Robust Membership Management for Ad-hoc Groups

Silja Mäki, Tuomas Aura, Maarit Hietalahti
Helsinki University of Technology
Laboratory for Theoretical Computer Science
FIN-02015 HUT, Finland
{Silja.Maki, Tuomas.Aura, Maarit.Hietalahti}@hut.fi

Abstract

In ad-hoc networks, the network nodes or users often form peer groups. The members of a group may share an application, a physical location, or administrative tasks. Defining who is a member of the group is also the first step towards establishing a shared secret key for secure communications. Group membership management involves adding and removing nodes in the group, as well as a method for authenticating the group members. In this paper, we present a fully distributed, certificate-based system for group membership management. It is designed to suit highly dynamic ad-hoc networks where communications is sporadic and nodes often fail unexpectedly.

1 Introduction

Ad-hoc networks are constructed and maintained without the help of permanent administration and preexisting infrastructure. They are often wireless and mobile. The size of ad-hoc networks may vary from a few connected nodes to hundreds or thousands of nodes. Some examples of ad-hoc networks are networks of personal, household and office appliances, and mobile radio networks for rescue, law-enforcement and military operations.

The security of most traditional networks relies on the existence of a permanent, specialized network administration that defines the security policy and provides the infrastructure for implementing it. In ad-hoc networks, all the services, including a security infrastructure, must be created from scratch and administered by the nodes that decide to form the network.

A basic concept in many ad-hoc networks is a *group* of users or network nodes. A group is a set of entities that want to communicate with each other and to co-operate

for some purpose. The need for forming a group could be a shared application, physical location, or administrative tasks that associate some nodes with each other. Forming a group may also be the first step towards creating a shared secret, which will be used to separate the insiders from the outsiders.

Membership management of groups involves adding and removing nodes in the group, as well as providing a method to authenticate the group members. A node may prove its membership in a group also to nodes that are not members of the group. The most important security functions for a group are the following:

- mutual authentication of group members
- authentication of group members to outsiders
- authentication of members with special roles in the membership management
- mutual authentication for group key-exchange.

Ad-hoc groups need cope with with the special characteristics of ad-hoc networks. The connections are unstable, and energy resources of the nodes are scarce. Ad-hoc networks generally lack infrastructure and are vulnerable to security threats. As a consequence, the groups should be managed as lightly as possible, and independently from fixed network services and from each other. The management should be distributed and tolerant to failures of physical security. The groups should be easy to construct and modify, in a true ad-hoc manner.

We have made the following choices in designing our protocols:

- The membership is dynamic. New members are admitted and old ones leave the network. The membership of known physically compromised nodes, or ones whose integrity is suspected, can be revoked. However, we trust most nodes to leave voluntarily and, due to the sporadic communications and unregulated relations in ad-hoc networks, provide only best-effort revocation.
- The membership protocol must be resistant to communications failures. The connections in wireless networks may fail temporarily and the network may be partitioned at times.
- The membership protocols must be resistant to the unexpected and permanent removal of nodes. Connections in an ad-hoc network may fail permanently, e.g. when a node leaves the operating area, its battery becomes exhausted or, in military applications, it is destroyed.
- Due to the frequent communication and power failures and poor physical security, all functionality is distributed to avoid single points of failure. There-

fore, the authority to admit new members and to revoke old ones is distributed to several nodes. However, we allow each authorized node to make decisions alone and do not resort to threshold schemes or other separation of powers. This is because the ad-hoc networks are often deployed in circumstances where fast decision making is more desirable than absolute security.

Our scheme for group membership management is based on public key cryptography and on the use of signed certificates. The members are represented by their public signature keys, and each group has a public signature key to represent the group as a whole. Certificates signed by the group key are used to indicate the membership of the nodes.

1.1 Certificates

Certificates are an application of public key cryptography. A certificate is a message signed with the private key of a certifier. In traditional networks, certificates signed by a trusted certification authority (CA) typically bind the identity of the key owner to the public key (*identity certificates*), or they bind some access rights to the name or to the public key of an entity. In our group management scheme, certificates signed by the group key are used to bind the status of a group member or leader to the member's public key. We assume that the public keys are distributed through secure channels such as personal contact. We make no use of identity certificates or CAs. This kind of architecture is called *key oriented.*

The authority of a certificate depends on whether the issuer is trusted or not. Certificates delegate trust: if one trusts the issuer, then one is assumed to trust the receiver of the certificate on what the certificate states. *Certification chains* involve several subsequent certificates delegating some authority from a party to another. Trust is either considered transitive, so that if the first issuer in the chain is trusted then also the last certificate in the chain is trusted, or the re-delegation of the rights may be forbidden.

A certificate may have an expiration date. After the validity period is over, the certificate loses its effect. A chain of certificates is valid only if all the certificates in the chain are valid.

The length of the validity period is a trade-off between the security improvement allowed by a periodic review of the delegation and the cost of renewing the certificates. If the private key of the receiver of the certificate is compromised, frequent refreshing prohibits long-term misuse of a certificate. On the other hand, the more often the certificates have to be re-issued, the more expensive the system is to maintain, and the more likely it is that secure communication is unavailable. In particular, the expiration of a certificate in a chain affects also the rights of all the following entities in the chain. Thus, frequent expiration may cause the system

with long certificate chains to become unreliable.

In most systems, certificates may be revoked by the issuer of the certificate, and sometimes by other authorities. Lists of the revoked certificates have to be distributed throughout the network. In a small network this may be an effective and rapid way to react against possible security failures. In a large distributed system, the time margin between the revocation and the time the information reaches a user may be so long that attacks using revoked certificates cannot be fully prevented.

Revocation is usually combined with limited validity periods for certificates so that the revocation lists or servers need only to remember those revoked certificates that have not yet been expired. Again, there is a trade-off between the size of revocation lists and how often the certificates need refreshing.

2 Related work

The nodes of an ad-hoc network are often more equal than those in a traditional communications system. This is due to the lack of permanent authorities. Consequently, security mechanisms for ad-hoc networks also treat the nodes with relative equality. While communications security has traditionally been based on point-to-point communication with trusted servers, the basis for the security of an ad-hoc system is, in many of the proposed architectures, *multicast inside a group*. For example, the ad-hoc network management protocol by Chen et al. [6] is based on secure multicast that should be received only by a given group of nodes. Unfortunately, the security features of the protocol are less mature than the network management part.

The most common security function for an ad-hoc group is to establish a secure communications channel between the members. In practice, this means creating a shared session key for the encryption and authentication of data that will be sent between the members. Efficient protocols for a group key exchange have been presented, for example, by Burmester and Desmedt [4], and by Ateniese et al. [2]. Asokan and Ginzboorg [1] discuss key exchange in ad-hoc networks with emphasis on the robustness of the protocol against the failure of some nodes. It should be noted that a shared secret key does not protect the group members from eavesdropping and impersonation by each other. The idea is that the group members trust each other with respect to the purpose of the group. End-to-end security must be used or a new group must be be established when any group member is not trusted to access some data.

Multicast is the preferred way of communicating in a radio network because it saves the bandwidth in comparison to sending separate copies of a message to individual receivers. A good overview of multicast security by Gong and Shacham

[11] also discusses the extension of Internet multicast protocols to mobile networks. However, their emphasis is on traditional mobile networks with a fixed support architecture. Secure group communication for mobile stations has been implemented in the TETRA mobile network for public safety applications [15]. It relies on manual distribution of the group key either on smartcards or over the air.

The IP [12] multicast is used mostly for the distribution of public data streams on the Internet. The number of receivers for these streams can be extremely large. Hence, the protocols are designed to minimize the load on the server. They allow anyone to subscribe to a stream, i.e., to join the multicast group, without any permission from the sender. Receivers can be selected securely by encrypting the data stream and by distributing the session key only to the authorized receivers. This lesson can be extended to all group communication: allow anyone to route and to receive the encrypted data and distribute the decryption key only to the group members.

Another context where secure group multicast has been used is in the synchronization of process groups in distributed computer systems. For example, the Isis distributed programming toolkit [13] maintains synchrony between process groups distributed to several sites and provides some protection against corrupted sites. The Isis protocol has the interesting detail that the public key of the group is a part of the group address. Isis requires reliable communication channels and a trusted central authentication service.

To our knowledge, no completely decentralized systems for group membership management have been presented in the literature. There is usually a group leader or other trusted entity that manages the group. If this central entity fails, the group will cease to exist. The reason for the centralized solutions is probably the need to keep track of the group membership or to make decisions about it in one place. Nevertheless, we believe that a more distributed protocol for group membership management matches better the reality of ad-hoc networks. When communication between network elements is sporadic and nodes are likely to be removed unexpectedly, all group management functions must be distributed to several leaders who may act independently.

Our protocols is based on public-key certificates. Implementations could use standard a key-oriented certificate format such as the SPKI authorization certificate [10, 9] or the one specified by the KeyNote trust-management system [3]. SDSI [14] names can also be used to implement groups.

The downside of the increased distribution is that it becomes difficult to obtain a snapshot of the membership and revocation of member rights cannot be done as reliably as in centrally controlled groups. In our system, membership may be revoked by group leaders but there is no guarantee of the revocation information reaching all the entities outside the group in any fixed time. Naturally, applications

may provide more reliable propagation of the revocation lists. In comparison to other systems based on certificates, we have chosen to revoke membership in a group and not the signature keys, as in most systems based on the X.509 certificate standard [5], or the individual certificates, as in SPKI.

Threshold schemes have been frequently suggested for improving the security of mobile networks where the nodes are in physical danger. Zhou and Haas [16], for instance, propose the use of threshold schemes and signatures [8] in order to prevent one or a few compromised nodes from signing messages for the group. A threshold signature must be signed by several members of the group before it becomes valid. We found that although threshold schemes clearly improve the security of the system, they can easily lead to the denial of service. In practical applications, protection against compromised nodes is often less important than that decisions are made fast and actions taken immediately. This is true even in law enforcement and military applications where the network nodes are frequently exposed to a real physical threat.

3 Managing group membership with certificates

Our system for group membership management is based on public-key signatures and on the use of public-key certificates. A group consists of members, which are identified by their public keys. There is also a public key representing the group as a whole. Some members of the group are leaders, which have the right to admit new members to the group.

Each member of the group possesses a membership certificate, which states that the entity is indeed a member of the group. The certificate is signed by a leader of the group.

3.1 The basic group structure

To create a new group, one generates a new signature key (KG) for the group, the *group key*. This group key will be used as the identifier of the group and to certify the members. The public keys of the members are called the *member keys*. Originally, when a new group is created, the group key is the only member key of the group.

To become a member of a group, an entity with a public key needs a *membership certificate* signed by the group key KG. Basically, a certificate is issued to the public key of the member and it states that the key is a member of the group KG. The membership certificates can be verified with the public group key. Since the group key is the group identifier, everyone doing business with the group will

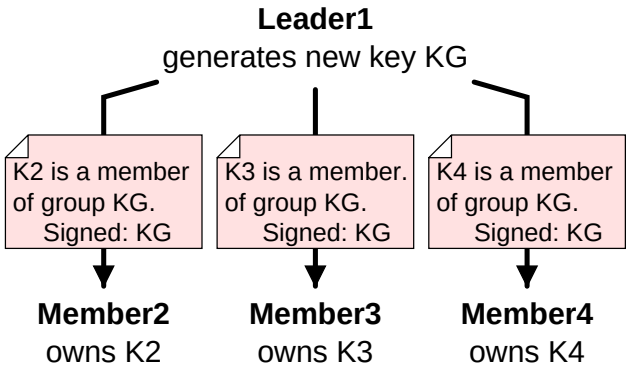

Figure 1: A simple group with 4 members

automatically know it.

A membership certificate contains the following information:

- the group identifier (i.e. the public group key)
- the public member key
- a validity period
- a signature signed with the group key

The group key is called a *leader key*, since it can be used to add new members to the group. Accordingly, the owner of the group key is the *leader* of the group. Note that anyone capable of generating a fresh public key pair may start a new group and act as the leader of the group.

A group with four member keys (including the group key) is illustrated in Fig. 1. (For simplicity, the validity periods of the certificates are not shown in the figure.) To verify that Member 2 is a member of the group KG, one has to obtain its membership certificate, to verify that the signature on the certificate is properly signed by KG, and to verify that Member 2 possesses the member key $K2$, which is stated on the certificate.

3.2 Increased robustness with multiple leaders

In groups of the the previous section, the owner of the group key is the only member who could let new members join the group. In an ad-hoc network, a node may be out of reach at a moment, e.g., due to a routing problem or to a technical failure of the node. If the leader cannot temporarily be reached, all decisions related to

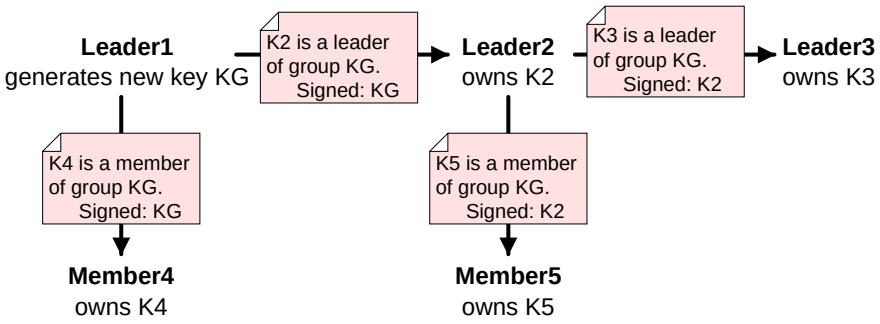

Figure 2: A group with multiple leaders

the membership must wait until a connection is established again. If the leader is permanently removed from the system, the only way to make any changes to the membership is to establish a new group with a new leader.

In order to increase the robustness of the membership management, the authority of the leader must be distributed to several members. We do this by letting the original leader (i.e. the group-key owner) delegate the leadership to other members. It can authorize other keys to act as equivalent leaders by issuing *leader certificates.*

All leaders of a group are also its members. Hence, a leader certificate is a form of member certificate, and the certificates have the following additional field:

- member or leader

Since the new leaders have an equivalent authority with the original leader, they all may admit new members and new leaders to the group by signing certificates with their own keys. Figure 2 shows an example of a group with multiple leaders.

In order to prove membership in the group, a member that has been certified directly by the group key needs its own private key and the membership certificate. A member that has been certified by another leader needs all the certificates in the path from the group key to its member key. Hence, when a leader certifies other leaders or members, it must pass along all the certificates that prove its own status as a leader in the group. This way, a chain of certificates is formed from the group key to each member key.

Fig. 2 illustrates the multiple leaders. For example, when Member 5 of Fig. 2 joins the group, he gets the membership certificate signed by Leader 2 and the leader certificate issued to Leader 2 signed by KG. The latter certificate proves the authority of Leader 2. When Member 5 wants to prove its membership, it presents both certificates and proves its possession of the key $K5$.

The manner in which the leaders and members are added to the group creates a tree

structure. The group key is at the root of the tree and the members without leader status are at the leaves.

If any one of the leaders is removed from the system either temporarily or permanently, the other leaders may still continue their operation. Thus, the distribution of authority increases the robustness of the network against the loss of nodes or connections. A disadvantage of having multiple leaders acting independently may be that there is no single entity aware of all the members of the group.

3.3 Protection against the compromise of keys

We wish to protect the group membership protocol against the compromise of keys. The nodes of an ad-hoc network are often exposed to physical danger and, regardless of any safeguards, it is likely that some private keys or equipment containing them will sometimes fall into the hands of the adversary.

Reconstitution of the group is a secure and often recommendable way to continue with the trusted members only. However, full reconstitution may acquire time as some of the trusted nodes might occasionally be out of reach. The members of a group also need some instant mechanisms for canceling the membership of a single key without sacrificing membership of the other, still trusted members.

Unfortunately, canceling a membership that has already been granted is not easy. The membership certificates may be created, stored and verified concurrently at different parts of the system. There are basically two ways of getting rid of untrusted members: membership expiration and membership revocation.

Group reconstitution To replace a group with a new one, a new group key is generated and new membership and leadership certificates are issued to the old members. Group reconstitution may be done periodically or when there has been enough changes in the group membership. Because of its conceptual simplicity, group reconstitution should be used as the primary protection against compromised members. It is possible to to create the new group while the old one is still functional and gradually shift from using the old one to the new one. That way, the new group key and the new certificates can be propagated to all necessary nodes before the old group key is abandoned. (The subgroup certificates presented in Sec. 4.1 can be used to attach the old group temporarily as a subgroup of the new one.)

Membership expiration The membership certificates may have a validity period that is decided by the issuer. By choosing short validity periods and by refreshing the certificates frequently, the leaders of the group can periodically review the members' status. The expiration mechanism has the major advantage that, once a

membership certificate has expired, it can be simply forgotten. No communication or storing of the expired certificates is needed. On the other hand, the members that want to retain their membership have to periodically obtain new certificates. The expiration also relies on loosely synchronized clocks at every network node.

Importantly, the issuer of short-lived certificates has to periodically consider the reliability of the member. Thus, the limited lifetime keeps up awareness of the security risks and protects against long-term accumulation of compromised members. The issuers may adjust the certificate lifetime for each member depending on the cost of distributing the refreshed certificates and the likelihood of a compromise. For example, the cost of distribution is higher for a leader certificate as the refreshed certificate must be passed down to every member admitted by that leader earlier. Therefore, the leader certificates are likely to have fairly long lifetimes.

Membership revocation Revocation enables the network to react immediately against the possibility of a security failure. However, when a decision is made to revoke a membership, the information about the revocation must be propagated to all the parts of the system where relevant certificates may be used. Distribution over unreliable connections may cause that the information about a revoked membership may not reach all the entities that need it. Delays in the propagation of the revocation data may also be considerable. Hence, there will always be a margin of error where a revoked member might be accepted as a valid one.

We have chosen to give all the leaders of a group the right to revoke themselves and any other leaders and members of the same group. They do this by signing revocation lists of group-key pairs. The lists and additions to them are propagated in a best-effort manner to all the members of the group and to other parties that may verify the group membership. The exact method of distributing the information depends on the application and on the network management and routing protocols.

Members should be revoked only when there is a reason to suspect that the private key has been fallen into the hands of an adversary. The cost of distributing the revocation lists is too high for anything but emergency situations. For example, if the private key has only been lost, revocation is not needed. Leaders may be revoked just like ordinary members but the operation will also affect all the members certified by the revoked key.

If a private leader key has been compromised, the adversary in the possession of the key may revoke other members and leaders. Revocation lists signed by already revoked leader keys must nevertheless be accepted. The reason is that if the compromise is noticed and the leader key revoked, the adversary could timestamp its revocation lists with a date prior to its own revocation. So, in a case where two leaders revoke each other, it is impossible to know which one of the leaders is really the compromised one. Therefore, both keys must be considered revoked.

3.4 Increased robustness with erased group keys and redundant certificates

One effect of the expiration or revocation of a leader key is that it causes not only the removal of that leader but also affects every member below the removed leader in the tree structure formed by the certificates. The result is particularly dramatic if the membership of the group key itself is revoked. For if the group key becomes under suspicion and needs to be revoked, the whole group perishes.

Revocation of the group key may be desired when one wants to replace the group key with a new one and reconstitute the entire group. However, unexpected revocation of the group key may also cause confusion. In this section, we discuss some techniques to counter the effects of removing possibly corrupted leaders from the group.

Erasing the group key A perfect way of protecting the private key against a compromise is to erase it. An erased key cannot be recovered or misused in any way. The certificates signed with the erased key continue to be valid and they can still be verified with the public key. Of course, the erased key cannot issue any new certificates. A private key is at most a few kilobits long and therefore relatively easy to purge from the memory. If one regularly maintains large sets of keys, the techniques of Crescenzo et al. [7] may be used to erase the keys reliably.

In our group context, the newly generated private group key can be used to certify a few leaders and then erased. Several leaders should be certified with the group key so that if the membership of one of them must be revoked, the remaining leaders can still continue to administer the group. The group members should be informed that the group key is protected and cannot be compromised. This way, they know that the group key will never be revoked. For example, Leader 1 in Fig. 3 uses the group key to certify its own key and another key as leaders of the group. After signing the certificates, the group key is erased from Leader 1's memory.

The leader certificates signed with the erased group key must have long enough lifetimes. The certificates signed by a key cannot be refreshed after erasure of the key, so when these leader certificates are about to expire, the group needs to be reconstituted.

Issuing redundant certificates Erasing the private group key removes a single point of failure in the sense that there is no single key whose compromise would disable the entire group. However, large parts of the certificate tree can still be removed from the group by revoking one of the leaders certified by the group key. A member can alleviate this threat by obtaining multiple independent certificates. The leaders may also issue redundant certificates to each other.

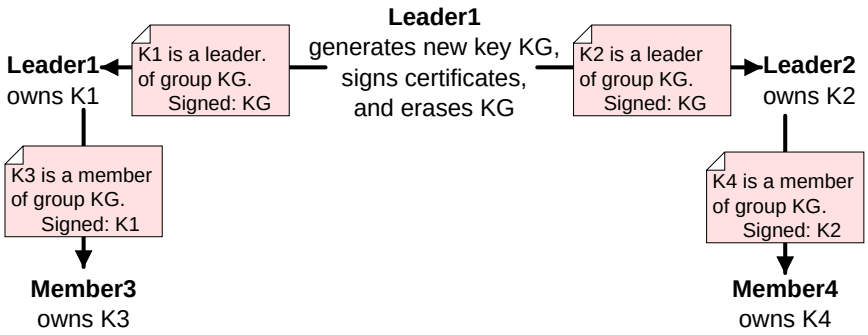

Figure 3: Erased group key

In Fig. 4, even if Leader 2 loses its authority (its leader certificate expires or its membership is revoked), Member 4 still remains a member in the group. Note that if the membership of either Member 3 or Member 4 should be revoked, redundant certificates do not prevent or complicate the revocation in any way. In fact, by revoking the whole membership of an untrusted member, the leader need not be aware of the certificates that have been issued to the compromised member. This is why we have chosen to revoke the memberships and not single membership certificates.

In the tree structure formed by the certificates, the chains can become too long to be practical. The deeper the tree, the higher the cost of storage and verification of the certificate chains. The lengths of the certification chains also tend to increase when keys are erased. This can be countered by letting the members obtain redundant certificates directly from top-level leaders and use them instead of the original certificates given when the members first joined the group.

The redundant certificates confuse the tree structure of the group, and there will exist many certification chains for one member instead of the one leading from the member to the root. A member should be aware of the different chains, so that when one chain is broken, he knows to use the others.

4 Relations between groups

The groups as defined so far are independent of each other. All member and leader certificates include the group identifier and they have effect only on the membership of that group. The same is true for the entries of a membership revocation list. The only way the groups may interact is that a key may be a member in more than one group, or a physical entity may possess several keys that are members of different groups. Public-key certificates, however, are capable of expressing far

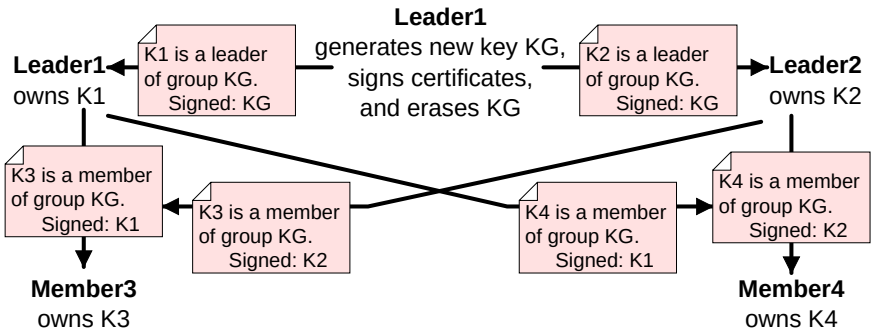

Figure 4: Redundant certificates

more complicated relations between keys and groups.

One important relation between groups is that one group may be a subgroup of the other. Informally, a subgroup is a group, that has its own management as a part of another group. In this way, the management of bigger groups can be further distributed. Subgroup structure may also create a hierarchy or follow an existing one. A new type of certificate, that indicates the subgroup structure, is presented in this section.

Dealing with revocation becomes more challenging as the groups grow. Subgroup structure brings yet more aspects to revocation that are also discussed. Finally, other potential types of certificates are shortly mentioned. Certificates could indicate 'friends' that do not belong to the group but provide some services like routing to it.

4.1 Subgroups

Subgroup relations are created with *subgroup certificates.* With such a certificate, it is possible to admit all the members of a group to another group.

Any group leader may issue a subgroup certificate to another group. The certificate is issued to the identifier (i.e. public group key) of the subgroup. Like the member and leader certificates, it also states explicitly the name of the issuing group and it is signed by a group leader of the issuing group.

The group whose leader issued the subgroup certificate is called the *supergroup*. All the members of a subgroup are also members of the supergroup. Nevertheless, the subgroups preserve their own management. The leaders of a subgroup may continue to admit new members and to revoke memberships in the subgroup. Consequently, any changes in the subgroup affect also the membership of the supergroup.

Membership of a group is transitive in a hierarchy of subgroups. That is, if $KG2$ is a subgroup of $KG1$ and $KG3$ is a subgroup of $KG2$, then the members of $KG3$ are also members of $KG1$. The leadership, on the other hand, is not transitive. The leaders of a subgroup do *not* automatically become leaders of the supergroup. Hence, they do not have the power to admit new leaders to the supergroup, or to certify or revoke members outsider their own subgroup. Furthermore, the subgroup is independent in the way that the supergroup can only certify and revoke the entire subgroup as its members. The supergroup has no authority on the individual members of the subgroup. It is, of course, possible to get around these limitations by issuing leader certificates directly to those members of the subgroup (supergroup) who should have leader rights in the supergroup (subgroup).

Figure 5 shows how the group $KG2$ becomes a subgroup of the group $KG1$. The leaders of the groups $KG1$ and $KG2$ can continue to admit new members, but not directly to each others' groups.

4.2 Subgroups and revocation

We have chosen to limit each group's rights to its subgroups and supergroups to the minimum. This has been done to keep the membership management functions as simple and local as possible. Even then, the introduction of subgroups causes some profound global changes in the system.

First, the subgroup leaders can admit new members in unexpected ways. All the subgroups may add members to their groups, and thus indirectly to the supergroup. The subgroups may also issue subgroup certificates, which transitively introduce members to all the supergroups. It may happen that the members of a group are not even aware of all their supergroups nor their subgroups.

Second, when changes to the membership are made non-locally, in subgroups, the nature of the membership revocation changes. The revocation lists are primarily distributed to the group members that have membership certificates written for the same group key as the revoked one. Inside a single group, it is easy to recognize the relevant items in a revocation list and the nodes to which they should be propagated. With subgroups, this becomes more difficult.

The problems with revocation stem from the distributed nature of the system. In ad-hoc network, there is no central place to keep track of the revoked members. It is also impossible to automatically propagate all revocation data to everyone in the network because certificates may be issued and revoked by anyone. (Anyone may create one's own groups, even the adversary.) Since communication in an ad-hoc group may be completely unreliable and sporadic, we also cannot require the groups to create their own reliable channels for distributing the revocation lists.

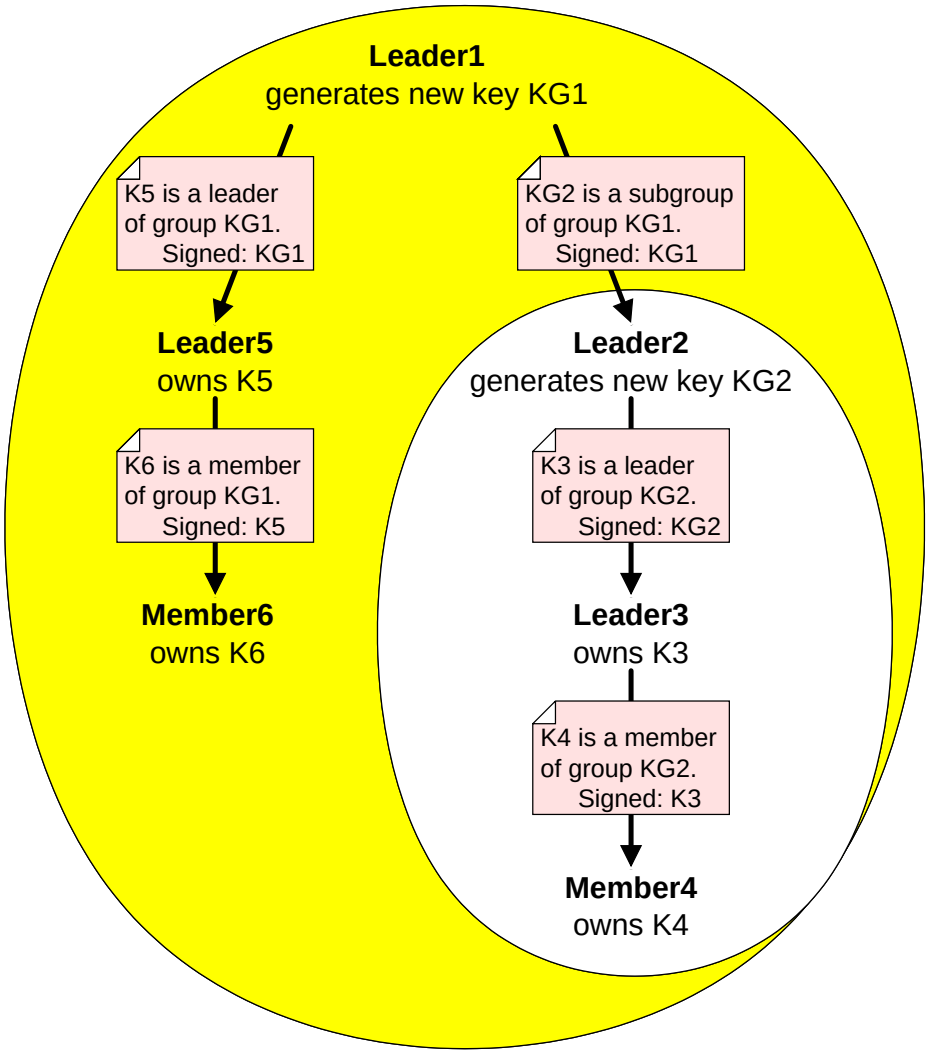

Figure 5: Subgroups

It is still beneficial to have a revocation system, given that every participant recognizes its unreliability and acts accordingly. This means that when a group wants to keep track of its membership at one or a limited number of places, the subgroup certificates must be issued with care, and the subgroup leaders must report back to the supergroup about any new members. Additionally, it must be left to user agreements how the revocation information is exactly distributed. As specified in the beginning, revocation is considered as a best-effort process instead of a perfectly reliable one.

The members of the ad-hoc network may belong to several different groups. The revocation messages always announce from which group a key was revoked. Hence, the revoked member may remain a member in the group through some other subgroup. It is therefore a good idea for the members to always check if the key they are dealing with has been, according to their current revocation list, revoked in any group, and to treat with special care keys that have been revoked in anywhere.

4.3 Other relations

Depending on the application, other types of relations between groups may be needed. For example, the routing and network management functions of ad-hoc networks may require access control arrangements or other trust relations between the groups. These relations can be expressed as certificates similar to the subgroup certificate. The semantics of each new type of certificates has to be defined separately. In particular, it must be decided if any access rights are propagated transitively or not.

For example, we would like to connect the group management with ad-hoc network routing. The groups could also certify others as their ‘friends’ if they allow messages of those groups to be routed through themselves or allow others to route their messages. These ideas require further study.

5 Conclusions

We have presented a protocol for managing groups in ad-hoc networks. The protocol is robust against physical failures in the network and independent from any permanent network services. Groups may be formed at any time and by any entity, and they can be maintained with little effort. Membership of a group is dynamic, and the membership management is distributed. The groups have identities and are distinguishable from each other. Throughout the design of the management, the characteristics of ad-hoc networks are taken into account.

As this is a rather new approach to the security of ad-hoc networks, there are many

issues that require further examination. Some open questions are an optimal best-effort revocation procedure, and the relation of the groups with routing and network management.

Acknowledgments

This research was funded by Defence Forces Research Institute of Technology, Finland. Silja Mäki and Tuomas Aura were also supported by Helsinki Graduate School in Computer Science and Engineering (HeCSE) and by Academy of Finland grant #47754.

References

[1] N. Asokan and Philip Ginzboorg. Key-argeement in ad-hoc networks. *Elsevier Preprint*, 2000. To appear.

[2] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik. New multiparty authentication services and key agreement protocols. *IEEE Journal on Selected Areas in Communications*, 18(4):628–640, April 2000.

[3] Matt Blaze, Joan Feigenbaum, John Ioannidis, and Angelos Keromytis. The KeyNote trust-management system version 2. RFC 2704, IETF Network Working Group, September 1999.

[4] Mike Burmester and Yvo Desmedt. A secure and efficient conference key distribution system. In *Advances in Cryptology - EUROCRYPT '94*, volume 950 of *LNCS*, pages 275–286, Perugia, Italy, May 1994. Springer.

[5] CCITT. *Recommendation X.509, The Directory - Authentication Framework*, volume VIII of *CCITT Blue Book*, pages 48–81. 1988.

[6] Wenli Chen, Nitin Jain, and Suresh Singh. ANMP: ad hoc network management protocol. *IEEE Journal on Selected Areas in Communication*, 17(8):1506–1531, August 1999.

[7] Giovanni Di Crescenzo, Niels Ferguson, Russell Impagliezzo, and Markus Jakobsson. How to forget a secret. In *Proc. 16th International Symposium on Theoretical Aspects of Computer Science (STACS '99)*, volume 1563 of *LNCS*, pages 500–509, Trier, Germany, March 1999. Springer.

[8] Yvo G. Desmedt. Threshold cryptography. *European Transactions on Telecommunications*, 5(4):449–457, July–August 1994.

[9] Carl Ellison, Bill Franz, Butler Lampson, Ron Rivest, Brian M. Thomas, and Tatu Ylönen. Simple public key certificate. Internet draft, July 1999.

[10] Carl Ellison, Bill Franz, Butler Lampson, Ron Rivest, Brian M. Thomas, and Tatu Ylönen. SPKI certificate theory. RFC 2693, IETF Network Working Group, September 1999.

[11] Li Gong and Nachum Shacham. Multicast security and its extensions to a mobile environment. *Wireless Networks*, 1:281–295, 1995.

[12] Vicki Johnson and Marjory Johnson. How IP multicast works. White paper, IP Multicast Initiative (IPMI), 1997.

[13] Michael Reiter, Kenneth Birman, and Li Gong. Integrating security in a group-oriented distributed system. In *Proc. 1992 IEEE Symposium on Research in Security and Privacy*, pages 18–32, Oakland, CA USA, May 1992. IEEE Computer Society Press.

[14] Ronald L. Rivest and Butler Lampson. SDSI - a simple distributed security infrastucture. Technical report, April 1996.

[15] Gert Roelofsen. TETRA security - the fundament of a high performance system. In *Proc. TETRA Conference 1997*, 1997.

[16] Lidong Zhou and Zygmunt J. Haas. Securing ad hoc networks. *IEEE Network Magazine*, 13(6), November-December 1999.