Secure Multicast Groups on Ad Hoc Networks

T. Kaya, G. Lin, G. Noubir, A. Yilmaz
College of Computer and Information Science
Northeastern University, Boston, MA, USA.
{tansel, lingl, noubir, ati}@ccs.neu.edu

ABSTRACT

In this paper we address the problem of secure multicast of data streams over a multihop wireless ad hoc network. We propose a dynamic multicast group management protocol that aims at solving problems that are specific to ad hoc networks such as mobility, unreliable links, and cost of multihop communication. The main idea is to have group members actively participate to the security of the multicast group, therefore reducing the communication and computation load on the source. Since the group security is distributed among the group members, we propose a service right certificate, to verify that a node is authorized to join the group, and also a corresponding revocation mechanism. We simulated our protocol within the *ns*-2 environment under various mobility, group size, and group dynamic scenarios. Our simulation results indicate that the communication cost and join delay of the protocol scale well when the group dynamic and nodes mobility increase. We have implemented the basic protocol in our ad hoc network testbed. We also proposed an extension to the basic secure multicast tree using multi-link capability combined with a *k*-out-of-*N* coding approach.

Categories and Subject Descriptors

D.3.3 [**Computer-Communications Networks**]: Network Protocols.

General Terms

Reliability, Security.

Keywords

Secure mutlicast, multihop ad hoc, MANET, tracking.

1. INTRODUCTION

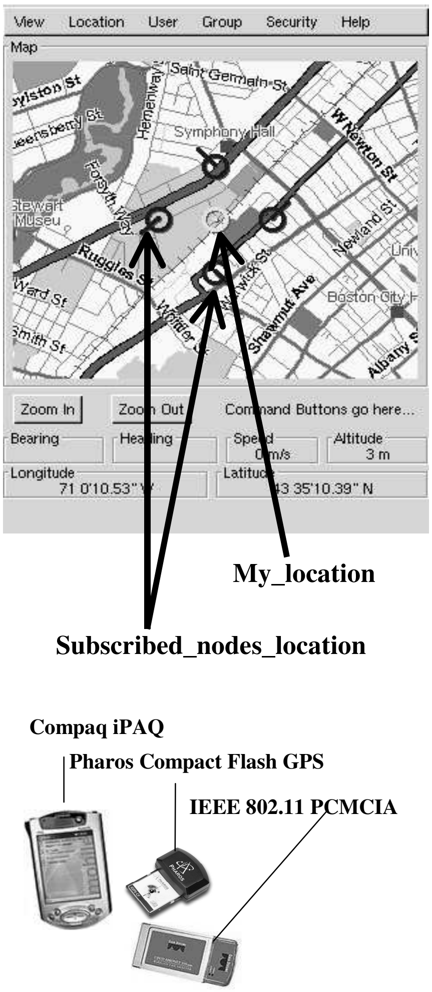
One important research problem for secure dissemination and sharing of information over wireless multihop ad hoc networks (MANET) is how to restrict the information access to the group of authorized nodes. The data information has to be encrypted and only authorized users should be able to decrypt it. The security of the group has to be maintained when new members join/leave or when a node is revoked. The problem can be defined as follows: given one source multicasting a stream of data and multiple receivers that join and leave the multicast session, the goal is to design a low bandwidth/delay protocol that allows authorized nodes and only authorized nodes to access the data stream multicast by the source node. The underlying communication network is a multihop wireless ad hoc network with mobile nodes. Therefore, the secure multicast group management protocol has to take into account unreliable links, nodes mobility and limited communication and computation power of the nodes.

The protocols were implemented in a demonstration application[1] aiming at secure monitoring and tracking of mobile nodes interconnected by a MANET. The prototyping testbed is composed of a set of iPAQ PDAs and laptops, equipped with a wireless interface (IEEE802.11) and location acquisition interface (Compact Flash GPS). The nodes are running the Linux operating system. Figure 1 shows the graphical user interface of secure monitoring and tracking of nodes. The underlying unicast routing protocol is Dynamic Source Routing (DSR).

In this paper, we focus on the secure multicast protocols. We do not address the problem of multicast routing of data information. The multicast data can take the same path as the security traffic or a different path. In the latter case the security traffic carries the key to decrypt the data packets. The authorized nodes are given a *service right* certificate that allows them to prove to other authorized nodes that they are authorized to access the multicast data stream. Another aspect that is not addressed by this paper is denial of service. Several mechanisms have to be incorporated in the proposed protocols to prevent malicious nodes from denying joining nodes to attach to the multicast tree or to exhaust their energy resources.

In Section 2, we present the related research work. In Section 3, we describe our approach, and the proposed protocols. In Section 4, we describe the security services and mechanisms provided such as authentication, data integrity, and nodes revocation. Section 5, summarizes the performance results of the proposed protocols both in terms of communication cost and delay. Finally, Section 6 provides optimization techniques to enhance the reliability of the protocols.

[1] Work supported by Draper Lab IR&D projects under contract #523120.



Speed 0 m/s
Altitude 3 m
Longitude 71 0'10.53" W
Latitude 43 35'10.39" N

Figure 1. GUI of the application and components of the MANET testbed nodes.

We first describe the main characteristics of multihop wireless ad hoc networks that will impact the design of a secure multicast protocol.

- **Mobility:** routes between nodes change with time resulting in higher packet loss and necessity of frequent discovery of paths. Long paths have a higher probability of breaking and consume more resources.
- **Wireless:** links are unreliable. The loss of a packet that contains information to update a group key will prevent the node from updating the group key. This has the same effect as excluding this node from the group. In some existing schemes this would result in requiring the node to send a new join request in order to be able to decrypt subsequent group key information.
- **Multihop:** the communication cost of multicast depends on the number of hops. The communication cost is defined as the average total number of packets to be transported by the network to allow a user to join or leave the group. Therefore a packet traveling over two hops counts twice. Most existing protocols only evaluate the communication cost at the source. In a MANET environment the cost of a join request and group key update has to take into account all hops of communication. This is an especially important constraint because of the scarcity of radio resources.
- **Ad hoc:** nodes have limited computation power. One implication is that a single node might be unable to manage the keys of a large group. On the other hand nodes can play an active role in the group security therefore reducing the computation load and memory cost at the source.

2. RELATED WORK

Previous research in the area of secure multicast has mainly focused on wired networks and various techniques were proposed considering various constraints [1-14]. The main limitation of these algorithms is that they were not designed for multihop wireless ad hoc networks. The most known technique is the construction of a logical key tree where group members are associated with leaves and each member is given all the keys from his leave to the root. The root key is the group key. This approach allows reducing the communication cost for key update, on the event of group membership change, to $O(\log M)$ where M is the number of group members. Various extensions were proposed to deal with reliability [15], node dependent group dynamic [16], and time variant group dynamic [4, 11]. Extensions to wireless networks were first discussed in [17] and several secure multicast protocols were proposed [18-20]. These protocols addressed both issues related to mobility and unreliability. However, these protocols have mainly focused on single hop wireless networks where base stations or satellite beams cover large areas. Very recently secure multicast in multihop wireless ad hoc networks was investigated in [21]. It was shown that significant energy saving can be achieved for secure multicast over ad hoc networks by placing the nodes on the key tree according to their physical location. The proposed heuristics addressed the case of dynamic groups where the nodes are *non-mobile* or with very low mobility.

The area of securing ad hoc and sensor networks [22, 23] gained lot of interest in the last few years and several protocols and techniques were proposed for key pre-distribution to allow secured connectivity [24, 25], protection against denial of service [26-28], enforcing fairness [29], authentication and integrity of data streams [30-32]. Several of these techniques are complementary to our work in securing the ad hoc network.

CONTRIBUTIONS:

In this paper, we consider the problem of secure multicast over a multihop ad hoc network. The nodes are assumed to have a reasonable computation capability such as some of the recent PDAs (e.g., iPAQ H3800 and H3900). Our main focus is on reducing the communication cost. We propose a set of protocols that use locality to reduce the communication complexity of secure multicast for dynamic groups of mobile multihop nodes. We aim at reducing the overall network communication cost using an anycast type of group join. Nodes attach to the multicast group through the closest neighbor already within the group. The join requests are broadcast within a limited range (using a TTL bound) to reach any group member. This reduces the cost of broadcasting join requests in the communication limited ad hoc network. It allows using short paths for key update, and finally prevents group membership change from impacting the whole group. The protocol prevents unauthorized nodes from accessing the multicast data and allows fast revocation of nodes.

3. THE PROPOSED APPROACH

In this section we describe our approach, the proposed protocol, and the application and network architecture.

3.1 Assumptions and security requirements

- All nodes have similar minimum computation and communication capabilities. In our testbed we are using 400MHz iPAQ PDAs and laptops. The most computation demanding public key operation being a digital signature. A 1024 bit RSA digital signature requires less than 10ms on a 750 MHz Pentium III laptop when using the cryptlib library [33]. Therefore a reasonable number of public key operations can be carried out.
- Only the nodes with a valid service right certificate should be able to access the data.
- Nodes should not be able to receive any data after the revocation of their certificate.
- The data integrity should be maintained (i.e., protection against message fabrication, unauthorized modification, and replay).

3.2 Physical security group tree and group discovery

Our basic scheme for secure multicasting is based on maintaining a physical security tree of the group members. This is similar to the Iolus approach [6] and the physical tree approach in [5]. However, in our approach we take into account MANET constraints. Joining members dynamically discover and attach to the "best-closest" tree node. The best-closest node is defined according to the load at that node, path to the joining node, and path to the source. In our current implementation we only consider the closest group member in terms of number of hops. Since, only members already accessing the information can act as intermediaries, then the clear data (or group key) is not accessed by any third party. Furthermore, since *all* group members can act as intermediaries the communication complexity of the protocol is lower than schemes that require the joining node to attach to the source or to a limited number of group controllers. The security multicast tree is used to securely forward the group key to authorized members. Leaving members inform downstream nodes that they will be soon leaving the group and request them to attach to another node. Whenever a node leaves the group, its upstream node stops sending the data to it. Nodes refresh their participation in the group through periodic messages to their upstream nodes.

3.3 Application and network architecture

The multicast group security is implemented within the secure group sessions layer. Each session has a manager that is responsible for authenticating joining members, checking service right certificates, maintaining information about attached group members, forwarding the multicast data and processing revocation requests.

3.4 Protocol description

In this Section we describe the join and leave process. The revocation mechanism is described in Section 4.3. The group join has three main steps: broadcast group join request, process group join replies, mutually authenticate and establish a link key with an upstream node that is already within the group. The remaining steps are for optimization and tree maintenance purpose.

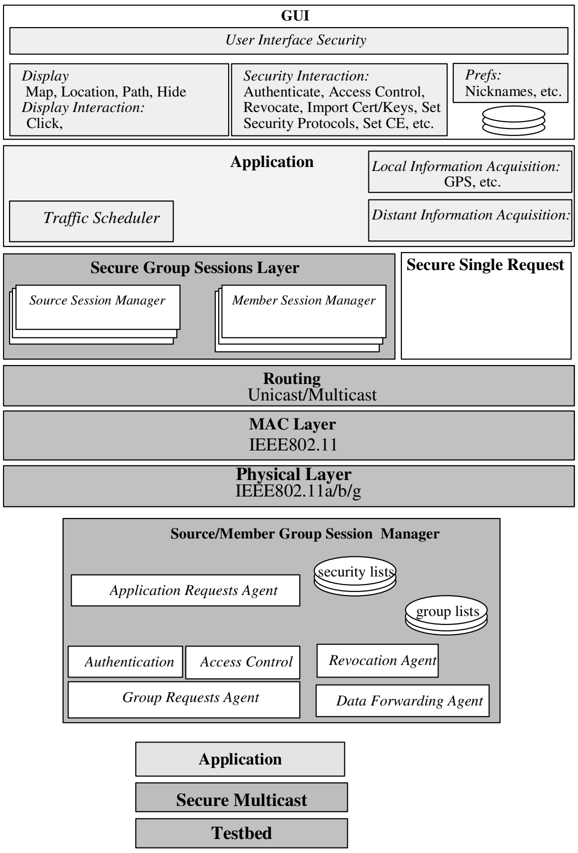


Figure 2. Application and multicast protocol architecture.

Joining steps:

- **Broadcast Group Join Request:** the parameters of this request are *group_id*, and *TTL*. Depending on at which protocol layer this request is processed, the *group_id* could be a tuple uniquely describing the source, and the information service the user intends to subscribe to, or it could be an IP multicast address and port number. The *TTL* parameters can be first set to a small value and then exponentially increased until when a satisfactory number of replies are received. Each request has a sequence number *SN* to avoid multiple forwarding of the same request.
- **Receive Group Join Replies:** Nodes that are already receiving this information service send replies to the requester. Unless if the number of connected nodes to them exceeds some threshold, or if this requesting node is on their path to the source. The replies contain information about: number of hops to the source, logical path to the source (the sequence of group members that

lead to the source will be used in a handover to avoid loops), path quality to the source, and number of nodes already connected to this node. Upon receipt of at least one satisfactory reply, the requesting node will initiate a registration with the sender of the most satisfactory reply. A simple criterion to accept a reply as satisfactory is if the aggregate quality of the path from the joining node to the intermediate node and path from the intermediate node to the source exceeds some threshold, and if the number of already connected nodes is below some threshold. In our basic protocol implementation the criterion is the number of hops to the intermediate node.

- **Authentication, registration, and key establishment:** the requester and intermediate node will first *mutually* authenticate each other. The authentication process will lead to the establishment of a shared key. Then, they will both check that they are allowed to access this information. This proof of access right is done using a service-right certificate. The authentication is complete only when the certificates are verified not to be revoked.
- **Tree optimization:** once registered the newly joining node can send a *path_optimization* message to nodes that are already in the tree but could optimize their path by attaching to the joining node. The joining node knows the nodes that can benefit from the optimization by using the information gathered from the request replies.
- **Receive encrypted data:** after authentication and verification of the revocation information, the joining node can start receiving the encrypted data. The data encrypted using the secret key established during the authentication phase.
- **Receive tree update events:** three types of tree update events can be received: *Group Join Requests*, *Handover Requests*, and *Path Quality Drop*. The *Group Join Requests* will be processed as explained above. The *handover request* will initiate a search for a better intermediate node. This search process can be done in a proactive way. When a node receives a handover request it will delay replying to incoming join requests. The drop in path quality will also initiate a search for a better intermediary node.

Leaving steps:

- **Inform downstream nodes.** The depending downstream nodes should initiate a handover and send a handover complete message once reconnected to the tree.
- **Inform upstream node.** Once all downstream nodes are reconnected or after a *timeout*, the leaving node requests its upstream node to disconnect him from the tree.

3.5 Operation Modes

We identify three modes of operation depending on the separation between data and control traffic routing and also on how the data encryption is changed.

- The data and control traffic travel together. This mode is used when the amount of data traffic is relatively small.
- The data multicast tree is different from the security multicast tree and the group key is changed at each group membership change. This requires the source to be updated at each group membership change. This information can be piggybacked and merged with the *IsAliveAck* messages periodically sent by the downstream nodes to the upstream nodes.
- The data multicast tree is different from the security multicast tree. The group key is periodically changed. For example every packet is encrypted separately or every period of time the key is changed. The data encryption key travels along the security multicast tree. If more than one packet are sent between periodic updates this implies a vulnerability period where a leaving node is still receiving the group data. This vulnerability period can be controlled by the source depending on the sensitivity of the multicast data.

Our application multicasts short packets containing location and sensed information, therefore we operate in the first mode. The secure multicast protocol can operate on top of any unicast ad hoc network routing protocol such as DSR or AODV [34]. Such protocols allow maintaining the shortest path between a node and its upstream group member. As an underlying routing protocol our testbed uses a linux implementation of DSR that was also ported to linux iPAQ platforms.

4. Security Services and Mechanisms

In this section we describe the security services and mechanisms: authentication to join the group, revocation mechanism, and integrity protection of multicast data. We first start by describing the service right certificates that allow joining nodes to prove their right to become part of the group. Then we discuss possibilities for the revocation mechanism and detail the revocation protocol when initiated by the multicast group source.

4.1 Service right certificates

In a MANET setting where information is disseminated over the network, it is necessary to protect the access to services in a distributed manner. In the case of a multicast session, joining nodes have to prove that they are authorized to access the multicast session before being accepted in the group. The access control mechanism is distributed among the group members. A *"service right certificate"* allows a user/node to prove that it is authorized to access some service. Informally such a certificate is a message describing the service and signed by an authorized entity. The format of such certificates is as follows:

[DataId | Issuer | TypeOfService | ValidityPeriod | RevocationSequenceNumber | UserPublicKey | Signature].

- The *DataId* field is used to uniquely identify the data/session to be accessed. This could be a tuple (*source-id*, *session-id*), where *source-id* is the source of the data and *session-id* is an identifier that is unique within the source context. The data can be either a stream of information or some non-time varying information disseminated over the network.

- The *issuer* field uniquely identifies the entity that is signing this certificate.
- The *TypeOfService* defines the type of operations/requests that are allowed for this user. That could be the frequency update for a node location. However, ensuring the security of these operations is much more complex than a simple read access. For now only a read operation is considered. Another type of service certificates is the one that grants the right to a user to generate certificates on behalf of the source node.
- The *ValidityPeriod* is a time interval where this certificate is valid: [*NotValidBeforeDate*, *NotValidAfterDate*].
- The *RevocationSequenceNumber* field is used to make the revocation efficient. Whenever a new revocation list is issued, this sequence number is incremented. Therefore if the latest sequence number of the issuer is equal to the certificate sequence number, there is no need to check the revocation list.
- The *UserPublicKey* field provides the public key of the user.
- All the above information is signed, by the issuer, in the *Signature* field.

Additional information can be included in the certificate such as the signature algorithm identification, algorithm parameters, and certificates serial number. Service certificates can be used in chains as traditional certificates of the X.509 PKI framework.

Multilevel certificate can also be used to certify that a user has access to all the data of a specified source, or from any source at some level in an access right hierarchy. More work has to be done on syntax, semantic, and algorithmic aspects of multilevel security.

The certificates can be provided to the group members, offline (before a mission) or through an out-of-band communication with certificates issuing authorities. In the latter case a node would request a new certificate and mutually authenticate itself with the certification authority. The certification authority will first check its access control lists and policies before generating the service certificate.

4.2 Authentication

We use a classical public key authentication protocol based on certificates [35]. The purpose here is not to authenticate the identity but the right to access the multicast data. At the end of the authentication the already group member verifies that the joining node is not in the revocation list (See Section 4.3). The joining node also obtains a fresh revocation list to prevent malicious nodes from misleading joining nodes.

4.3 Revocation Process

Several *revocation models* can be investigated. The revocation can be initiated by the service certifier (e.g., information source or CA), or by a third party (e.g., one or k other nodes that could belong to a revocation hierarchy). A *revocation hierarchy* can be used to provide rights for third parties to revoke certificates. The revocation process has implications on service availability and robustness to denial of service. We focus on the basic scenario where the revocation is initiated by the source and is always reliably propagated through the multicast source. The *revocation information* travels with the data and its integrity has to be protected.

Denial of service is an important problem when dealing with revocation. If revocation information cannot be reliably delivered to the node, because of an adversary, then legitimate nodes will not be able to verify the service right certificates, which would prevent the multicast group from correctly functioning. This might lead to forcing joining nodes to directly communicate and attach to the source. The integration of revocation information with data transmission and the requirement of reliability of delivery forces the attacker to carry a full communication denial of service attack to prevent the revocation process.

Revocation of nodes is a rare event however it is very important in insuring the security of the system. Since it is a rare event we can assume that we can afford reasonably more computation and bandwidth resources to deal with it. In this section, we describe the revocation initiated by the source.

The revocation process from the source is as follows:

1. Periodically multicast a certificate revocation list (CRL). The list is reliably multicast by requiring all downstream nodes to acknowledge its receipt. The list format is: **[MinSN | CurrentSN | ListOfRevokedCert | Timestamp | Signature]**. *ListOfRevokedCert* contains all the certificates that were revoked since *MinSN*. Therefore all certificates with a *SN* in **[MinSN, CurrentSN]** interval and not in the revocation list are valid. A certificate with a *SN* lower than *MinSN* will have to be verified by the source unless if the intermediate node cached previous revocation lists. In order to make the revocation list more compact only the hashes of the certificates have to be included in the list.
2. Whenever, a certificate is revoked by the source, the *CurrentSN* is incremented and a new revocation list is issued. The source can decide to increment the *MinSN* if the revocation list is getting too long in order to maintain a fixed size revocation message. The source then sends the new CRL to its downstream nodes and delays sending the data to them until it gets an acknowledgement for the CRL.

The members of the secure multicast tree process the certificate revocation list as follows:

1. If the revocation list contains one of their downstream nodes, they will stop forwarding the multicast data to it.
2. They will append the new list to their previous list and securely store it.
3. They will forward the new CRL to their downstream nodes and delay forwarding data to them until when an acknowledgement is received.
4. If a group member does not receive a fresh CRL after the transmission period even after requesting it from its upstream node, the group join phase is restarted.

Revocation checking at Join time:

When a node *A* requests to join the group it will send its service right certificate to other group members. Assume that *A* is attaching to node *B*. If *A's* certificate has a *SN* that is within [*MinSN*, *CurrentSN*] of the CRL stored at *B* then *A* will be approved to join, if it is not in the list of revoked certificates. If the *SN* of the certificate of *A* is lower than *MinSN* in *B*'s CRL then *A* needs to get the approval from the source or a certificate with a higher *SN*. *B* will also obtain a fresh CRL from *A* and will verify that *A* is not revoked. The freshness of the CRL is verified through the timestamp. This assumes a loose time synchrony between the source and the joining node. Further improvement on the revocation process might be obtained using techniques similar to [36].

4.4 Data integrity

Data integrity can be provided in different ways. The most straightforward way is to have the source sign a hash of all the previous messages every period *T: MAC* = Signature(Hash(t | M_i | M_{i+1} |…| M_{i+N}), where M_js are the packets transmitted during the last *T* units of time and *t* is the current time and is included to prevent replays. Since the time here is used as a counter only a minimum form of time synchronization is required. This simple technique amortizes the cost of public key signatures. However, it has two disadvantages. The data stream is only authenticated after *T* units of time and it still requires public key encryptions. Although this approach is reasonable for our testbed, we are investigating the use of the technique proposed in [31, 32] for authenticating multicast streams by the only use of symmetric keys. This requires each receiver to know an upper bound on the difference between his clock and the sender's clock. If the GPS time source is available (as it is in our case) and trusted then the TESLA approach can be easily implemented. If the receiver can directly communicate with the source then an upper bound on clocks difference can be computed by the receiver as proposed in [32]. However, this has to be an infrequent operation to avoid violating the locality concept.

5. Communication cost and delay performance

The rationale behind the design of our protocol was to take into account MANET constraints to increase reliability and reduce communication cost and delay. The join/leave/revocation operations only require local updates (i.e., with upstream/downstream nodes). Therefore their cost is bounded by the maximum degree of the physical tree. Since all nodes can serve as intermediaries the degree can be kept very small while in the basic Iolus scheme the number of intermediary group key controllers is limited and therefore their degree has to be high. In comparison with the key graphs approach at each group change a multicast packet of size *O*(log *group-size*) has to be sent to all nodes while in our approach only local information has to be exchanged. Also in the key graph technique whenever a node does not receive a key-update packet, it has to rejoin the group since it will not be able to decrypt subsequent key-update packets. Finally, the computation complexity is balanced over all the nodes of the group and is not concentrated in one or few group controllers.

As a first step towards evaluating our approach and comparing it to prior research in the area, we simulated the basic key management protocol called *Ad hoc Group Key* (AGK) in the ns-2 network simulator (http://www.isi.edu/nsnam/ns/) using the Rice Monarch wireless extensions (http://www.monarch.cs.rice.edu/). In this initial simulation the protocol does not have the tree optimization and handover techniques implemented. We simulated AGK on randomly generated networks, with a total number of 120 nodes. The grid size is 1200 x 1200 m^2. The radios have a data rate of 2Mbps and 250 meters nominal range. We simulated AGK in both static and mobile networks. The motion of the nodes follows the random waypoint model [34], with maximum speed 20m/s and pause time 20 or 40 seconds.

To study the effects of node density, we allow *N* (20, 40, and 60) nodes to be able to join/leave the multicast group. This allows to estimate the performance of the protocol under reasonable (~10%) to high density (50%). The group dynamic is introduced into the simulation, by specifying the average time a node stays within the group as well as the average time it stays outside. For our simulation, we use three pairs of average time {(J5, L3), (J10, L6), (J20, L12)}, where J5 indicates that the average time duration within the group is 5 seconds and L3 indicates that the average time duration outside the group is 3 seconds. These values correspond to relatively highly dynamic groups. Another source of dynamic behavior comes from the nodes mobility. All simulation results are averages of 3 (or 5) runs on different randomly chosen scenarios. In the following we present our preliminary simulation results. Based on them we discuss the join delay and communication cost versus group density, mobility and dynamic.

Figure 3, indicates that the average delay for a group with potentially 40 group members is slightly but not significantly lower than the delay for a group of 20 members. This slight reduction in delay can be explained by the fact that the joining members can find a closely located group member to attach to. However, in the case of 60 nodes the delay increases. Further analysis reveals that it is because the packets drop rate significantly increased due to the high traffic volume. Our conclusion is that the delay is not affected but the nodes density if the traffic does not exceed the network capacity. The delay also increases with mobility but stays within a reasonable range.

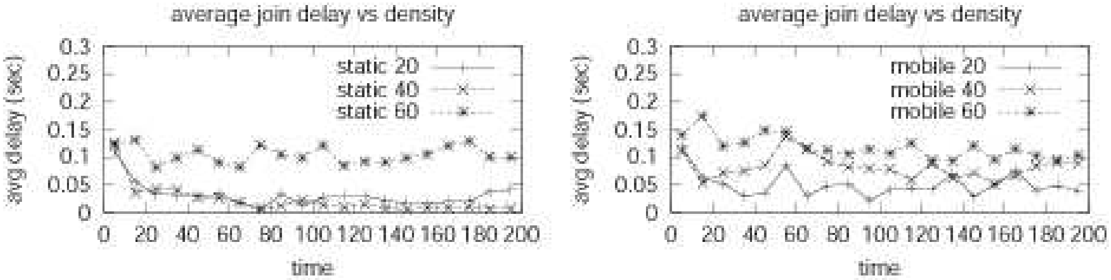


Figure 3. Average join delay for a static and mobile network with potential group members of 20, 40, and 60 and average join/leave duration of 10/6 seconds.

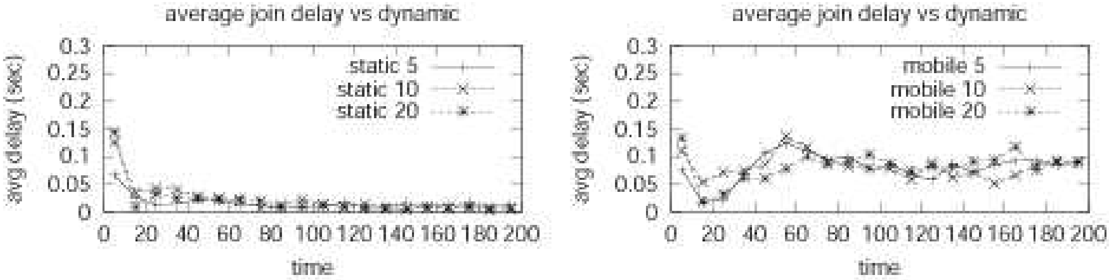


Figure 4. Average join delay for a static and mobile network with potentially 40 members average join/leave duration of 5/3, 10/6 and 20/12 seconds.

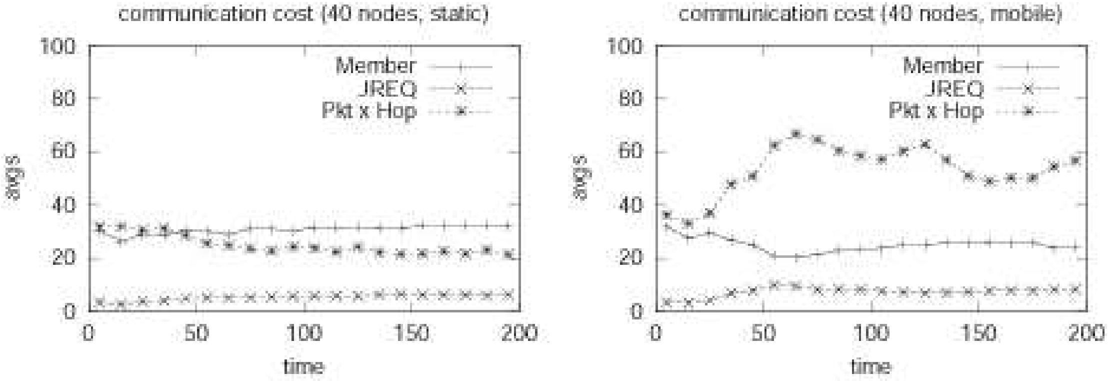


Figure 5. Communication cost in a non-mobile and mobile network 40 potential members with an average join duration of 10 and average leave duration of 6 seconds. The speed for the mobile network is 20 m/s and the pause time 20s. The displayed curves indicate the averages of number of members in the group, number of *JoinRequest* packets and the total number of packets counted on all hops.

In Figure 4, the delay curves for the three simulated group dynamics 5, 10, and 20 are very similar. This holds for both the static case and the mobile case. This indicates that the group dynamic does not impact the join delay. However, we observe that the delay in the mobile network is higher.

Figure 5, shows the average number of members in the group, the average number of Join-Request packets, and total number of control packets multiplied by the number of hops traveled. The *PktxHop* measures the communication cost of the protocol. In a mobile network, the communication is higher than in a static network. The increase in cost can partially be explained by the fact that we did not simulate to tree optimization technique. The slight increase in *JoinRequest*s can be explained by the higher packet loss rate due to mobility.

These are preliminary simulation results that have to be extended by implementing the proposed optimization techniques, further varying the group dynamic, and comparing the performance of the proposed protocol to schemes where the nodes join at the source.

6. Enhanced Reliability Techniques

6.1 Multi-link group attachment

Although attaching to closer neighbors can reduce inherent unreliability of a MANET we would like to further increase the reliability of the multicast tree. The basic idea is to maintain more than one upstream node. This makes sense if the goal is to increase the reliability of some sensitive traffic. For example if this traffic is lost, then the data traffic will be unrecoverable for a long period of time. In the case of secure multicast if no redundancy coding is used (see Section 6.2) maintaining more than one link has a high cost. Therefore it will only be considered if the data and security traffic are separated. The multi-link attachment is a variation of the basic physical tree protocol. In this case the joining node authenticates and attaches to n upstream nodes.

6.2 K-out-of-N coding approach for link resiliency

This algorithm addresses two issues: efficient reliability and increased security. The reliability is improved by allowing a node to maintain more than one link but at a lower communication cost. The security is increased by requiring a joining node to authenticate with at least k members of the group.

Example: $k = 2$, $N = 3$. The simplest scheme can consist of each group member connecting to $N = 3$ other members of the group. Each new member only needs to receive two messages out of three.

- **Data traffic:** the data is multicast directly to the group members or through the security multicast tree.
- **Security traffic:** the first solution is to have the group key changed by the source for each packet. This is a reasonable assumption if the size of the packet is much larger than the size of the key. The second solution is to change the key by the source at each group membership change. In this case the source has to be informed of membership change. The cost is still reasonable because this information is may be already sent to allow the source to keep track of who is in the group. Also, the membership change only requires a multicast packet and not $O(N)$ packets (where N is the size of the group).
- **Transmitted key information:** the group key K can be partitioned into k (= 2) portions. Here $K = K_0K_1$, and if the size of the key is 256 bits, each portion will have 128 bits. The joining member can request K_1 from his first link, K_2 from his second link and $K_0 \oplus K_1$ from the third link. Another solution is to have each node send to its dependents $K_0 + Id*K_1$ (where the addition and multiplication are computed over a suitable finite field, and Id is the unique identification number of the sending node). With both solutions the receiver can recover the key K using the information sent by two upstream links. In addition a node cannot recover the key if it is *only* connected/authenticated to one group member. The excess computation cost of the second solution can be balanced by the fact that a node always sends one message. Furthermore, the second solution can be easily extended to larger values of k and N.

General case: each node sends $K_1 + Id*K_2 + \ldots + Id^{k-1}*K_{k-1}$. Any k correctly received messages are used to recover the key K using a Lagrange interpolation. Further resiliency to malicious nodes, that send wrong portions, can be achieved by using error-correction techniques such as the Berlekamp-Welch algorithm [37].

7. Conclusions and Future Research

In this paper we have proposed a secure multicast group management protocol that addresses some issues specific to ad hoc networks (namely communication cost, mobility, and link unreliability). Nodes attach to the best closest neighbor already in the group therefore reducing the cost of join requests broadcast and reducing the communication and computation cost incurred by the source. Moreover, using shorter paths for key update increases the reliability of the secure multicast. Preliminary simulation results show that the protocol scales well with various mobility and group dynamic scenarios. However, further simulation is needed for better assessment of the protocol performance. For future research, we plan to investigate the computation efficiency issues and robustness to denial of service attacks. We also plan to investigate scenarios of heterogeneous nodes where some nodes do not have the computation power to execute public key operations.

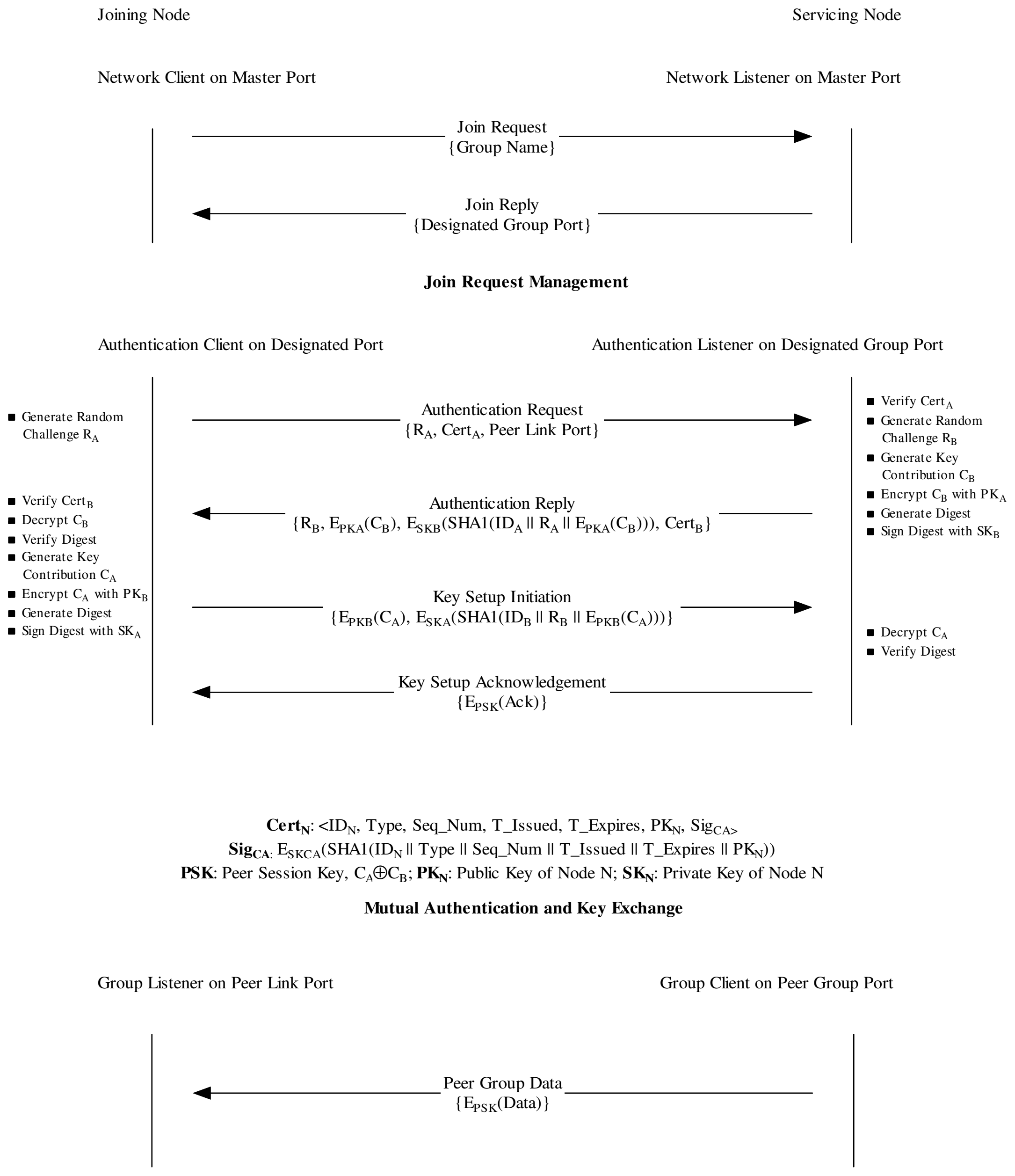
8. References

1. C.K. Wong, M. Gouda, and S. Lam. "*Secure group communications using key graphs*". in *Proceedings of ACM SIGCOMM*. 1998.
2. Guevara Noubir, "*Optimizing Multicast Security over Satellite Links*". 1998, European Space Agency.
3. Guevara Noubir and L.v. Allmen. "*Security Issues in Internet Protocols over Satellite Links*". in *Proceedings of IEEE Vehicular Technology Conference (VTC'99 Fall)*. 1999. Amsterdam, Holland.
4. Guevara Noubir, Feng Zhu, and A.H. Chan. "*Key Management for Simultaneous Join/Leave in Secure Multicast*". in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*. 2002.
5. Refik Molva and A. Pannetrat, "*Scalable Multicast Security with Dynamic Recipient Groups*". ACM Transactions on Information and System Security, 2000. **3**.
6. Suvo Mittra. "*Iolus: A Framework for Scalable Secure Multicasting*". in *Proceedings of ACM SIGCOMM '97*. 1997. Cannes, France.
7. Ran Canetti, et al. "*Multicast Security: A Taxonomy and Some Efficient Constructions*". in *Proceedings of INFOCOMM*. 1999: IEEE Press.
8. A. Perrig, D. Song, and D. Tygar. "*ELK, a new protocol for efficient large-group key distribution*". in *Proceedings of IEEE Security and Privacy Symposium*. 2001.
9. D. Balenson, D. McGrew, and A. Sherman, "*Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization*". 1999, Internet Draft.
10. D. M. Waller, E. C. Harder, and R.C. Agee, "*Key Management for Multicast: Issues and Architectures*". 1998, Internet Draft.
11. F. Zhu, A. H. Chan, and G. Noubir. "*Optimal Tree Structure for Key Management of Simultaneous*

Join/Leave in Secure Multicast". in *Proceedings of MILCOM*. 2003. Boston, MA, USA.

12. Guevara Noubir. "*A Scalable Key Distribution Scheme for Dynamic Multicast Groups*". in *Proceedings of Third European Research Seminar on Advances in Distributed Systems*. 1999. Madeira Island, Portugal.
13. S. Setia, S. Koussih, and S. Jahodia. "*Kronos: A Scalable Group Re-Keying Approach for Secure Multicast*". in *Proceedings of IEEE Security and Privacy Symposium*. 2000. Oakland, CA, USA.
14. Adrian Perrig and D. Tygar, "*Secure Broadcast Communication in wired and wireless networks*". 2002: Kluwer.
15. Y. Yang, X. Li, and S. Lam. "*Reliable Group Rekeying: Design and Performance Analysis*". in *Proceedings of ACM SIGCOMM*. 2001. San Diego, CA, USA.
16. R. Poovendran and J.S. Baras. "*An Information Theoretic Analysis of Rooted-Tree Based Secure Multicast Key Distribution Schemes*". in *Proceedings of Advances in Cryptology CRYPTO'99*. 1999.
17. L. Gong and N. Sacham, "*Multicast Security and its Extension to a mobile Environment*". Wireless Networks, 1995. **1**(3): p. 281-295.
18. Danilo Bruschi and E. Rosti, "*Secure Multicast in Wireless Networks of Mobile Hosts: Protocols and Issues*". Mobile Networks and Applications, 2002. **7**: p. 503-511.
19. C. Zhang, et al. "*Comparison of Inter-Area Rekeying Algorithms for Secure Wireless Group Communications*". in *Proceedings of Performance 2002*. 2002. Rome, Italy.
20. Thomas Kostas, et al. "*Key Management for Secure Multicast Group Communication in Mobile Networks*". in *Proceedings of DARPA Information Survivability Conference and Exposition*. 2003.
21. Loukas Lazos and R. Poovendran. "*Energy-Aware Secure Multicast Communication in Ad-hoc Networks Using Geographic Location Information*". in *Proceedings of IEEE International Conference on Acoustics Speech and Signal Processing*. 2003. Hong Kong, China.
22. Frank Stajano and R. Anderson. "*The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks.*" in *Proceedings of Security Protocols, 7th International Workshop*. 1999: Lecture Notes in Computer Science, Springer Verlag.
23. L. Buttyan and J.P. Hubaux, "*Report on a Working Session on Security in Wireless Ad Hoc Networks*". ACM Mobile Computing and Communications Review (MC2R), October 2002.
24. Haowen Chan, Adrian Perrig, and D. Song. "*Random Key Predistribution Schemes for Sensor Networks*". in *Proceedings of IEEE Symposium on Security and Privacy*. 2003.
25. L. Eschenauer and V. Gligor. "*A key-management scheme for distributed sensor networks*". in *Proceedings of 9th ACM Conference on Computer and Communications Security*. 2002. Washington D.C., USA.
26. Yih-Chun Hu, Adrian Perrig, and D.B. Johnson. "*Efficient Security Mechanisms for Routing Protocols*". in *Proceedings of Network and Distributed System Security Symposium*. 2003.
27. B. Dahill, et al., "*ARAN: A secure Routing Protocol for Ad Hoc Networks*". 2002, UMASS Amherst.
28. P. Papadimitratos and Z. Haas. "*Secure Routing for Mobile Ad Hoc Networks*". in *Proceedings of CNDS*. 2002.
29. L. Buttyan and J.P. Hubaux, "*Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks*". ACM/Kluwer Mobile Networks and Applications (MONET), 2003. **8**(5).
30. Adrian Perrig, et al., "*SPINS: Security Protocols for Sensor Networks*". Wireless Networks. **8**(5): p. 521-534.
31. A. Perrig, et al. "*Efficient authentication and signing of multicast streams over lossy channels*". in *Proceedings of IEEE Symposium on Security and Privacy*. 2000.
32. Adrian Perrig, et al. "*Efficient and Secure Source Authentication for Multicast*". in *Proceedings of Network and Distributed System Security Symposium*. Feb. 2001.
33. "*Cryptlib*". http://www.cryptlib.orion.co.nz/.
34. J. Broch, et al. "*Performance Comparison of Multi-hop Wireless Ad Hoc Network Routing Protocols*". in *Proceedings of ACM MobiCom*. 1998: ACM Press.
35. Alfred J. Menezes, Paul C. van Oorschot, and S.A. Vanstone, "*Handbook of Applied Cryptography*". October 1996: CRC Press.
36. David Cooper. "*A More Efficient Use of Delta-CRLs*". in *Proceedings of IEEE Symposium on Security and Privacy*. 2000.
37. Lloyd R. Welch and E.R. Berlekamp, "*Error correction of algebraic block codes.*" US Patent, 4,633,470, 1986.

9. Appendix


Joining Node
Servicing Node
Network Client on Master Port
Network Listener on Master Port
Join Request
{Group Name}
Join Reply
{Designated Group Port}
Authentication Client on Designated Port
Authentication Listener on Designated Group Port
Generate Random Challenge R_A
Authentication Request
$\{R_A$, $Cert_A$, Peer Link Port$\}$
Verify $Cert_A$
Generate Random Challenge R_B
Generate Key Contribution C_B
Encrypt C_B with PK_A
Generate Digest
Sign Digest with SK_B
Verify $Cert_B$
Decrypt C_B
Verify Digest
Generate Key Contribution C_A
Encrypt C_A with PK_B
Generate Digest
Sign Digest with SK_A
Authentication Reply
$\{R_B, E_{PKA}(C_B), E_{SKB}(SHA1(ID_A \parallel R_A \parallel E_{PKA}(C_B))), Cert_B\}$
Key Setup Initiation
$\{E_{PKB}(C_A), E_{SKA}(SHA1(ID_B \parallel R_B \parallel E_{PKB}(C_A)))\}$
Decrypt C_A
Verify Digest
Key Setup Acknowledgement
$\{E_{PSK}(Ack)\}$
Group Listener on Peer Link Port
Group Client on Peer Group Port
Peer Group Data
$\{E_{PSK}(Data)\}$

Join Request Management

$Cert_N$: $<ID_N$, Type, Seq_Num, T_Issued, T_Expires, PK_N, $Sig_{CA>}$

Sig_{CA}: $E_{SKCA}(SHA1(ID_N \parallel Type \parallel Seq\_Num \parallel T\_Issued \parallel T\_Expires \parallel PK_N))$

PSK: Peer Session Key, $C_A \oplus C_B$; **PK_N**: Public Key of Node N; **SK_N**: Private Key of Node N

Mutual Authentication and Key Exchange

Group Data Transfer

Figure 6. Authetication Protocol.