t0 = 0
X	T2	t1 = T2 − T1
X	T3	t2 = T3 − T2
X	T4	t3 = T4 − T3
		Mean = μ_t

To develop the model, we require each node to maintain a table called a *neighbor remove table*, where it keeps track of any node moving out of the path. The table is populated by successive *Hello* misses in AODV, or from the *unreachable node address* field in the RERR packet in DSR. A snapshot of the Table 1 is shown below:

Each node periodically scans the table to find whether any particular node is leaving at frequent intervals. It computes the mean, μ_t of the time difference of any particular node leaving the network. If μ_t is found lower than a threshold value (denoted by $t_{\text{threshold}}$), then the node is identified as malicious and the confidence level is computed as follows:

$$C_N(X) = \frac{\Sigma(HF_N(X)/RF_N(X))\, x\, Pkt\_size}{\Sigma\, Pkt\_size}\, x\, m(X) \quad (4)$$

where m(X) = malicious index of node X

$$= \begin{cases} 0; & \text{if } \mu_t \leq t_{\text{threshold}} \\ 1; & \text{if } \mu_t > t_{\text{threshold}} \end{cases}$$

The choice of the threshold value can be selected based on the typical application for which the ad hoc network is deployed. A network that demands frequent topology change can have a higher threshold to accommodate the normal network behavior. The choice is left as an administrative decision and not discussed in this paper.

Finally, to combine all the malicious behavior discussed earlier and to reflect those behavior in trust computation, the confidence level of node N on X is computed as shown below:

$$C_N(X) = \frac{\Sigma(HF_N(X)/RF_N(X))\, x\, Pkt\_size}{\Sigma\, Pkt\_size} \times x\, \alpha_x(N)\, x\, m(X) \quad (5)$$

The final overall evaluation record (OER), when computed based on the local LERs, will reflect the different malicious behavior of a node as computed in the confidence level, and finally any malicious act gets detected and punished.

5. Protocol evaluation

In the following sections we evaluate the efficiency of our protocol through simulation and also analyze its security by evaluating different threat scenarios.

Table 2
Parameters chosen for simulation.

Scenario 1	Independent variable	Set of parameters compared		
	Number of nodes	Routing overhead	Number of routes selected	Number of route errors
Scenario 2	Independent variable	Set of parameters compared		
	Node speed	Routing overhead	Number of routes selected	Number of route errors

5.1. Simulation and results

We have used Glomosim [57] for our simulation. Glomosim is a scalable simulation software used for mobile ad hoc networks. We have carried out the simulation with two different scenarios. We defined a region of 2 km by 2 km and placed the nodes randomly within that region. In the first scenario, the nodes moved with uniform speed chosen between 0 to 10 m/sec with 30 sec pause between each successive movement. We increased the number of nodes and studied the network performance. In the second scenario, we have increased the node speed, keeping the similar infrastructure, to carry out our analysis. With these two scenarios, we are able to evaluate the scalability of our protocol with increased node participation and more frequent topology changes. The parameters for both the scenarios are shown in the Table 2 below.

In our earlier work, when we designed the T-AODV routing protocol, we found that it had a very small increase in routing overhead than AODV, which we think can be traded off with the incorporation of security into the protocol. We have compared the routing overhead for AODV, T-AODV and modified T-AODV in figure 2, from which we can conclude that modified T-AODV also has a small increase in its overhead. This increase in overhead is due to retransmission of some route request packets because of delayed receipt of route reply by the source nodes as we do not encourage intermediate nodes to send route reply in our protocol. Actually, this overhead can be brought down by increasing the NET_TRAVERSAL time which will allow the source node to wait for a longer period of time to receive route reply packets. The average percentage increase in overhead for modified T-AODV than AODV has been found to be 5.5%.

As no intermediate node is encouraged to come up with route replies, we obviously have lesser number of routes selected in T-AODV and modified T-AODV than that in AODV. This can be seen from figure 3, which compares the number of routes selected for all the three protocols. However, this should not give any misconception that some of the routes are not properly selected. In fact, both T-AODV and modified T-AODV have lesser number of route errors reported than that in AODV as can be seen from figure 4. Lesser number of routes selected, in effect, renders lower processing overhead for the source nodes, as they do not have to process all the route replies from the intermediate nodes.

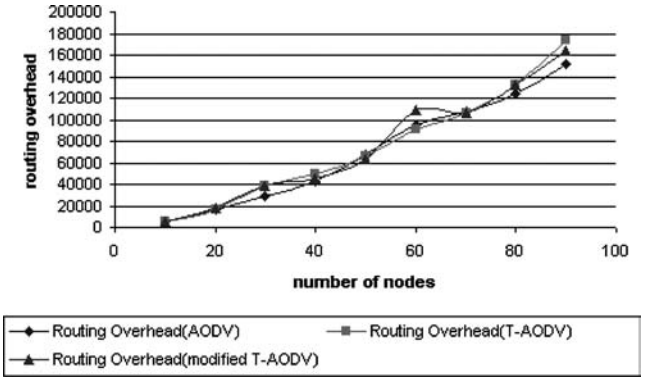

Figure 2. Comparison of routing overhead.

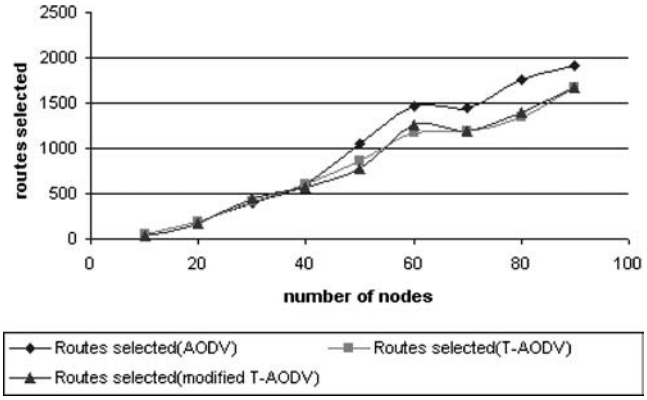

Figure 3. Comparison of number of routes selected.

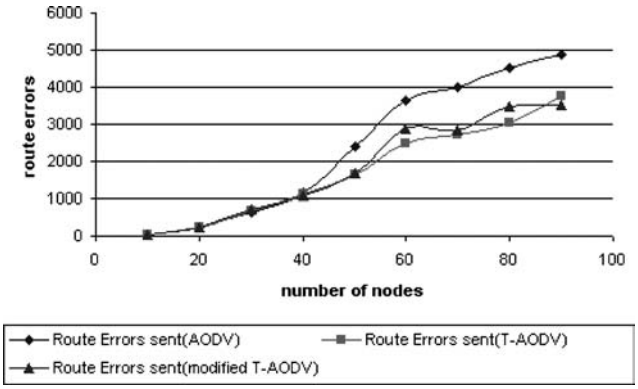

Figure 4. Comparison of route errors sent with number of nodes.

The results from our second set of simulations are shown below. We have varied the speed of node movement from 5 m/sec to 30 m/sec and compared our modified T-AODV protocol with the original AODV. We can see from the comparison of routing overhead (figure 5) that the modified T-AODV performs better than AODV with higher speed of node movement. The possible explanation for this is that, the topology of the network changes faster with the faster movement of nodes; as a result of which more route requests are generated. In AODV more and more intermediate nodes come up with route replies, which increases the overhead. Whereas, in modified T-AODV, only destination nodes come up with replies,

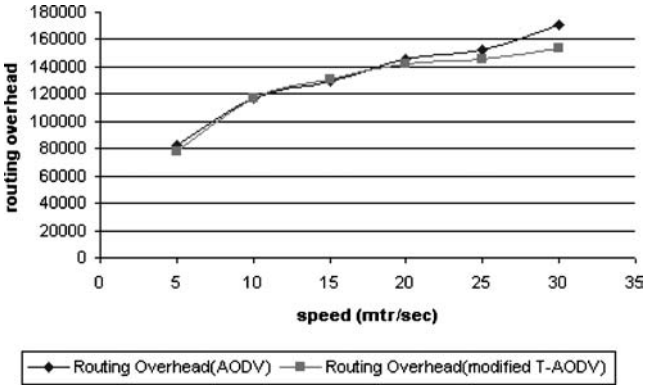

Figure 5. Comparison of routing overhead with node speed.

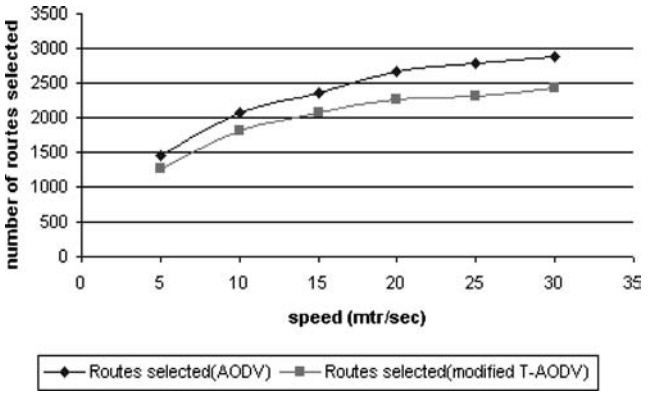

Figure 6. Comparison of routes selected with node speed.

hence the overhead is smaller. The small increase in overhead for modified T-AODV with lesser speed is because of the retransmission of some route request packets due to delayed receipt of route reply by the source nodes. This overhead can be brought down by increasing the NET_TRAVERSAL time, as we have discussed earlier.

Figures 6 and 7 respectively compare the number of routes selected and the number of route errors sent with node speed. We can see the efficiency of our protocol from these two parameters also. Significantly lesser number of routes selected by modified T-AODV at higher node speed imposes lesser processing overhead at the nodes. A significant reduction of route errors can also be observed for modified T-AODV at higher speed.

To conclude briefly, we have also noted that the average running time for modified T-AODV is about 11% higher than that of T-AODV. This increase in the running time is because of the incorporation of cryptographic operations into modified T-AODV protocol. In the following section, we analyze the security of the protocol by evaluating different threat scenarios.

5.2. *Security analysis of the protocol*

The security of our protocol lies in the verification of the information provided by other nodes. We evaluate different scenarios of attack by a malicious entity, acting either independently or in collusion, and show that the protocol is secure against these attacks.

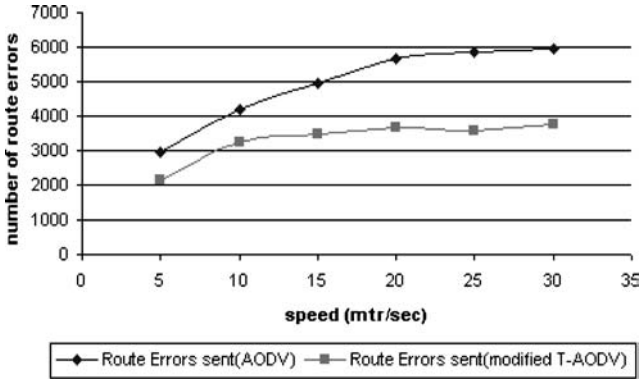

Figure 7. Comparison of route errors with node speed.

Scenario 1: A malicious node wants to include itself into the path and provides wrong information in the RREQ packet—this attack has already been prevented while designing the T-AODV protocol. The malicious node is effectively isolated by the collaborative effort of its neighbors.

Scenario 2: A node falsely accuses another node and alters the information provided by the later—the accuser has to append the signature computed by the accused node. In order to alter the information, it has to decrypt the signature, change the original information and recompute it. But it fails to recompute the signature as it lacks the knowledge of the accused node's Private key. Thus any attempt to alter the original information gets detected.

Scenario 3: A node falsely accuses another node, alters the information provided by the later and recomputes the signature with its own Private key—this malicious act gets detected, as the nodes receiving the warning messages cannot decrypt the signature using the accused node's Public key.

Scenario 4: A node falsely accuses another node and provides the signature of a different node other than the accused one—this act also gets detected, as the neighboring nodes receiving the warning messages cannot decrypt the signature using the accused node's Public key.

Scenario 5: A node whose trust level has changed, falsely accuses another node by using a copy of the old Signature field that the later used at some earlier point of time. This false accusation gets detected as the decryption of the signature will reveal the actual source address and broadcast ID pair.

We recognize that these scenarios are not exhaustive but at least demonstrate that the protocol is secure under these threats. We are currently working on developing a formal model that proves the security of the protocol.

6. Conclusion

We have proposed a secure routing protocol for ad hoc networks where a secure end-to-end path is found free of malicious nodes. Any malicious entity, trying to inject wrong routing information either independently or acting in collusion, is effectively singled out. We have carried out extensive simulation to show the efficiency of the protocol. Comparison between AODV, T-AODV and modified T-AODV have shown that the protocol has a little more routing overhead but lesser number of routes selected and route errors. Moreover, modified T-AODV has been shown to be more efficient with higher node speed and frequent topology changes. The security of the protocol is also analyzed by considering different threat scenarios. We have also proposed a model for computing, distributing and updating trust information in the network that analyzes different malicious behavior reflecting each behavior in the model.

Our proposed solution for securing the network against colluding malicious nodes works in the case of false accusation. However, if there is no accusation at all from a malicious node, the protocol does not find a secure end-to-end path. This has been left as a future work and we are currently working on the solution.[2]

Acknowledgment

I want to acknowledge the anonymous reviewers for their valuable comments.

References

[1] A. Abdul-Rahman and S. Hailes, A distributed trust model, in: *ACM New Security Paradigm Workshop* (1997).

[2] P. Albers et al., Security in ad hoc networks: A general intrusion detection architecture enhancing trust based approaches, in: *Wireless Information Systems*, Ciudad Real, Spain (2002).

[3] Becker and Wille, Communication complexity of group key distribution, in: *Proceedings of the 5th ACM Conference on Computer and Communications Security* (San Francisco, California, United States, 1998) pp. 1–6, ISBN:1-58113-007-4.

[4] T. Beth, M. Borcherding and B. Klein, Valuation of trust in open networks, in: *Proceedings of the European Symposium on Research in Computer Security (ESORICS),* (Brighton, UK, 1994) pp. 3–18, LNCS 875, Springer-Verlag.

[5] M. Blum, How to exchange (secret) keys, ACM Transactions on Computer Systems 1(2) (1983) 175–193.

[6] S. Buchegger and J.-Y. Le Boudec, Performance analysis of the CONFIDANT protocol (Cooperation of nodes: Fairness in dynamic ad-hoc networks), in: *MOBIHOC '02* (Switzerland) (June 9–11 2002).

[7] S. Buchegger and J.-Y. Le Boudec, Nodes bearing grudges: towards routing security, fairness, and robustness in mobile ad hoc networks, in: *Proceedings of the Tenth Euromicro Workshop on Parallel, Distributed, Network-based Processing* (Canary Islands, Spain), (Jan. 2002) pp. 403–410.

[8] S. Buchegger and J.-Y. Le Boudec, The effect of rumor spreading in reputation systems for mobile ad-hoc networks, in: *Proceedings of WiOpt '03, Modeling and Optimization in Mobile Ad Hoc and Wireless Networks* (Sophia-Antipolis, France, March 2003).

[9] M.V.D. Burmester and Y. Desmedt, A secure and efficient conference key distribution system, in: A.D. Santis (ed.), *Advances in Cryptology—EUROCRYPT ''94*, vol. 950 of Lecture Notes in Computer Science, (Springer-Verlag, 1995) pp. 275–286.

[10] L. Buttyán and J-P Hubaux, Stimulating cooperation in self-organizing mobile ad hoc networks, MONET Journals of Mobile Networks (2002).

[11] S. Čapkun, J-P Hubaux, and L. Buttyán, Mobility helps security in ad hoc networks, in: *MobiHoc '03*, (Annapolis, Maryland, USA), June 1–3.

[12] C.R. Davis, A localized trust management scheme for ad hoc networks, in: *Proceedings of the 3rd International Conference on Networking (ICN '04)* (March 2004).

[13] H. Deng, W. Li and D.P. Agrawal, Routing security in wireless ad hoc networks, IEEE Communications Magazine (2002).

[14] Y. Desmedt, Society and group oriented cryptography: A new concept, in: *Advances in Cryptology—Crypto'87* (1987) p. 120–127.

[15] W. Diffie and M.E. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory IT-22(6) (1976) 644–654.

[16] L. Eschenauer, V.D. Gligor and J. Baras, On trust establishment in mobile ad hoc networks, in: *Proceedings of the Security Protocols Workshop*, (Springer-Verlag, Cambridge, U.K. April 2002).

[17] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, Robust and efficient sharing of RSA functions, in: *Crtpto '96* (1996).

[18] R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin, Robust threshold DSS signatures, *Advances in Cryptology—Eurocrypt '96* (Springer-Verlag 1996).

[19] T. Ghosh, K. Makki and N. Pisinou, An overview of security issues for multihop mobile ad hoc networks, Network Security: Technology Advances, Strategies, and Change Drivers (2004) ISBN: 0-931695-25-3.

[20] T. Ghosh, N. Pissinou and K. Makki, Collaborative trust-based secure routing against colluding malicious nodes in multi-hop ad hoc networks, in: *Proceedings of the 29th IEEE Annual Conference on Local Computer Networks (LCN)* (Tampa, USA Nov 16–18 2004).

[21] Q. He, W. Dapeng and P. Khosla, SORI: A secure and objective reputation-based incentive scheme for ad-hoc networks, WCNC (2004).

[22] M. Hietalahti, Key establishment in ad hoc networks, Technical Report, Lab of Theoretical Computer Science, Helsinki University of Technology (2001).

[23] Y-C Hu, D.B. Johnson and A. Perrig, SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks, in: *Fourth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '02)* (June 2002), p. 3–13.

[24] Y-C Hu, A. Perrig and D.B. Johnson, Ariadne: A secure on-demand routing protocol for ad-hoc networks, in: *MobiCom '02,* (Atlanta, Georgia, USA) (Sept. 23–26 2002)

[25] J-P Hubaux, L. Buttyán and S. Čapkun, The quest for security in mobile ad hoc networks, in: *MOBIHOC* (2001).

[26] D.B. Johnson and D.A. Maltz, The dynamic source routing protocol for mobile ad hoc networks, in: *Internet Draft, MANET Working Group, IETF* (Oct. 1999).

[27] L. Kagal, T. Finin and A. Joshi, Moving from security to distributed trust in ubiquitous computing environments, IEEE Computer, (Dec. 2001.)

[28] J. Kong, P. Zerfos, H. Luo, S. Lu and L. Zhang, Providing robust and ubiquitous security support for mobile ad-Hoc networks, in: *International Conference on Network Protocols (ICNP)* (2001).

[29] P. Krishna, M. Chatterjee, N.H. Vaidya, and D.K. Pradhan, A cluster based approach for routing in ad hoc networks, in: *Second USENIX Symposium on Mobile and Location Independent Computing* (April 1995).

[2] We define an internal threat as an active attack by a compromised node or a node that has ended its loyalty to the network. These nodes can actively take part in ongoing communication, as they possess all relevant network secrets.

[30] X. Li, M.R. Lyu and J. Liu, A trust model based routing protocol for secure ad hoc networks, in: *Proceedings 2004 IEEE Aerospace Conference* (Big Sky, Montana, USA). (March 6–13 2004).

[31] W. Lou and Y. Fang, A survey of wireless security in mobile ad hoc networks: challenges and available solutions in: X. Cheng, X. Huang and D.Z. Du (Eds.), *Ad Hoc Wireless Networking* (Kluwer Academic Publishers, 2003) pp. 319–364.

[32] S. Marti, T.J. Giuli, K. Lai and M. Baker, Mitigating Routing Misbehavior in Mobile Ad Hoc Networks, in: *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking* (*MobiCom*) (Boston, Massachusetts, United States), August 06–11, 2000.

[33] E.C.H. Ngai and M.R. Lyu, Trust and clustering-based authentication services in mobile ad hoc networks, in: *Proceedings of the* 2nd *International Workshop on Mobile Distributed Computing (MDC '04)*, (Tokyo, Japan, March 23–26 2004).

[34] P. Papadimitratos and Z.J. Haas, Secure routing for mobile ad hoc networks, in: *Proc. SCS Communication Networks and Distributed Systems Modeling and Simulation Conference(CNDS 2002)* (San Antonio, TX, Jan. 27–31 2002).

[35] P. Papadimitratos and Z.J. Haas, Secure link state routing for mobile ad hoc networks, in: *Proc. IEEE Workshop on Security and Assurance in Adhoc Networks,* in conjunction with the *2003 International Symposium on Applications and the Internet* (Orlando, FL, Jan. 28, 2003).

[36] C. Perkins and E. Royer, Ad hoc On-demand distance vector routing, in: *Proc. IEEE Workshop on Mobile Computing Systems and Applications* (1999).

[37] N. Pisinou, T. Ghosh and K. Makki, Collaborative trust based routing in multihop ad hoc networks, in: *Proceedings of Networking '04: Springer Verlag, Series: Lecture Notes in Computer Science,* vol. 3042 (Athens Greece, May 9–14, 2004) pp. 1446–1451.

[38] A.A. Pirzada and C. McDonald, Establishing trust in pure ad-hoc networks, in: 27th *Australian Computer Science Conference* (The Univ. of Otago, Dunedin, New Zealand 2004).

[39] K. Sanzgiri et al., A secure routing protocol for ad hoc networks, in: *Proc. of the 10*th *IEEE International Conference on Network Protocols (ICNP'02)* (2002).

[40] A. Shamir, How to share a secret, Commun. ACM 22 (1979) 612–613.

[41] B.R. Smith, S. Murthy and J.J. Garcia-Luna-Aceves, Securing distance-vector routing protocols, in: *Proceedings of Internet Society Symposium on Network and Distributed System Security* (San Diego, CA, Feb. 1997).

[42] F. Stajano and R. Anderson, The resurrecting duckling: security issues for ad hoc wireless networks, 15th Sept. (1999).

[43] M. Steiner, G. Tsudik, and M. Waidner, Diffie-Hellman key distribution extended to group communication, in: *Proc. 3*rd *ACM CCS* (New Delhi, India, 14–16 May 1996) pp. 31–37.

[44] M. Steiner, G. Tsudik and M. Waidner, Key agreement in dynaimc peer groups, IEEE Transactions on Parallel and Distributed Systems 1(8) (2000) 769–780.

[45] G. Theodorakopoulos and J.S. Baras, Trust evaluation in adhoc networks, in: *WiSe'04* (Philadelphia, Pennsylvania, USA Oct. 1 2004).

[46] G. Tsudik and E. Van Herreweghen, On simple and secure key distribution, *in: 1st Conf. Computer & Comm. Security '93-11/93* (VA, USA 1993).

[47] R.R.S. Verma, D. O'Mahony and H. Tewari, NTM—Progressive trust negotiation in ad hoc networks, in: *Proceedings of the 1st Joint IEI/IEE Symposium on Telecommunications Systems Research* (Dublin, Nov. 27 2001).

[48] K. Wrons, Distributed security: ad hoc networks & beyond, in: *Ad Hoc Networks Security Pampas Workshop* (Rhul, Sept. 2002) pp. 16–17.

[49] Z. Yan, P. Zhang and T. Virtanen, Trust evaluation based security solution in ad hocnetworks, http://www.nokia.com/library/files/docs/Trust_Evaluation_Based_Security_Solution_in_Ad_Hoc_Networks.pdf.

[50] H. Yang, X. Meng and S. Lu, Self-organized network layer security in mobile ad hoc networks, in: *WiSe '02* (Atlanta, Georgia, USA) (Sept. 28 2002).

[51] A. Yasinsac et al., A family of protocols for group key generation in ad hoc networks, in: *International Conference on Communications and Computer Networks (CCN02)* (Nov. 3–4, 2002).

[52] S. Yi, P. Naldurg and R. Kravets, Security-aware ad hoc routing for wireless networks, Report No. UIUCDCS-R-2001-2241, UILU-ENG-2001-1748 (Aug. 2001).

[53] S. Yi and R. Kravets, Key management for heterogeneous ad hoc wireless networks, Report No. UIUCDCS-R-2002-2290, UILU-ENG-2002-1734 (July 2002).

[54] S. Yi and R. Kravets, Composite key management for ad hoc networks, Report No.UIUCDCS-R-2003-2392, UILU-ENG-2003-1778 (2003).

[55] Y. Desmedt, Some recent research aspects of threshold cryptography, in: *Proceedings of the First International Workshop on Information Security* (1997) pp. 158–173, ISBN: 3-540-64382-6.

[56] M.G. Zapata and N. Asokan, Securing Ad hoc Routing Protocols, in: *WiSe'02* (Atlanta, Georgia, USA, Sept. 28, 2002).

[57] 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 Simulations – PADS '98* (May 26–29, Alberta, Canada, 1998).

[58] L. Zhou and Z.J. Haas, Securing ad hoc networks, IEEE Network (Nov./Dec. 1999).

[59] H. Zhu, B. Feng and R.H. Deng, Computing of trust in distributed networks, http://eprint.iacr.org/, 2003/056.

Tirthankar Ghosh is a PhD candidate in the Telecommunications and Information Technology Institute at Florida International University. His area of research is routing security and trust computation in wireless ad hoc and sensor networks. He received his Bachelor of Electrical Engineering from Jadavpur University, India and Masters in Computer Engineering from Florida International University. E-mail: tirthankar.ghosh@fiu.edu

Dr. Niki Pissinou received her Ph.D. in Computer Science from the University of Southern California, her M.S. in Computer Science from the University of California at Riverside, and her B.S.I.S.E. in Industrial and Systems Engineering from The Ohio State University. She is currently a tenured professor and the director of the Telecommunication & Information Technology Institute at FIU. Previously Dr. Pissinou was a tenured faculty at the Center for Advanced Computer Studies at the University of Louisiana at Lafayette where she was also the director of the Telecommunication & Information & Technology Laboratory partially funded by NASA, and the co-director of the NOMAD: A Wireless and Nomadic Laboratory partially funded by NSF, and the Advanced Network Laboratory. Dr. Pissinou is active in the fields computer networks, information technology and distributed systems. E-mail: pissinou@fiu.edu

Dr. Kami (Sam) Makki has earned his Ph.D. in Computer Science from the University of Queensland in Brisbane Australia, his Masters degree in Computer Science and Engineering from the University of New South Wales in Sydney Australia, and his Bachelor and Masters Degrees in Civil Engineering from the University of Tehran Iran. Before joining the department of Electrical Engineering and Computer Science at the University of Toledo he has held a number of

academic positions and research appointments at the Queensland University of Technology in Brisbane, Royal Melbourne Institution of Technology in Melbourne and at The University of Queensland in Brisbane Australia. He is an active researcher in the fields of distributed systems, databases, mobile and wireless communications, and has more than 30 publications in peerreviewed journals and international proceedings. He has served as a chair and technical program committee member and reviewer for a number of IEEE and ACM sponsored technical conferences and has received a number of achievement awards.
E-mail: kmakki@eecs.utoledo.edu