

Matthias Hollick:

Security Awareness in Service Discovery for Multimedia Collaboration. In: *Proceedings of ACM*

[Hol01] *Multimedia 2001 Workshops, Multimedia and Security: New Challenges*, p. 60--63,

Association for Computing Machinery, Inc. (ACM), New York, NY, USA, October 2001. ISBN

ISBN 1-58113-393-6.

Security Issues in Group Integrity Management for Multimedia Multicasting

Andreas Meissner¹, Matthias Hollick¹, Lars Wolf², Ralf Steinmetz^{1,3}

{meissner, hollick}@ipsi.fraunhofer.de, Lars.Wolf@uni-karlsruhe.de, Ralf.Steinmetz@KOM.tu-darmstadt.de

¹ Fraunhofer Institut für Integrierte Publikations- und Informationssysteme, Dolivostrasse 15, 64293 Darmstadt, Germany

² University of Karlsruhe, Zirkel 2, 76128 Karlsruhe, Germany

³ Darmstadt Technical University, KOM, Merckstrasse 25, 64283 Darmstadt, Germany

Abstract. Existing multicast group management solutions fall short of providing adequate support for secure *group integrity* maintenance. This is especially true in the *multimedia* context, where, on top of intra-group integrity, inter-group (i.e. inter-media) considerations have to be taken into account. We demonstrate, along a comprehensive example, the applicability of our *integrity framework* for multicast groups, which includes *integrity conditions* imposed on groups describing valid group states and state transitions as well as *action* and *transition* policies for maintaining integrity. Related *security issues* are analyzed and discussed for data and control plane.

1 Introduction

Multicasting has been discussed for more than a decade as an efficient way to exchange data within groups [1], however it has not yet found widespread application in the Internet. One of the reasons for this failure is the lack of powerful, semantically rich and secure group management capabilities in current approaches. Our work, carried out in the context of the *GCAP project* [2], addresses this shortcoming and suggests a group integrity framework for multimedia multicasting, to be applied within a three-tier structured group model. In this conceptual model, we have one multicast *group* per type of media, and subdivide a group into *subgroups* for actual data transfer (e.g. subgroups for a low quality and a high quality version of a video stream).

A group, along with all its subgroups, is managed by a central *group manager*. Several media groups form a *meta group* that is controlled by a *meta group manager* and integrates conceptually, for example, interrelated audio and video groups. On all three levels, our approach allows us to specify a variety of *integrity conditions* [3] as part of our policy framework, which includes integrity conditions on *state* and *state transition* as well as *action* and *transition* policies for group management. As the acceptability of a solution often depends on its robustness and security awareness, a major part of the work presented in this paper is concerned with *security aspects* of multicast group management, including a discussion of security goals and architectures.

2 Integrity Framework

In our multicast group integrity framework [4], which we apply on top of existing lower-layer group management protocols such as IGMP or MLD [6], a *group manager* is in charge of monitoring and enforcing *group integrity conditions*. We therefore mandate all users who want to become a group member, or perform any other operation relevant to group integrity, to first obtain *permission* from the group manager. The group manager may also *instruct* members to perform specific operations. Our group integrity protocol

builds on reliable unicast and is applied between the central group manager and a user, or, more precisely, an application-layer module referred to as *user controller* that we deploy at each user. Furthermore, group managers communicate with the meta group manager using our protocol.

Group integrity conditions are requirements imposed on groups, describing valid *group states* and *state transitions* with regard to the group's *membership set* (including member roles and rights), *topology* and *organization*. For example, the number of senders in a group might be limited to one. *Traffic integrity conditions* describe requirements that user data traffic has to meet, e.g. mandatory encryption or adherence by all senders to a specific coding format. As the manager might be unable to monitor all user data traffic, it may outsource this job to trusted members called *agents*.

If a group integrity condition is not met, group integrity is said to be violated. In such a case, the group manager has to re-establish integrity according to an *action policy*, i.e. a procedure stating how to react on an observed violation of an integrity condition. For example, a member producing illegal data might be forcefully removed from the group. A *transition policy* is a procedure to be applied if conflicting requests have been received from users that would result in violation of integrity conditions in the next state or by the transition to the next state. For example, if too many users want to become a sender, only the first requests are granted. We also specify when to proceed from *collecting* requests to *evaluating* them. Integrity conditions and action and transition policies make up our policy set; they are applied on *subgroup* and *meta group* level accordingly.

While subgroup policies only address a single subgroup, the group policy set includes conditions to be applied *across* subgroups, e.g. requiring a balanced member count for all subgroups. Similarly, meta group policies may refer to several of its groups, e.g. by requiring that all media group members must also be members of some control group.

We use the following notation:

$G_j, j \in \{1, \dots, M\}$: groups
 $G_j^k, k \in \{0, \dots, N-1\}$: subgroups of G_j
 $M(G_j) \equiv M(G_j^k) \forall k$: group manager of G_j
 G^I, G^{II}, \dots : meta groups
 $MM(G^I)$: meta group manager of G^I
 $P(G_j^k), \text{sink}(G_j^k), \text{source}(G_j^k)$: set of members, receivers, senders in subgroup G_j^k , respectively. Accordingly for groups and meta groups.
 $^iP(G_j^k)$: set of members of G_j^k during state i . Accordingly for sinks and sources, groups and meta groups.
 $\text{scope}(G_j)$: geographical or topological reach of G_j .

In our group integrity framework, for which we will give an example in section 3, we assume that the following tasks have been accomplished out-of-band beforehand:


- Users have been registered in a *user directory*. Any user who is not found in this directory is considered *unlisted*.
- At users who intend to participate in the integrity protocol, the user controller module is operational.
- Multicast groups have been created with no members and an initial set of integrity conditions that must, at least, allow one *administrator user* to join the group who has the right to modify the integrity policy set and create subgroups. The group managers are operational.
- If a meta group is to be formed later, the meta group manager is operational, without associated groups.

3 Example: Multicast Lecture

In this section we demonstrate the applicability of our framework along a comprehensive example: A multimedia lecture with video, audio and whiteboard media, given by a professor from his office, is transmitted by multicast to an audience in the United States. All multicast listeners can receive whiteboard and audio traffic, and California users can additionally receive video, generally at low quality, but in the San Francisco Bay Area also at high quality. Occasionally, the professor interrupts his lecture in order to allow students to ask questions. While every registered user in the country is allowed to listen, only registered students have the privilege to ask such questions. We use this scenario as an illustration of our integrity framework; we do however not intend to mimic dedicated video conferencing tools, which come, for example, with powerful floor control capabilities. Where appropriate, we refer, space permitting, to our security discussion in section 4.

3.1 Initial State

For our example, we consider the following initial state: The (simplified) user directory UD stores, for all registered users shown in figure 1, User ID, Name, Administrator status, Region, and Type (1=student, 2=professor, 0=other).



UD

UID	Name	Admin	Region	Type
1	Ann	false	USA.CA.LA	0
2	Bob	true	USA.CA.Bay	0
3	Charles	false	USA.IL	1
4	David	true	USA.CA.Bay	2
5	Eric	false	Germany	1
6	Fred	false	USA.CA.Bay	1

Figure 1 - User Directory

A finer granularity for "administrators" is not needed here; for *keys* required additionally see our discussion in section 4. Three groups have been established with operational group managers and initial integrity conditions as follows:

G₁: Video group; **G₂**: Audio group; **G₃**: Whiteboard group

$$P(G_1) \subseteq \text{admin}(UD) \quad (1)$$

$$P(G_2) \subseteq \text{admin}(UD) \quad (2)$$

$$P(G_3) \subseteq \text{admin}(UD) \quad (3)$$

where $\text{admin}(UD)$ is the result of the query

select UID from UD where Admin = true

Groups initially have no members and no subgroups. The meta group manager $MM(G^1)$ is operational, but no groups are *associated* to G^1 . By default, administrators are the only individual users who may join a meta group; in this example they are required to approve any request received from a group manager to associate its group to the meta group.

3.2 Basic Configuration

Administrator Bob and $M(G_1)$ establish a mutual trust relationship as discussed in section 4. Then Bob sends, by way of his user controller, a join request as sink for G_1 to $M(G_1)$. For reasons we will see later, the manager immediately checks this request against (1), grants it, Bob performs the join, and we have $P(G_1) = \text{sink}(G_1) = \{2\}$. Subsequently, Bob creates two subgroups for user data transfer: G_1^0 for high quality and G_1^1 for low quality video, with the following integrity conditions:

$$\text{scope}(G_1^0) = \text{USA.CA.Bay} \quad (4)$$

$$|\text{source}(G_1^0)| \leq 1 \quad (5)$$

i.e. only Bay Area users may join, and at most one source is permitted. Note that we did not opt to specify lists of acceptable or non-acceptable *individual* users. For G_1^1 we have:

$$\text{scope}(G_1^1) = \text{USA.CA.*} \quad (6)$$

$$|\text{source}(G_1^1)| \leq 1 \quad (7)$$

using * as wildcard. Since all subgroup members are, implicitly, also group members, it is sufficient to demand at *group level* that only registered users (i.e. those listed in UD) may join G_1 and that only registered professors be sources:

$$P(G_1) \subseteq \text{any}(UD) \quad (8)$$

$$\text{source}(G_1) \subseteq \text{prof}(UD) \quad (9)$$

Furthermore, we demand at group level:

$$\text{source}(G_1^1) \subseteq \text{source}(G_1^0) \quad (10)$$

i.e. only users who are a source in G_1^0 may be a source in G_1^1 , since a lecture with a different professor appearing in the low-quality video makes little sense. By (10) it is not possible for a professor to feed only G_1^1 but not G_1^0 .

Generally, a group state is divided into a *collection* phase, during which the group manager collects user requests, and an *evaluation* phase, during which the set of received requests is evaluated against integrity conditions. All subgroups follow the same phase timing. Thus, the above conditions would allow a professor to hand over "on the fly" a lecture to another professor regardless of the order in which the associated join and leave requests are sent, provided they are made during the same state's collection phase. However, the question arises how to handle the (admittedly unlikely) case of two new professors competing to take over an ongoing lecture. To resolve the resulting conflict (e.g. two join requests for G_1^0 , but only one leave request received by $M(G_1)$, leading to an imminent violation of (5) in the subsequent state), a subgroup *transition policy* is applied. In our example, this might be to grant only the leave request and the *first* join request. Furthermore, a group transition policy has to address the case that the professors fail to request the same handover for G_1^1 during the same phase, which would result in an imminent violation of (10). Transition policies, like integrity conditions and action policies, may be changed by administrators during the collection phase, to become effective for the next state. In our example, we assume that an administrator manually triggers the group manager to proceed from the collection to the evaluation phase, initiating a transition to the next state. In order to avoid a deadlock, if no administrator is currently a member, a join request from an administrator is immediately evaluated and, if granted, performed.

Having now configured the video group, Bob joins, according to the request and authentication procedure

outlined above, audio group G_2 and whiteboard group G_3 . He creates one subgroup for each, G_2^0 and G_3^0 , and applies:

$$\text{scope}(G_2) = \text{USA}. * \quad (11)$$

$$P(G_2) \subseteq \text{any}(UD) \quad (12)$$

$$\text{source}(G_2^0) \subseteq \text{prof}(UD) \quad (13)$$

$$|\text{source}(G_2^0)| \leq 1 \quad (14)$$

(noting however that we will later allow registered students to act as audio sources while asking questions) and likewise

$$\text{scope}(G_3) = \text{USA}. * \quad (15)$$

$$P(G_3) \subseteq \text{any}(UD) \quad (16)$$

$$\text{source}(G_3^0) \subseteq \text{prof}(UD) \quad (17)$$

$$|\text{source}(G_3^0)| \leq 1 \quad (18)$$

Next, Bob joins G^1 . As an administrator member of G_1 , G_2 and G_3 , he requests at $\text{MM}(G^1)$ (by way of the respective group managers) that G_1 , G_2 and G_3 be associated to G^1 , which is granted after his approval as G^1 administrator. As a G^1 administrator, he adds meta group integrity conditions:

$$P(G_2) = P(G_3) \quad (19)$$

$$P(G_1) \subseteq P(G_2) \quad (20)$$

i.e. the membership set of the audio and the whiteboard group must be identical, and no other users than audio group members are permitted as video group members. We do not specify any transition policies here and do not require group managers to obtain permission from $\text{MM}(G^1)$ for their operations, so no attempt is made by the meta group manager to resolve any conflicts arising from user requests (such as the failure of a user to join both G_2 and G_3). Consequently, violations of (19) or (20) may occur. Our *action policy* in this case is to perform a forced-leave for non-complying users and to notify the meta group administrator about the issue.

Finally, Bob disables (1), (2) and (3) and sends leave requests for all groups and the meta group to the respective managers. They are granted, so he performs the leaves, and the subgroups, groups and the meta group are ready for use.

3.3 Membership Build-up

User David, a professor and administrator, establishes mutual trust relationships *with* and sends join requests as source and sink *to* the respective managers for G_1^0 , G_1^1 , G_2^0 and G_3^0 , which are immediately evaluated, granted and performed. They result in implicit joins as source and sink to G_1 , G_2 and G_3 . He also joins G^1 as an administrator.

During the following state, the group managers receive a number of join requests. For ease of expression, we now say that users “join” instead of “authenticate themselves and send join requests to be evaluated by the group manager upon a trigger signal by an administrator”. For a list of keys to be exchanged, we refer to section 4.3.

User Fred joins as sink G_1^0 , G_2^0 and G_3^0 , resulting in the respective implicit group joins. User Ann performs the same joins, however, due to (4) and (6), she has to pick G_1^1 rather than G_1^0 . User Eric’s attempt to join any subgroup is rejected by the responsible group managers because his membership would violate (4), (6), (11) and (15).

User Charles knows that he is not admissible to any video subgroup, but he is unaware of (19), so he initially joins only G_2^0 (and G_2), which $\text{M}(G_2)$ permits since it is not in charge of evaluating (19), and $\text{MM}(G^1)$ did not require group managers to request prior permission for their operations.

Finally, user Greg, who is *unlisted*, attempts to join G_3^0 . This is rejected by $\text{M}(G_3)$ as it would violate (16).

Just before he is ready to start his lecture, David additionally triggers $\text{MM}(G^1)$ to evaluate the meta group integrity conditions. The only violation detected is Charles’ failure to join both G_2 and G_3 , so, according to the action policy, he is removed from G_2 and therefore from G_2^0 . We assume that he does not make any re-join attempts.

As a result, we now have the following membership sets: $P(G_1^0) = \{4,6\}$; $P(G_1^1) = \{1,4\}$; $P(G_2^0) = \{1,4,6\}$; $P(G_3^0) = \{1,4,6\}$.

3.4 Modification of the Policy Set

The lecture multimedia data is now transmitted in the subgroups. During the course of his lecture, David gives his students the opportunity to ask questions. For this purpose, they have to temporarily receive permission to act as sources in the audio group. Thus, David temporarily disables (13) and (14) and applies new conditions:

$$\text{source}(G_2^0) \subseteq \text{prof}(UD) \cup \text{stud}(UD) \quad (21)$$

$$|\text{source}(G_2^0) \cap \text{prof}(UD)| \leq 1 \quad (22)$$

$$|\text{source}(G_2^0) \cap \text{stud}(UD)| \leq 1 \quad (23)$$

$${}^{i-1}\text{source}(G_2^0) \cap {}^i\text{source}(G_2^0) \cap \text{stud}(UD) = \emptyset \quad (24)$$

Now one registered student at a time can assume a source role in G_2^0 and ask questions. If more than one student wants to ask a question at the same time, we apply a simple transition policy: The first student request received by $\text{M}(G_2)$ is granted, and subsequent requests are rejected. If a student does not voluntarily give up his source role after a while, administrator David may forcefully revoke this role. *State transition* integrity condition (24) prevents a student from immediately grabbing the source role again after having asked questions already in the previous state.

Fred may assume a source role in G_2^0 and ask questions, and, since we did not prohibit this by a state transition integrity condition, even new registered students such as Charles may join G_2^0 and G_3^0 specifically for asking questions in G_2^0 . User Ann, however, may not ask questions due to her non-student status.

At the end of the question period, David disables (21), (22), (23) and (24), re-enables (13) and (14), adds a new action policy for (13) that forcefully revokes the source role in G_2^0 from any member in non-compliance, and triggers a state transition to make $\text{M}(G_2)$ evaluate these conditions.

At the end of the lecture, David “cleans up” by re-applying (1), (2) and (3) and disabling all other conditions. New action policies for these initial conditions require any member in non-compliance to be forcefully removed from the respective group. Finally, David deletes all subgroups, eliminates, by way of the group managers and the meta group manager, the association of G_1 , G_2 and G_3 to G^1 , and leaves all groups and the meta group.

4 Security

An important goal of our integrity framework is to ensure that membership in multicast groups can be controlled in a deterministic fashion. As we have seen, group and meta group managers have the responsibility to perform control tasks and enforce integrity conditions. Since their job raises a number of security issues, such as the need for user authentication, we discuss in this section how we enhance our integrity framework with *security mechanisms* built upon cryptographic methods. In the following, we first

show *preconditions* and analyze the *security needs* with respect to our integrity framework. Thereafter, we embody the necessary *security goals* for control and data flows. Finally, we present our approach towards security within the multimedia multicast integrity framework.

4.1 Preconditions

Since we deal with a closed administrative domain, we assume that a *Public Key Infrastructure* (PKI) is in place, which is coupled to our user directory (UD). To present a trusted concept of identification and to have a reliable starter for further cryptographic operations, we define this PKI to be reliable, keeping in mind however that real-world systems should be carefully examined in this respect [5].

Coming back to our example in section 3, we note that the identity of group and meta group managers must be certain, so nobody risks to accept instructions from malicious managers. Thus, their digital representation must carry their identity and public key information. Certificates stored within the PKI-component of our directory can achieve this. Additionally, users must be identifiable in order to allow membership in determined, i.e. *classified* groups. This is particularly true for members with special functions, such as *administrators* and *agents*. In our scenario, all entities are part of an administrative security domain *SD* defined by UD and the participating group and meta group managers.

4.2 Security Goals for Control and Data Plane

We consider various attacks against the control and the data plane of our framework, including destructive attacks, intellectual property theft, identity theft, and privacy violations. The nature of the attacks varies depending on the target. We further assume that we are dealing with a strong attacker capable of carrying out *passive* (e.g. eavesdropping to catch the data stream) and *active* attacks (e.g. trying to join without permission of the group manager, or inserting malicious data to jam existing multicast groups, or even tampering with entities).

Attacks against the *control plane* may include spoofing of control messages or masquerading of the group manager. Replay attacks may be mounted against the system. Besides the availability of the control path, confidentiality, integrity and validity of any control information transmitted should be ensured. Thus, as a countermeasure, all entities involved in the protocol exchange have to be protected since we are operating in an open networking infrastructure.

Attacks against the *data plane* may include attacks against user data traffic and the network itself. The security goals are analogous to the ones mentioned above for the control plane. Moreover, such attacks may be targeted at the underlying infrastructure services. This includes attacking the PKI (or, in our case, the user directory), cryptanalysis of weak protocols or cryptographic mechanisms, or attacks against the network layer in general.

Trust, Authenticity and Integrity

A meta group manager has to authenticate itself to all managers of associated groups, and vice versa, building a *mutual* trust relationship. This is done on behalf of an administrator, who also has to share a mutual trust to all managers he is in charge of. Building upon these mutual

trust relationships, data integrity and confidentiality is mandatory for all control messages exchanged among the managers, and between the administrators and the managers. Furthermore, a group manager has to authenticate itself to all members of its group, while all members have to prove their authenticity to the manager during the join phase. Building upon this *mutual* trust relationship, authenticity of all control messages, as sent either by the group manager or the group members, is guaranteed. Implementation, however, may include a *session-based* concept (which brings in symmetric cryptography) for the users trusting the managers, and a *transaction based* approach (built upon asymmetric cryptography) for the managers trusting the users, to adjust the security related overhead to the number of expected protocol interactions. *Agents* and other members with special functions, such as *administrators*, are, under security aspects, treated like group managers, and thus a *mutual* trust has to be established by the manager with their respective controllers.

There are two different ways to ensure authenticity at the data plane: authenticity at *subgroup level* (proved by the knowledge of one shared secret) or authenticity at *member level* (i.e. each sender has to sign the messages using his own private key, and each receiver has to check the validity of the signature and determine if the sender is in fact a member of the subgroup). As we deal with closed groups, we regard authenticity at *subgroup level* as adequate (with the drawback that we only can determine if a message comes from *any* member of the group, but not exactly from whom).

Confidentiality

Confidentiality for control messages is mandatory and built upon the mutual trust relationships mentioned above. Analogously to authenticity at subgroup level, we regard data confidentiality at subgroup level, which means that a group manager and all members of a subgroup use a secret key *common to the subgroup* to ensure confidentiality within the subgroup. The authorization of members to participate is checked by the group manager during the join phase.

Availability

Apart from authenticity, integrity and confidentiality, our security analysis addresses *availability* issues. However, availability of the communication partner or data cannot be guaranteed in today's open Internet infrastructure. Denial of service attacks like flooding may take up all bandwidth, effectively interrupting all communication - a problem which can only be resolved at the source(s) of the attack. Under these circumstances we aim at *robustness* of our integrity framework. Filtering all unencrypted messages can attain robustness against inappropriate packets from outsiders. Malicious packets from insiders (which know about the common secret) cannot be dealt with under the condition of authenticity at *subgroup level*.

4.3 Resulting Architecture

As shown in figure 2, the *control flows* for our framework form a hierarchy. $MM(G^1)$ at the top is responsible for overall group integrity issues and by definition has the authority to control, and to take appropriate action against, any manager $M(G_i)$ of an associated group G_i within its administrative security domain *SD*. $M(G_i)$, at the next hierarchy level,

has the authority to perform actions against any user in $P(G_i)$. The resulting trust relationships can be deduced to be bi-directional between $MM(G^j)$ and $M(G_i)$. $M(G_i)$ has to be trusted by each authenticated member of G_i for control flows, and it implicitly trusts each G_i member because of the authentication performed. Moreover, the managers may act on behalf of administrators which share mutual trusts with their respective managers. The implementation of the trust relations is achieved using our user directory and PKI.

The *data flow* is local within subgroups and allows for the easy solution of complete trust among all subgroup members by one shared secret for confidentiality. Integrity and authenticity may be implicitly based on the knowledge of the same secret (i.e. authenticity by encryption).

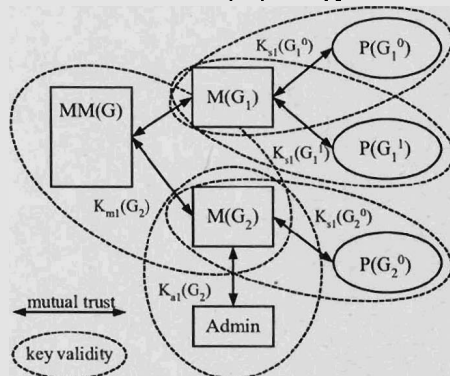


Figure 2 - Selected Trust Relationships and Key Validity

Design: Keys and Key Establishment

Building upon our security analysis, we now introduce the cryptographic foundations of our security framework. The following catalog describes the cryptographic keys and their function within our framework. We are aware of other solutions like IPSec, which would fit for securing the control-plane of our framework, nevertheless we describe our approach towards security in general, which allows to get the overall picture:

$KU_1(P(G_i^j))$, $KU_1(M(G_i))$, $KU_1(MM(G^j))$ and $KR_1(P(G_i^j))$, $KR_1(M(G_i))$, $KR_1(MM(G^j))$ denote the key-pairs (KU - public key, KR - private key) for each entity participating in the system, i.e. all users in $P(G_i^j)$ including administrators, $M(G_i)$ and $MM(G^j)$.^{*} KU and KR are master keys used to derive symmetric session keys. Moreover, a digital fingerprint is derived for identification purposes. We use public key cryptography to ensure the authenticity of the control messages of the managers. Confidentiality of the control messages is assured using $K_{s1}(G_i^j)$ as described below.

A key K_{master} exists for each entity and is generated from the entity's secret (e.g. passphrase). It ensures the secrecy of each entity's private key (which will likely be managed in software). The use of K_{master} with an appropriate algorithm should ensure that the entity's secret is the weakest link (and not some proprietary way to protect the private key) compared to the cryptographic surroundings. (Nevertheless the entropy of the user's secret determines its strength.) For

the manager components, the use of the private key can be activated on behalf of an administrator.

For each subgroup, we have a secret session key $K_{s1}(G_i^j)$. It is used to ensure data confidentiality and authenticity at the data plane. $K_{s1}(G_i^j)$ can be created using some key establishment protocol. ($M(G_i)$ may generate an appropriate key and distribute it securely to all G_i^j members.) However, performance issues in large groups should be carefully investigated and may lead to improved and distributed key agreement protocols. Integrity conditions can be applied to enforce re-keying if necessary, e.g. if a member leaves voluntarily or is forced to leave, or if time or the transmitted data amount reach some threshold.

Between each pair of $MM(G^j)$ and $M(G_i)$, for all i , we have a key $K_{mi}(M(G_i))$ as secret session key for data confidentiality and message authentication. The keying is initiated by $MM(G^j)$ for each associated group.

Between each pair of $MM(G^j)$ or $M(G_i)$ and the respective administrator, we have a key $K_{a1}(M(G_i))$ as secret session key. Key establishment is initiated by the administrator separately for each associated group.

There may of course be additional keys for management purposes or storage of sensitive data on the end-systems.

5 Conclusion

Our multicast group integrity framework, which we illustrated along a comprehensive example, allows us to specify integrity conditions as well as action and transition policies at subgroup, group and meta group level. Based on this framework, we analyzed related security issues. Our resulting security framework is divided into control and data plane security, the former using end-to-end mechanisms to allow for fine-grained granularity of security associations, the latter to allow for efficient and easy-to-handle security mechanisms at subgroup level. Moreover, this distinction facilitates the use of well-known security solutions for parts of the framework.

Network security traditionally has to deal with numerous variables, interacting in a complex fashion. Our approach is in position to carry out security pervasively under the assumption of a trusted and always reachable PKI and an identifiable user community. Derived from the security goals our design principle has been to keep the security part as simple as possible for the given prerequisites. For future work, we consider additional security goals, in particular privacy.

References

1. C. Diot, W. Dabbous, J. Crowcroft: Multipoint Communications: A Survey of Protocols, Functions, and Mechanisms. IEEE Journal on Selected Areas in Communications, Vol. 15 (3), April 1997
2. M. Diaz, R. Canonico, L. Costa, S. Fdida, D. Hutchison, L. Mathy, A. Meissner, S. Owezarski, R. Vida, L. Wolf: GCAP: A New Multimedia Multicast Architecture for QoS. Proc. 6th Int. Conference on Protocols for Multimedia Systems (PROMS 2001), Enschede/NL, Oct. 2001
3. L. Mathy, G. Leduc, O. Bonaventure, A. Danthine: A Group Communication Framework. Broadband Islands '94, W. Bauerfeld, O. Spaniol, F. Williams, eds., Elsevier North-Holland, 1994
4. A. Meissner, L. Wolf, R. Steinmetz: A novel Group Integrity Concept for Multimedia Multicasting. Proc. 8th Int. Workshop on Interactive Distributed Multimedia Systems (IDMS 2001), Lancaster/UK, Sept. 2001
5. C. Ellison, B. Schneier: Ten Risks of PKI: What You're Not Being Told About Public Key Infrastructure. Computer Security Journal, Vol. 16 (1), 2000, pp. 1-7
6. R. Vida, L. Costa, R. Zara, S. Fdida, S. Deering, B. Fenner, I. Kouvelas, B. Haberman: Multicast Listener Discovery Version 2 (MLDv2) for IPv6. Internet Draft - <draft-vida-ml-dv2-00.txt>, Feb. 2001

^{*} To be algorithm independent, we might also use two key pairs per user: one for encryption/decryption purposes and one for signature/verification. For the remainder of this paper we only mention one key / key-pair in our discussion. Nevertheless we mean both the signature and encryption keys.