4

Multicast Security and Reliable Transport of Rekey Messages over Hybrid Satellite/Terrestrial Networks

Franco Tommasi, Elena Scialpi and Antonio De Rubertis
University of Salento - Department of Engineering for Innovation
Lecce (Italy)

1. Introduction

Security problems in satellite environments are one of the obstacles to the widespread deployment of satellite IP multicast and, more generally, of satellite multimedia applications (Cruickshank et al., 1998).
By satellite environments we refer to networks where the satellite plays an essential role. e.g. those where it is used to multicast IP packets to many nodes of a terrestrial network. We also speak of ”Hybrid Satellite/Terrestrial networks” in such cases.
The broadcast nature of satellites makes eavesdropping and active intrusion much easier than in terrestrial fixed or mobile networks. A further issue is specific to multicast: the number of members in a multicast group can be very large and, even worse, can change very dynamically. While the process of performing and securing key management for unicast connections is well understood (Harkins & Carrel, 1998), (Maughan et al., 1998), (Orman, 1998), multicast security is still an open field (see par. 2). Protocols that manage the process of distributing keys in a multicast environment are under development (see par. 2.3 and 2.4).
Access to the encryption key is controlled by a group key management system, which is responsible for sending the encryption key to authorized new users and for performing multicast group rekeying whenever the key changes. Specifically, a group key management system is said to implement two types of access control: backward access control and forward access control. If the system changes the encryption key after a new user joins, the new user will not be able to decrypt past group communications; this is called backward access control. Similarly, if the system rekeys after a current user leaves, or is expelled from the system, the departed user will not be able to access future group communications; this is called forward access control.
Many group key management solutions (see par. 2.2, (Jokela, 2006) (Mah, 2004)) have been proposed and a number of classifications of the available approaches can be found in the current literature (Dondeti et al., 1999), (Rafaeli & Hutchison, 2003), (Eskicioglu, 2003). Moreover, security mechanisms regarding satellite networks have been investigated in (Howarth et al., 2004), (Noubir & Allmen, 1999) and (Arslan & Alagöz, 2006).
Group key management protocols can be categorized as following:

- *Centralized architectures*. A single entity, a GC (Group Controller), is employed for controlling the whole group, hence a group key management protocol seeks to minimize

storage requirements, computational power on both client and server sides and bandwidth utilization.

- *Decentralized architectures*. The management of a large group is divided among subgroup managers, trying to reduce the problems arising from concentrating the work in a single place.
- *Distributed architectures*. There is no explicit manager and the members themselves do the key generation. All members can perform access control and the generation of the key can be contributory, meaning that all members contribute some information to generate the group key.

Rekey protocols should use a scalable Group Key Management Algorithm (GKMA) to send the minimum possible number of keys in a rekey message. LKH (see par. 2.3), OFT (Balenson et al., 2000), Subset difference based schemes (Lotspiech et al., 2001) are examples of GKMA.
Regardless of the chosen approach, rekey messages are generally frequent and their reception must be guaranteed in order for the multicast group members to avoid multicast services interruptions.
RFC 4046 (Baugher et al., 2005) describes a Group Key Management Architecture and proposes three classes of solutions for reliably sending keys to the multicast group members:

- repeatedly transmit the rekey message;
- use FEC for encoding rekey packets (with NACKs as feedback) (Yang et al., 2001);
- use an existing reliable multicast protocol/infrastructure (possibly profiting in a mixed way from the above solutions).

Up to now, not much work has been dedicated to the use of reliable multicast transports for rekey messages. In most cases ((Wong & Lam, 2000) (Zhang et al., 2003)) FEC (Rizzo, 1997) has been used to improve the reliability.
RFC 4046 also identifies the requirements a protocol for key transmission/rekeying must satisfy:

- *Reliability*. Every user must receive all of its (encrypted) new keys, no matter how large the group size.
- *Soft real-time*. It is required that the delivery of new keys to all users be finished with a high probability before the start of the next rekeying.
- *Scalability*. The processing and bandwidth requirements of the key server and those of each user should not increase much with the group size so that a single server is able to support a large group.

Moreover, multicast key distribution must take care of the "feedback implosion" problem (see par. 2.2.4 and (Baugher et al., 2005) resulting from NACKs or ACKs sent as feedback.
Satellite networks may intrinsically offer a serious alternative to terrestrial networks solutions in that they can enable reliable multicast techniques to scale to large group of receivers. Such advantage is an effect of their intrinsic properties such as: high bandwidth availability, their broadcast nature and the reduced occurrence of congestion between sender and receivers as compared to terrestrial networks.
With these considerations in mind, we focused our attention on the following protocols for the multicast reliable transmission of encryption keys: Pragmatic General Multicast (PGM) (see par. 3.1), NACK-Oriented Reliable Multicast (NORM) (see par. 3.2) and our SRDP-Sign (see

par. 3.3). PGM was chosen for being an interesting IETF experimental protocol. While not yet a standard, it has been implemented in some networking devices (such as Cisco routers) and operating systems including MS Windows XP. NORM was chosen because RFC 4046 quotes it as a well-suited protocol for reliable multicast of rekey messages.
In the following, paragraph 2 will detail the state of the art on the subject of multicast security with a particular attention to the solutions based on a centralized approach, paragraph 3 will discuss some reliable multicast protocols with an interest in their utilization in satellite networks. Paragraph 4 will present the preliminary results of some tests we conducted with the aim of evaluating the performances of above listed reliable multicast protocols. They have been tested on a hybrid satellite/terrestrial network in the specific case of transmission/rekeying of keys for a multicast security environment.

2. Multicast security

The original conception of an IP network was aimed at the exchange of information between two nodes. However, very soon the popularity of the Internet gave rise to a number of applications for which a better model would be desirable. Such applications would benefit from a network direct support to the delivery of the same packets from one source to many destinations. Some of them are today's killer applications, e.g. IPTV. The need for multiple unicast connections implied by the basic model made them simply not scalable enough within the original rules.
Around 1989, to address such problem the introduction of a new functionality was proposed for IP networks: IP multicast (Deering, 1991) (Deering, 1989). As a result of it, an host wishing to send the same packet to many hosts at the same time was allowed to output that single packet on its network interface, leaving to the network's routers the burden of duplicating it wherever required. As an extreme example, a packet intended for a number of hosts on a distant LAN would travel alone until the last router which would replicate it at the last hop for as many hosts as needed.
The positive effect of such approach can be perceived, increasingly with the number of the multicast group members, both on the conservation of computational resources of the sending machine and in the (potentially huge) savings of bandwidth resources in the network.
The idea required the introduction of a special class of IP addresses (Class D, from 224.0.0.0 to 239.255.255.255) each of them representing a "multicast group".
The essential protocol for managing the multicast group membership is IGMP (Internet Group Multicast Protocol) (Deering, 1989). It works without problems in a network where all the routers support it. When support is spotty, more complex techniques are required (Semeria & Maufe, 1997).
Although IP Multicast would be the ideal technique for many important applications (e.g. to distribute real-time video on the Internet) for many well-known reasons it is not globally supported on the Internet (Diot et al., 2000). There are indeed many ISPs supporting IP multicast in their AS (Autonomous Systems) and multicast peering agreements are frequent among ISPs but even then the common user isn't left the faculty to send multicast traffic to other users in the same AS. Clearly this ability is regarded as a primary asset within an ISP network and acquiring it (when available) can be subjected to substantial fees. Many methods to overcome this limitation have been proposed (Eriksson, 1994) (*MBONED*, 2011) (Sardella, 2005) but none of them has proved very successful until now.

A natural way to transmit IP multicast over a large geographical area is satellite broadcasting (Tommasi et al., 2010)(Tommasi & C.Melle, 2011).
Among the many applications made possible by the multicast model are those for which security is a critical requirement. Without going too far, the very same IPTV application, when run to pursue economic goals, needs a method to allow only paying customers to access transmitted contents. However many other situations where security is a crucial factor can be imagined (especially in the fields of control and signaling).
According to a recommendation from International Standards Organization (ISO) (*ISO 7498-2*, 1989), while designing a secure system the following criteria are to be considered: confidentiality, integrity, authentication, non-repudiation and access control. To meet such criteria in an IP multicast environment, a Multicast Security (MSEC) Workgroup (*MSEC*, 2011) has been formed within the IETF, with the aim of standardizing protocols for securing group communication over the Internet. Obviously enough, a fundamental topic in the workgroup's activities is the standardization of a group key management architecture. The present paragraph will make use of many of the results coming from the group's efforts and documented so far.

2.1 The multicast group security architecture

The description of the security architecture for IP multicast group communications involves a number of aspects. To reduce the complexity of the presentation, the proposed protocols are grouped in three functional areas, each addressing an aspect of the solution. RFC 3740 (Hardjono & Weis, 2004) outlines the Reference Framework formulated by the IETF Workgroup and identifies such areas (see Fig. 1):

1. *Multicast data handling*. This area includes all the operations on the multicast data performed by the sender and the receiver. Such handling implements:
   - *Encryption*. To support access control and confidentiality, data are encrypted by the use of the group key.
   - *Source authentication and data integrity*. Source identity must be guaranteed by suitable algorithms. Steps are also to be taken to secure the integrity of the received contents.
   - *Group authentication*. This is a minor requirement (guaranteeing the data come from within a group does not necessarily indicate their integrity). However such authentication is very easily achieved and prevents DOS (Denial of Service) attacks.
2. *Group Key Management*. This is the area where secure key distribution and the refresh operations are dealt with.
3. *Multicast Security Policies*. According to (Hardjono & Weis, 2004) Multicast Security Policies represent "the security mechanisms for the group communication" and "the rules for the governance of the secure group".

The Framework also identifies the main elements of a multicast security architecture both in a centralized and in a distributed solution. A central role is played by the "Group Controller" and by the "Key Server". Such entities are usually merged in a single server (GCKS) which is responsible for the "Group Key Management" functional area. Senders and receivers (called GM, Group Members) do interact both with GCKS and with the "Policy server", which is in charge of the "Multicast Security Policies" area.
In order to increase the scalability of the architecture, a distributed approach (see Fig. 1), based on a number of cooperating GCKS, can be opted for. In such case mutual authentication must

be guaranteed among GCKS. In a distributed system all receivers will comply with the same security policies and receive the same keys.

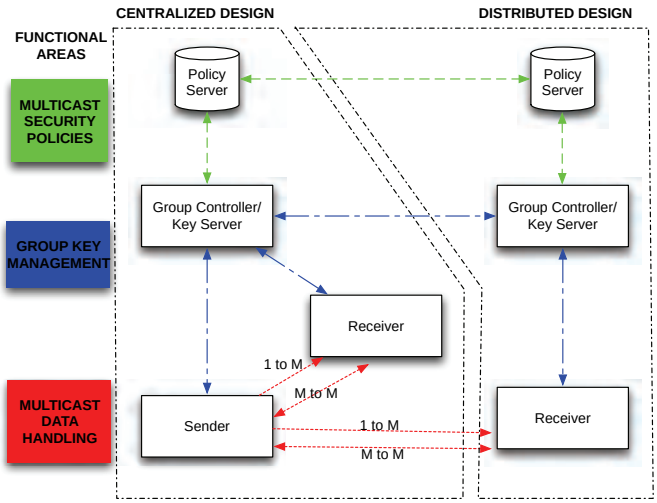


Fig. 1. Multicast Group Security Architecture (from (Hardjono et al., 2001))

2.2 Group Key Management Architecture

The Group Key Management Architecture (Hardjono & Weis, 2004) defines the Group Security Association (GSA) and the main features of the registration and the rekey protocols.

2.2.1 Group Security Association (GSA)

In a protocol designed to manage security on an end-to-end connection, such as IPSEC (Kent & Atkinson, 1998), a Security Association (SA) is a set of shared attributes used by the two ends to secure the connection. Such attributes consist of cryptographic keys, algorithm, identifiers and everything else needed to conduct the communication.
The complexity of a multicast environment imposes the need for more than one key to secure a session. In this context the notion of Group Security Association (GSA) (see Fig. 2) is introduced (Hardjono & Weis, 2004) (Hardjono et al., 2001), which stands for a group of SAs related to the session. SAs in a GSA belong to three different categories:

- REG SA (Registration SA) is used to set up a full-duplex unicast communication channel between GCKS and a GM. GMs start the registration phase by obtaining all needed information directly from GCKS. REG SA is used to protect the other SAs and cannot be set apart from them. It is important to note that no special communication protocol is strictly required here or, for that matter, no communication protocol at all, since a REG SA can even be set up in advance by using a smart card.
- REKEY SA is a multicast security association and it is used to create/renew an SA or to revoke access permission to a GM. It is started by the GCKS with no need of feedback from GMs sharing the same REKEY SA. Contrary to REG SA, it is not always present in GSA. In fact, the lifetime of a group may happen to be so short to make it useless.

- DATA SA (Data Security SA). As for the previous one, no negotiation is needed. It is created by the GCKS to protect the traffic of data flowing from the senders to receivers.

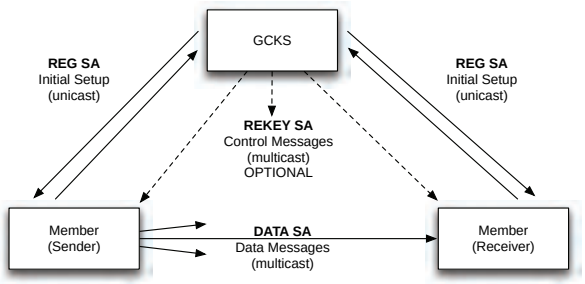


Fig. 2. Group Security Association (GSA) Structure (from (Hardjono et al., 2001))

By using the registration protocol each GM get the authorization and the authentication needed to access a group, to comply with its policies and to obtain its keys. There are two types of keys: Key Encryption Keys, KEK, needed to send keys in a secure way, and Traffic Encryption Keys, TEK, used to encrypt actual traffic. Also a Traffic-Protection Key (TPK) is used, which combines a TEK and a traffic integrity key. KEKs are relevant in a REKEY SA and TEKs/TPKs are relevant in a DATA SA.

2.2.2 Registration protocol

An entity desiring to become a GM will have to use a registration protocol on an unicast connection with the GCKS. The protocol involves mutual authentication between GCKS and the intended GM. When the authentication phase succeeds the GCKS supplies the joining member:

- with all the information needed to start a DATA SA (that is in the case the group security policy requires such a step right at registration and not, as the case may be, as a part of the rekey protocol);
- with all the information needed to start a REKEY SA (provided the group security policy requires a rekey protocol).

Obviously enough, the purpose of the registration protocol is to allow a secure (i.e. authenticated and confidential) transfer of the relevant information between the GCKS and the GM over a SA. Such an SA is called Registration SA. An analogous protocol is dedicated to the purpose of removing the REG SA (in case the GM has not chosen to do it itself).
The design of the registration protocol allows for a good level of flexibility and provides with the ability to support different scenarios. Any secure-channel protocol can be used to deliver the registration messages (e.g. IPsec or TLS). In fact this is what is done with tunneled GSAKMP (Harney et al., 2003). GDOI (see par. 2.4.2) uses IKE Phase 1 to get a secure channel to download REKEY and/or DATA SAs. Authenticated Diffie-Hellman exchanges of the type of IKE Phase 1 are used by protocols like MIKEY(see par. 2.4.3) and GSAKMP(see par. 2.4.1), although they are adapted to increase operations' efficiency.
If for some reason a GM loses the synch with the GSA, it might have to start over a registration with the GCKS. However, there are cases where a simpler method to return in synch may be available:

- the GM can open a plain TCP connection to GCKS and get the recent rekey messages. To open a TCP port to accept such requests might be seen as a dangerous exposition to DOS attacks. In fact, malicious re-synch requests could be an even more serious problem;
- the GCKS could publish the rekey messages on a public (e.g. web) site for the GM to download them from it.

It is desirable that the GCKS provides all three re-synching methods (i.e. new registration, TCP connection, public download).

2.2.3 Rekey protocols

In case of KEK/TPK expiration or group membership changes, the GCKS may update the REKEY SA. A REKEY SA is used to protect rekey messages.
The rekey protocol should possess the following properties:

- rekey information should reach GMs without excessive delays;
- the protocol must specify a way for the GM to contact the GCKS and proceed to a re-synch in case of keys expiration and lack of updates;
- the protocol must avoid implosion problems (see par. 2.2.4) and guarantee reliability in the delivery of rekey information.

The overall scalability of the group key management architecture relies heavily on the performances of the rekey protocol. Therefore scalability must be considered a prerequisite when designing a protocol intended to satisfy the above properties. Rekey protocol should use a scalable Group Key Management Algorithm (GKMA) to send the minimum possible number of keys in a rekey message. LKH (see par. 2.3), OFT (Balenson et al., 2000), Subset difference based schemes (Lotspiech et al., 2001) are examples of GKMA.
A rekey protocol has the following objectives:

- the synchronization of a GSA;
- privacy, authentication (symmetric or asymmetric), replay protection, DOS protection;
- efficient rekeying after changes in group membership or in case of keys (KEKs) expiration;
- allowing GMs to recovery synchronization with GSA;
- a reliable transport of rekey messages;
- good performances in throughput and latency;
- compatibility with multicast and multi-unicast.

A few major issues the design of the protocol must take into account are:

- messages format;
- reliable transport;
- feedback implosion;
- out-of-synch GSA recovery;
- the use of GKMA in rekey messages;
- GKMA interoperability.

2.2.4 Reliable transport of rekey messages

The reliable transport of rekey messages is a crucial point in the design of the protocol.
The content of rekey messages is typically made of KEKs, TPKs, REKEY SA and DATA SAs. Beyond confidentiality and authentication, the protocol must support protection against replay and DOS attacks. GCKS can send the messages to GMs by multicast or multi-unicast.
Confidentiality of rekey messages is obtained by encryption with the Group KEK. If a GKMA is used, the encryption of each part of the rekey message will be performed according to the GKMA specifications, by the pertinent KEKs.
For a GM to receive all intended data it is essential the GCKS is able to keep the SAs (DATA SA and REKEY SA) of such GM in synch. Therefore the reliability of the rekeying mechanism is a fundamental requirement. It can be achieved either by some procedure inherent to the algorithm or by choosing a reliable transport for the rekey messages.
The following solutions have been proposed:

- transmission of multiple copies of the rekey message. It must be recalled that a rekey message may span many IP packets;
- transport by an existing reliable multicast protocol or infrastructure;
- the use of Forward Error Correction (FEC) techniques (together with a feedback carried by NACKs) (Yang et al., 2001).

There is an ample choice of reliable multicast protocols that could be used in our context. While, as of this writing, none of them has started the standard track, a consensus has been reached within IETF on two protocols (Adamson et al., 2009) which are therefore likely to start the track not far from now.
Anyway, no particular reliable multicast protocol has been recommended by the IETF MSEC WG (*MSEC*, 2011) to guarantee reliability in group rekeying. In fact, the choice of the protocol could be subject to special application needs and to the operational environment. Nothing prevents, in the future, the standard use of a particular protocol for the needs of each class of applications.
A major problem arising when using a reliable multicast messaging protocol is implosion. Reliable multicast protocols often make use of ACKs or NACKs to get a feedback about the success of a particular transmission and to start a retransmission in case of failure. Any kind of condition leading to massive packet losses at the receivers can result in the transmission of NACKs from GMs to GCKS. The problem gets soon unmanageable with a large number of GMs. It is referred to as "feedback implosion".
Implosion has been one of the main areas of interest in the topic of reliable multicasting. Some of the solutions proposed to suppress or aggregate the feedback may be well suited in the context of group key management. To reduce the feedback, traffic members may be forced to wait for a random time before sending a negative feedback. During such a wait GMs may receive the needed updates and therefore avoid sending the feedback.
Feedback aggregation is another path followed by some reliable multicast protocols. In this specific domain, however, the concept has drawbacks related to authentication issues. The idea of local recovery (that is establishing local recovery servers to offload the main server) has the same type of problems since GMs should establish SAs with local servers. On the other hand, any subordinate GCKS or even any GM with adequate privileges may act as a local repair server and resend rekey messages.
The main purpose of a GKMA is to make rekeying scalable. Trying to manage a large group without an effective GKMA is plainly unfeasible.

The following points must be kept in mind when selecting a GKMA:

- *Protection against collusion*. GMs and non-members should not be able to join their knowledge in order to discover keys they are not allowed to know (according to GKMA keys' distribution rules).
- *Forward access control*. The GKMA must make sure a GM which has formally left the group is no longer able to re-join it.
- *Backward access control*. The GKMA must make sure when a GM joins the group it cannot decrypt past data.

In order to scale without difficulties GKMAs make generally use of a logical tree structure to organize keys. Obviously there are many ways to manage key trees and to identify a node within a key tree. Within each GKMA packet or at least during the initialization of a REKEY SA the following information has to be provided:

- GKMA name (e.g., LKH, OFT, Subset Difference);
- GKMA version number (implementation specific). Version may imply several things such as the degree of a key tree, proprietary enhancements, and qualify another field such as a key ID;
- number of keys or largest ID;
- version-specific data;
- per-key information:
  - key ID;
  - key lifetime (creation/expiration data);
  - encrypted key;
  - encryption key's ID (optional).

2.3 Logical key hierarchy: a Group Key Management Algorithm

To multicast in a secure an efficient way to a large group of users, a single TEK is generally used for encrypting traffic data. A crucial problem is represented by the users leaving or joining the group. To illustrate it we will refer to figure 3. In theory, for each single change in the users' base, the TEK (K_I in the figure) should be changed. Although grouping changes occurred in a given time interval and updating TEK once for all of them might alleviate the problem, the fact remains that a naive approach would mean transmitting the new TEK to each of the GMs encrypted with the unique key each GM has from its very inception (the KEK, Key Encryption Key, which the GCKS knows for all GMs, from K_A to K_H in figure 3). In other words, the GCKS should send to the group as many copy of the encrypted TEK as the number of GMs. That is an enormous traffic for large groups and, even worse, a constantly repeating one, considering the physiology of the "churn rate". The classical approach to attenuate the problems is that of Logical Key Hierarchy (Wallner et al., 1999) (Wong et al., 1998). Improvements of the LKH (Setia et al., 2000) (Rodeh et al., 1999) (Molva & Pannetrat, 1999) (Zhu et al., 2003) have been proposed with the main purpose of improving scalability of security group associations, in particular during the rekeying phase.
In the tree of figure 4 the circles are keys and the squares are users. The tree has been represented as a balanced binary tree for convenience although there is no particular restriction on its structure. The "leaf" keys are the keys each node has been assigned before

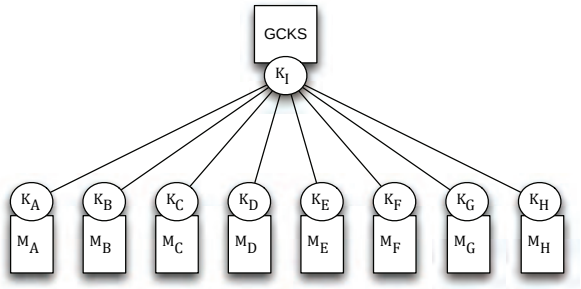


Fig. 3. N Root/Leaf Pairwise Keys

joining the group (e.g. by a smart card). The GCKS must know them beforehand for all nodes. K_O is called root key. By grouping the users in small groups and recursively grouping small groups in larger groups with a number of levels of grouping adapted to the expected total population of nodes, a significant reduction in rekeying traffic can be achieved. Let's see how the basic idea is a GM is supposed to know all the keys on the tree path from itself to the root. For example GM M_{16} must know, beyond its own KEK K_{16}, Key K_H, Key K_L, Key K_N, Key K_O (which is the TEK).
All the intermediate (auxiliary) keys (from K_A to K_N) do not need to be associated with any physical device.

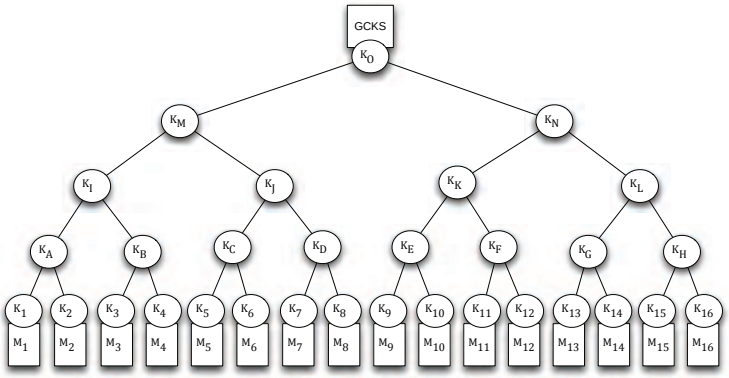


Fig. 4. LKH tree

2.3.1 Join operations

Suppose the user M_3 wishes to join the group. First it will be assigned, in one of many possible ways, its KEK K_3, known only to itself and to GCKS. Next, with some reasonable criterion, it will be associated to a subgroup (the subgroup with KEK K_B in the figure 4). At this point all the KEK from itself to the root (K_B, K_I, K_M, K_O) will have to be changed. They will be the new keys: K_B', K_I', K_M', K_O'. The new keys will have to be known from all the leaf nodes under them in the tree. To have K_B' known to all nodes under it (M_3 and M_4) GCKS will encrypt it with K_3 and K_4. To have K_I' known to all nodes under it (from M_1 to M_4) GCKS

will encrypt it with K_B' and K_A. To have K_M' known to all nodes under it (from M_1 to M_8) GCKS will encrypt it with K_I' and K_J. Finally to have K_O' (the TEK) known to all nodes GCKS will encrypt it with K_M' and K_N. The total number of encrypted keys in the rekey message will be $d * log_d(n)$ where n is the number of GMs and d is the degree of the key tree. By this scalable method all the GMs will be able to decrypt the new TEK using the auxiliary keys.

2.3.2 Leave operations

A typical situation is that of a GM leaving the group (e.g. a paying customer of a service willing to unsubscribe from it). The management of the "leave" operation is very similar to that of the "join" one. All the keys previously known to the leaving member will have to be changed in the same way as above.
For a fully populated tree of degree d and height h (where $h = log_d(n)$), the number of keys retransmitted when a member leaves the group is $d * h - 1$ and $d * h$ when a node joins the group (Wong et al., 1998); this compares favorably with the cost of n for a flat system.

2.4 Group key management protocols

A number of group key management protocol have been proposed. Within the multicast security workgroup there are three protocols related to group key management already on the standard track:

- Group Security Association Key Management Protocol (GSAKMP) (Harney et al., 2006). It is intended to be the generic key management protocol and defines methods for policy distribution, policy enforcement, key distribution, and key management.
- Group Domain of Interpretation protocol (GDOI) (Baugher et al., 2003). It uses the ISAKMP phase 1 negotiation as the authentication protocol and sets by it a secure connection between a receiver and the GCKS system. Phase 2 messages are defined within the protocol.
- Multimedia Inter KEYing (MIKEY) (Arkko et al., 2004). It is designed with real-time applications in mind.

2.4.1 Group Security Association Key Management Protocol

The following roles are specified in GSAKMP (Harney et al., 2006):

- Group owner (GO), it is in charge of the policies creation;
- Group Member (GM), it is the end-user (sender or receiver) of all security related procedures;
- Group Controller / Key Server (GCKS), it is responsible for the authentication of GMs, the enforcement of policies, the distribution and management of keys;
- S-GCKS, A GM which can act locally as GCKS when the functions of GCKS are distributed.

Operations of GSAKMP are described for three different scenarios: in the default one a single GM is the sender; in another one (support to which is mandatory) all GMs are potential senders. Support to the third scenario (only a few among the GMs are senders) is left as an option.
In order to enhance scalability, distributed operations are allowed through the set up of local GCKS (S-GCKS). An S-GCKS can provide a better management to its neighboring GMs (e.g. in corporate networks).

GSAKMP operates under the assumption there is at least one PKI (Public Key Infrastructure) for the group to trust. GSAKMP relies on such PKI while creating and verifying security policy rules. The public key of the GO must be known in advance to all GMs.
Upon creation of a new multicast group, the GO starts the process with the creation of a Policy Token (PT) describing the rules for access control and authorizations for that group. The token is signed by the GO. The token contains:

- identification for the PT and group;
- access control rules dictating who can have access to the group keys;
- authorization rules stating who can be a SGCKS;
- mechanisms for handling security, e.g. Security Protocol, Key Creation Method, Key encryption algorithm, Signature, etc.

After a PT is created and signed, it is sent by the GO to a potential GCKS. The latter verifies the signature and, based on the rules specified in the PT, decides whether it can act as a GCKS for the new group. If it can, then the new group is established and all GMs have to register with the GCKS (see Fig. 5). Upon receiving each registration request, the GCKS verifies the signature of the requesting GM and checks whether it is authorized to join the group. If the checks succeeds, the GM receives a "Key download" message. On its part a GM has to verify the GCKS has the authority to manage the group. Eventually, by using the information in the message, a GM can set up both REKEY and DATA SAs. If the GM has no need to send data to the group and it is planning to act as a receiver only, it will have no need to send a "Request to join" message and the "Key download" message is simply sent to the GM after its registration.

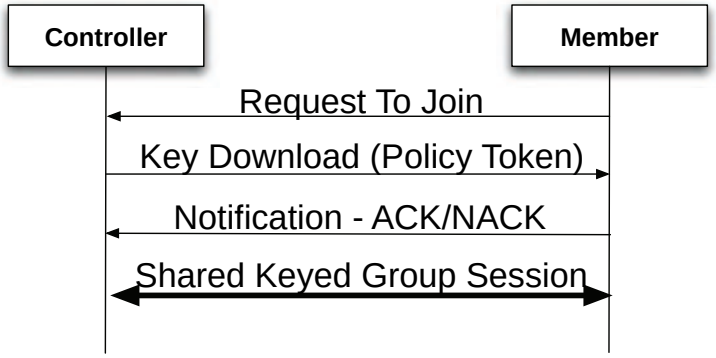


Fig. 5. GM registration in GSAKMP (from (Harney et al., 2006))

A rekeying is required whenever a GM joins or leaves the group and such operation will involve the GO. The latter is informed about node changes and reacts by creating a new PT. PTs must be pushed to the GCKS and the S-GCKS. Upon receiving a PT the GCKS nodes have to check whether the changes involve their own GMs. With no changes, the PT will be distributed according to the LKH by the use of the group key. If some of their nodes has changed than each client must receive the new PT and the only way to do it safely is to encrypt it according to the chosen GKMA and to send everything to every client.

2.4.2 Group Domain of Interpretation protocol

With reference to the ISAKMP (Maughan et al., 1998) terminology, GDOI (Baugher et al., 2003) specifies a domain of interpretation for group key management. While the ISAKMP specification is no longer current (being obsoleted by IKEv2 (Kaufman, 2005)), part of its framework is still used to detail the GDOI specifications.
The setup of secure connections is the result of a two-phases procedure in ISAKMP (and in GDOI). In our terms, the first phase allows to establish a secure unicast connection between the clients and the GCKS. Phase 2 is dedicated to rekeying and the creation of DATA SAs.
Identities of the involved entities are known (together with authorizations) to the GCKS from phase 1 but they can be integrated with certificates provided by the GO in phase 2.
Keys can be transmitted with two formats: GROUPKEY-PULL and GROUPKEY-PUSH. The first one is used by a GM in a client-server fashion to ask for TEKs, KEKs or KEKs arrays (with LKH) according to its needs.
On the other hand, GROUPKEY-PUSH is used by the GCKS when it needs to force the update of the REKEY SA or of the DATA SA.

2.4.3 Multimedia Inter KEYing protocol

The IETF WG has shown a definite interest in the protection of real-time traffic. In particular, the key exchange for SRTP (Secure Real-Time Transport Protocol) has been considered. The MIKEY protocol's design is the result of such focus. It is of use both in one-to-one and in one-to-many exchanges.
The MIKEY (Arkko et al., 2004) protocol specifies key management functionalities. It simplifies the architecture by allowing the sender to incorporate the functions of the GCKS. The Group Control part of the operations, the user's authentication, is performed throughout the course of the initial key exchange by signed messages. The protocol's emphasis on real-time data is represented by its efforts to provide a lower latency, its consideration for the usage over heterogeneous networks and for small groups' interactive exchanges.
The distribution of TEKs is based on the use of either shared keys (distributed in advance) or public keys encryption. With such methods the Traffic Encryption Key Generation Key (TGK) is a shared information between all hosts participating the session. Diffie-Hellman is used for one-to-one connections instead. In this case each client connects to the source (or to the separate GCKS node) and the TGK is different for each GM - GCKS pair.
To avoid the problems associated with the advance distribution of the shared keys, the use of certificates signed by a trusted CA can be preferred. Procedures for rekeying are not defined in MIKEY (the protocol is supposed to be run each time the rekeying is needed). MBMS (Multimedia Broadcast / Multicast Service) (*3GPP*, 2006) is an extension to the protocol designed to allow multicast rekeying in certain environments.

3. Reliable multicast

The topic of the reliable transport of multicast traffic has been already anticipated in par. 2.2.4, especially with reference to the classic problem of feedback implosion. Here we'll present three candidate protocols for the reliable multicast transport of encryption keys.
While protocols based on FEC do generally perform better (Setia et al., 2002) we wish to draw the attention to the fact that in a satellite environment, where the noise tends to be bursty and often a return channel is missing, a protocol simply transmitting multiple copies of the rekey messages might offer a viable alternative.

The three protocols we wish to present are Pragmatic General Multicast (PGM) (see par. 3.1), NACK-Oriented Reliable Multicast (NORM) (see par. 3.2) and our SRDP-Sign (see par. 3.3)

3.1 Pragmatic General Multicast (PGM)

"Pragmatic General Multicast (PGM) is a reliable transport protocol for applications that require ordered or unordered, duplicate-free, multicast data delivery from multiple sources to multiple receivers" (Speakman et al., 2001).
The protocol, developed by a large team of researchers, has the RFC status of "Experimental" as of this writing. Its design puts emphasis on simplicity and does not support much more than the essential capabilities for this class of protocols. Its main concern is the reduction of the repair traffic (driven from NACK implosion or caused by the useless feeding of redundancy to receivers not needing it).
For better operation PGM needs support from the routers crossed by the multicast traffic. That is each router should run PGM-aware software (or firmware) extensions (or, put in different terms, be a PGM NE or PGM-capable Network Elements). At any rate, the protocol can also work, although less efficiently, when some or all of the routers are unaware of it.
PGM runs over the standard IP multicast. As customary with that protocol, GMs can join and leave the group without notifying the source. The only guarantee for a GM is that, once joined the group, it will receive the data with no errors. Any GM can become an independent sender for the group it belongs to and its identification is given by a Transport Session Identifier that no one else can share. PGM is flexible enough to support many different types of applications "as disparate as stock and news updates, data conferencing, low-delay real-time video transfer, and bulk data transfer". Other supplementary options include Designated Local Repairer (DLR) support, fragmentation, late joining, and Forward Error Correction (FEC).
The protocol gets its feedback about the transmission results in the form of NACKs. The potential danger of a NACK implosion is reduced by NACK suppression and NACK aggregation in PGM NE routers (see below).
PGM define the following type of packets:

- ODATA, the Original copy of the transmitted DATA;
- NAK, a Negative AcKnowledgement issued when the receiver realizes a packet is missing in the sequence it received;
- NCF, NAK Confirmation;
- RDATA, a Retransmission of the original DATA;
- SPM, Source Path Message.

3.1.1 PGM transmit window

Mimicking the strategies followed by unicast reliable protocols, PGM keeps a sliding window within which to transmit data. The absence of data allowing to accurately shift the left side of the window leads to the use of a few expedients (based on fixed time waits, on a given period without received NAKs etc.).

3.1.2 PGM tree

To forward data to the intended recipients, PGM builds its own distribution tree (PGM tree) which is identical to the distribution tree natively built by the routers supporting the

IP multicast protocol when all such routers are PGM NE. More generally, PGM builds the distribution tree (the "overlay network") over the original IP multicast tree by having the sender transmitting Source Path Messages (SPM) to the group at regular intervals during the data transfer.

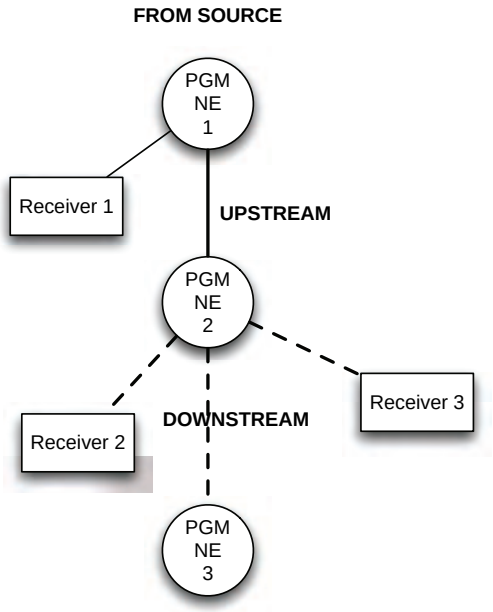


Fig. 6. PGM upstream/downstream attributes for router *PGMNE*2

SPMs are modified at each crossed PGM NE (see Fig. 6). When it reaches a PGM NE an SPM packet contains the address of the PGM NE it comes from. Before forwarding the packet, a PGM NE substitutes its own address to that address so that every PGM NE will always know the address of its upstream closest PGM NE (Gemmell et al., 2003).
When ODATA packets start to flow and a host detects a missing packet, after a random backoff time it sends a NAK to the upstream PGM NE it knows because of the above procedure. On its turn, the latter:

- sends back a multicast NCF packet by the interface that received the NAK;
- forwards to the PGM NE upstream the same NAK packet it received and receives an NCF from it.

The process continues upstream until the source or a DLR is reached. When the NAK reaches the source or a DLR, these may re-send the lost packet downstream to the multicast group, either in the original form or by some FEC encoding.

3.1.3 Local repair: DLR

If no DLR were present, all the repair packets had to be re-sent by the source. The presence of DLRs helps to reduce the outgoing traffic at the source and to limit it to the multicast tree

portion downstream the DLR. It also helps to speed the repair procedure. DLR can announce their presence so that PGM NEs can direct NAKs to them rather than to the source.

3.1.4 PGM with non-PGM-aware routers

PGM can operate even when all routers are not PGM-aware. Of course, with no PGM NE, many of the features of the protocol are lost. For example, each NAK packet will be multicast in the usual way, without the suppression of duplicated NAKs. It will be also impossible to perform efficient repairs, since RDATA packets will be transmitted again and again, no matter how many GM have requested the same packet. The protocol performances will however improve as the number of PGM NE increases.

3.1.5 Congestion control

Congestion control in PGM is performed by limiting the transmission rate at the source. Such limitation is based on the feedback received both from receivers and from PGM NEs. The feedback is given by appending special "report" fields at the end of a NAK packet. The reports communicate the "load" measured by receivers, in the form of packet loss rates, or by PGM NEs, in the form of packet drop rates.
The feedback can be of three different types:

- worst link load as measured by the PGM NEs;
- worst end-to-end path load as measured by the PGM NEs;
- worst end-to-end path load as reported by receivers.

Although congestion control is mandatory, there is no specification of how this data should be used to adjust the sending rate and the choice is left to the implementation (Gemmell et al., 2003).
An extension of the protocol aimed at congestion control has been proposed with PGM CC (Rizzo, 2000). PGMCC is described as "single rate" in that "all receivers gets the same rate and the source adapts to the slowest receiver" and "TCP-friendly" in that the sender tries not to transmit faster than the rate allowed by TCP specifications with the slowest receiver. The protocol adopts a window-based, TCP-like control loop.

3.2 Negative-ACKnowledgment (NACK) Oriented Reliable Multicast (NORM)

According to (Adamson et al., 2009) The Negative-ACKnowledgment (NACK) Oriented Reliable Multicast (NORM) protocol "can provide reliable transport of data from one or more senders to a group of receivers over an IP multicast network". Efficiency, scalability, support for heterogeneous IP networks and for bulk transfers are said to be the goals for the protocol's design. Another interesting target for the protocol is to provide "support for distributed multicast session participation with minimal coordination among senders and receivers". Starting with (Adamson et al., 2009), NORM is on the IETF standard track. In (Adamson et al., 2009) message types and protocol operation are explained in detail. (Adamson et al., 2008) discusses goals and challenges for reliable multicast protocols in general, defines building blocks to address these goals and gives a rationale for the development of NORM.
End-to-end reliable transport of application data is based on the transmission of NACKs from the receivers to initiate repair transmissions from the senders. Variability in network conditions is taken care of by using adaptive timers for the protocol operations. The protocol is designed to offer its transport services to higher levels in a number of ways in order to satisfy the needs of different applications.

NORM uses FEC in various ways. It can use it both in the encoding of the original stream and in the repair traffic sent to the group in response to NACKs from the receivers (proactive/reactive FEC). In general, the more FEC redundancy is put in the original stream the less NACKs will be received.
Most of the potential limitations of the scalability of the protocol come from the negative feedback generated from receivers. NORM uses a probabilistic suppression of the feedback based on exponentially distributed random backoff timers. To avoid disturbing the operations of concurrent transport protocols (e.g. TCP) a congestion control scheme is specified, although alternative choices are left to the implementers.

3.2.1 NORM building blocks

NORM is conceptually divided in three main blocks:

- NORM Sender Transmission, which takes care of data transmissions and reception of feedback (NACK) messages;
- NORM Repair Process, which processes the feedback information and tells the first block what to retransmit;
- NORM Receiver Join Policies, relates to policies and procedures involving receivers admission to the data distribution. While receiver joins are generally unconstrained, a sender might wish to limitate the number of potential NACK senders in various ways.

Other functions (congestion control, error correction etc.) are delegated to further modules.

3.2.2 NORM operations

Messages in NORM are basically divided in sender messages and receiver messages: NORM_CMD, NORM_INFO, and NORM_DATA message types are generated by senders of data content NORM_NACK and NORM_ACK messages generated by receivers within a session.
The NORM_DATA messages are used by senders to transmit application data and FEC encoded repair packets while NORM_NACK messages are generated by receivers to selectively request the retransmission of missing content. NORM_CMD messages are used for various management and probing tasks while NORM_ACK is the acknowledgement message for such commands.
As it is customary in this class of protocols, the receivers schedule random backoff timeouts before sending a NORM_NACK message, which could be repeated if the hoped-for repair has not come. The sender doesn't react to single NACK messages but rather tries to aggregate a number of them to decide how much to "rewind" its transmission. When it deems the rewind to be sufficient, it proceeds to the actual retransmission.

3.2.3 Congestion control

Congestion control for NORM is described in (Adamson et al., 2009). It is an adaptation of the TCP-Friendly Multicast Congestion Control (TFMCC) described in (Widmer & Handley, 2006). It is essentialy based on a rate-control approach rather than on the control of the transmission window. The protocol specification leave, freedom to opt for a window-based approach like that of PGMCC.

3.3 Satellite Reliable Distribution Protocol for Signaling (SDRP-Sign)

The Satellite Reliable Distribution Protocol (SRDP) protocols (Tommasi et al., 2006) (Tommasi et al., 2003) are reliable transport protocols designed with special attention to the use in satellite applications. SRDP-Sign can be seen as an extension of the original SRDP protocol.
The two protocols use the same UDP port and implement two different types of transports: SRDP-Bulk and SRDP-Sign. The first one is FEC-based and it is used for bulk data transfers. The second one is of the multi-send type (Tommasi et al., 2008) and has been originally designed for signaling. Despite the original design focus of SRDP-Sign has been the use with short messages (e.g. in multicast control applications) or more generally, with signaling, its relative immunity to burst errors makes it interesting in the context we are examining (Tommasi et al., 2009). One more reason of interest for the protocol is its capability to transmit information to users who can receive information from a satellite but do not possess a return channel. However reliability of transmissions cannot be assured with this subset of users. For all other users, SRDP-Sign is capable of accepting a return feedback both via satellite and terrestrially. The SRDP-Sign protocol is also optimized for a high number of users.

3.3.1 SRDP-Sign: Requirements and architecture

The requirements of the SRDP-Sign protocol are:

- high degree of scalability;
- fast delivery of messages;
- high resistance to burst errors;
- high probability of complete delivery of transmitted data for all users;
- guarantee of complete delivery of transmitted data for users with a return channel;
- limited use of control messaging between sender and receivers.

The objective of each session of the SRDP-Sign is to transmit messages M to R users. Reliability is ensured via transmission of N multiple copies of the messages (Setia et al., 2002). The SRDP-Sign protocol manages the transmission of a single message (SRDP-sign session). The protocol can transmit multiple simultaneous sessions, that is the transmission of the copies of two different sessions to be interlaced. Bundling more messages within a packet is not permitted (see Fig. 7). This preserves the simplicity of packet management.

3.3.2 SRDP-Sign operations

SRDP-Sign ensures reliability of the transmission of a message M, replicating it in N packets. P_i is the i-th reply in the N packets sequence. The delivery of a message is organized in two phases. During the Winding phase, the sender is restricted to replicate the message and there is no control. During the Unwinding phase, the sender makes an estimate of the number of receivers who did not receive the packet during the Winding Phase through a scheme of suppression of the number of receivers.
SRDP-Sign messages can be of the following types:

- DATA, a data packet;
- ABORT, sent to interrupt an ongoing session;
- STAT_REQ, a request to the receivers of a feedback about the correct reception of the message;

- STAT_REP, the answer to a STAT_REQ.

During the Winding Phase the sender transmits N copies (DATA) of the message M. In case of a correct reception of a packet, a receiver ignores all other packets of that message. The replicated packets are transmitted with exponential times that reduces the effects of the potential burst errors (Tommasi et al., 2003).
During the Unwinding Phase the sender multicasts a STAT_REQ to check whether the R receivers have received the message during the previous phase. This request is processed by the receivers through an algorithm of probabilistic suppression of the NACKs. This behavior has a high level of scalability (Nonnenmacher & Biersack, 1998). If a receiver sees the request and has not received the message, then the probabilistic suppression comes into play and if it results in an authorization to proceed, the receiver sends a STAT_REP to the source. A session finishes when the last STAT_REQ message in the sequence (see below) gets no answer. On the other hand, as soon as a STAT_REP message is received by the source, it stops the sending of STAT_REQ messages and proceeds to a new Winding Phase.
The ABORT message, when needed, is also repeated in a fixed way (exactly ten times at regular intervals).

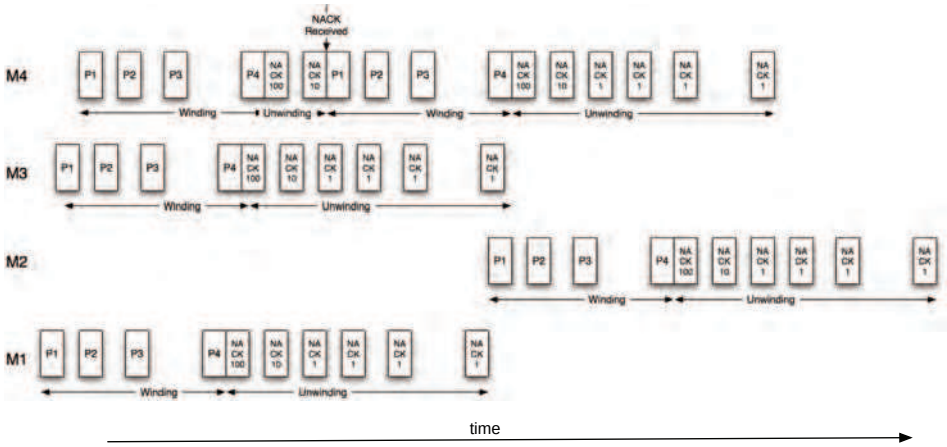


Fig. 7. Transmission of multiple messages (M1 and M2 cannot be interlaced)

3.3.3 Scalable Feedback Suppression (SFS)

Ideally, after transmitting a packet, the sender would like to receive boolean information (yes/no) related to the correct reception from all the receivers. In order to prevent NACK implosion, the Scalable Feedback Suppression (SFS) algorithm causes the random selection of a subset of all the receivers. Such subset is allowed to transmit a negative feedback to the sender. Obviously, only the receivers with a return channel can participate to the selection. The algorithm begins with the selection of an high number H (which represents a very rough and imprecise estimate of the number of potential receivers). The first STAT_REQ transmission is executed and the value probability P_H of 10^r (where $r = -\lceil log_{10} H \rceil$) is included in the message. A receiver that did not receive the original message is authorized to reply only if it generates a pseudorandom value (between 0 and 1) and such value is lower

than P_H. If the sender receives even a single NACK, it will abort the Unwinding phase and will re-initiate the Winding Phase (see Fig. 7). If, on the contrary, it does not receive any NACK, the sender iterates the STAT_REQ transmission putting in the message a value of P_H of 10^{r+1}. If the sender does not receive any NACK it increases the transmitted P_H value until it reaches 1. At this point it determines the message has been correctly received by everyone.

4. Performance evaluation

We set up an experimental network to test the performances of the PGM, NORM and SRDP-Sign protocols in different scenarios. Given the scope of the present chapter only a sample of the tests results are reported. The complete results will be the object of a forthcoming publication.
For PGM we selected OpenPGM, an open source implementation available at (*OpenPGM*, 2011). OpenPGM it is not yet a final release. The source code of a NORM implementation is available at (*NORM*, 2007).
The network topology we employed in our test is characterized by an (hybrid) asymmetric connectivity where a single sender is connected directly to the satellite uplink (1Mbit/s) and a small multicast group of receivers has a unicast terrestrial return path to the sender. In this topology, receivers have no access to the satellite uplink but, as it is usually the case, they can receive from the downlink either through a satellite receiver connected to their LAN or by an on-board card (see Fig. 8). The round trip time is about 600ms. We also considered a scenario in which there is no return path to the sender and therefore no kind of feedback is sent by the receivers.
We evaluated the performances of the protocols for various packet loss percentages at the receivers caused by the satellite link. The test is conducted in an homogeneous network with all receivers experiencing the same percentage of independent losses. The packet loss is emulated using Dummynet (Carbone & Rizzo, 2010).

Protocol	Parameter and value	Meaning
NORM	blocksize=64	Number of source packets per FEC coding block.
NORM	parity=32	Number of FEC parity packets.
NORM	auto=32	Number of proactively parity packets.
NORM	unicastNacks	NACK sent in unicast.
OpenPGM	Transmission Group size = 64	Number of source packets per FEC coding block.
OpenPGM	Proactively parity = 32	Number of proactively parity packets.
SRDP-Sign	N=3	number of replies for each message.

Table 1. Configuration Parameters

Protocols configuration parameters used for the present selection of the results are shown in table 1. To put the three protocols on a par, no PGM-aware router has been employed.
We calculated the Average Key Delivery Ratio (AKDR) and the data overhead to evaluate the performances of the above reliable multicast protocols. AKDR is the ratio {number of keys received}/{number of keys transmitted} averaged over all multicast group members. Data overhead is the ratio {total amount of data transmitted}/{net amount of keys data transmitted}.
Fig. 9a shows the results of the tests when a return channel is available, that is the receivers are able to send a feedback to the sender. Fig. 9b shows the results with no return channel available. Despite its simplicity and limited efficiency, it is interesting to note the fairly good

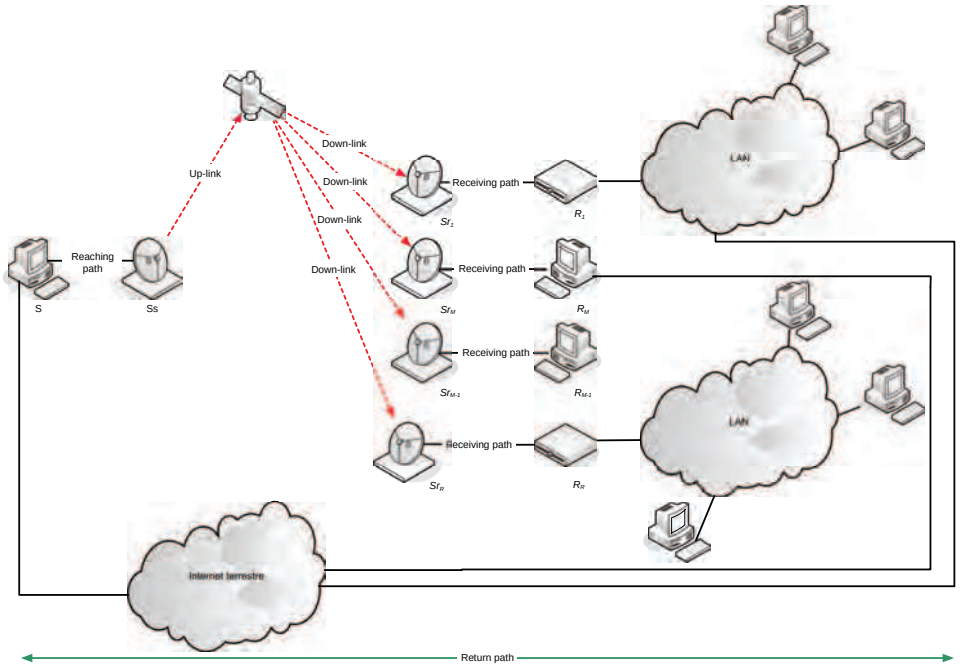
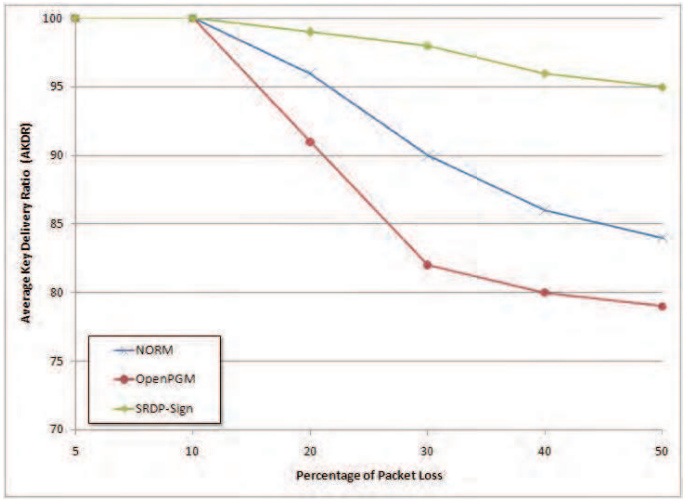


Fig. 8. Test network topology

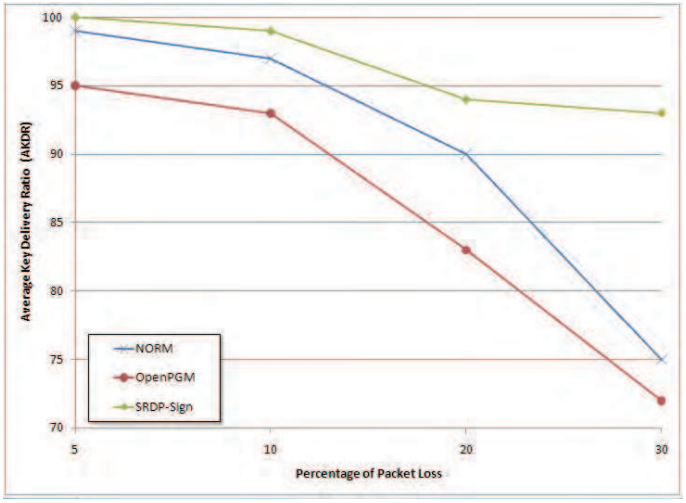
performances of SRDP-Sign with high packet losses. Increasing redundancy to compensate for high error rates, generally tends to favour the efficiency of FEC based protocols as compared to that of the replica-based ones. However, as our preliminary results suggest, bursty environments (like the satellite ones) tend to level the comparison.
Fig.10 shows a somewhat expected outcome: since SRDP-Sign sends a fixed number of replicas in the winding phase, no matter how much noisy the transmission is, its overhead is by far the largest at low levels of packet losses. On the other hand, when the level of losses increases, also PGM and NORM are forced to retransmit packets, ending up in reaching approximately the same amount of overhead as SRDP-Sign.

5. Conclusions

This chapter introduced the framework and the protocols IETF specified for a multicast security architecture. Three different protocols for key exchange (registration and rekeying), have been presented: GSAKMP, MIKEY, and GDOI. They were developed with different settings in mind, since a single protocol was not believed to be able to support all the typical scenarios in multicast security. LKH is used to allow the rekeying phases to efficiently scale over a large number of users. If the keys are sent via multicast, which is common for large groups and unavoidable with satellites, a reliable multicast transport is required. Three protocols offering such service have been considered: PGM, NORM and SRDP-Sign. The first two of them have been debated within the IETF MSEC WG. The third one was originally conceived for the utilization in multicast signaling (i.e. the reliable delivery of short control



(a) Receivers with a return path



(b) Receivers without return path

Fig. 9. Average Key Delivery Ratio

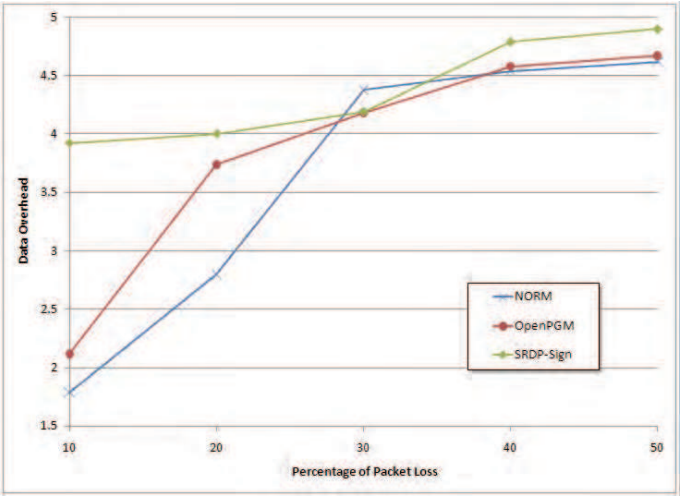


Fig. 10. Data Overhead (receivers with a return path)

messages). However its promising behavior in a satellite environment has prompted to test it in the present context. The preliminary results suggest that, while PGP and NORM do generally perform better, high levels of packet losses (which are typical of the bursty disruption of satellites transmissions) tend to put the simpler approach of SRDP-Sign in a more favorable position.

6. References

3GPP (2006). Security of Multimedia Broadcast / Multicast Service (MBMS). Technical specification TS 33.246.

Adamson, B., Bormann, C., Handley, M. & Macker, J. (2008). Multicast Negative-Acknowledgment (NACK) Building Blocks, RFC 5401. Obsoletes RFC3941.

Adamson, B., Bormann, C., Handley, M. & Macker, J. (2009). NACK-Oriented Reliable Multicast (NORM) Transport Protocol, RFC 5740. Obsoletes RFC3940.

Arkko, J., Carrara, E., Lindholm, F., Naslund, M. & Norrman, K. (2004). MIKEY: Multimedia Internet KEYing, RFC 3830. Updated by RFC 4738.

Arslan, M. G. & Alagöz, F. (2006). Security issues and performance study of key management techniques over satellite links, *Proceedings of CAMAD*.

Balenson, D., McGrew, D. & Sherman, A. (2000). Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization, Internet Draft (work in progress), draft-irtf-smug-groupkeymgmt-oft-00.

Baugher, M., Canetti, R., Dondeti, L. & Lindholm, F. (2005). Multicast Security (MSEC) Group Key Management Architecture, RFC 4046.

Baugher, M., Weis, B., Hardjono, T. & Harney, H. (2003). The Group Domain of Interpretation, RFC 3547.

Carbone, M. & Rizzo, L. (2010). Dummynet revisited, *Computer Communication Review* pp. 12–20.

Cruickshank, H., Evans, B., Mertzanis, I., Leitold, H. & Posch, R. (1998). Securing multimedia services over satellite atm networks, *International Journal of Satellite Communications* pp. 169–208.

Deering, S. (1989). Host extensions for IP multicasting, RFC 1112. Obsoletes RFC 988, RFC 1054, Updated by RFC 2236.

Deering, S. (1991). Multicast Routing in a Datagram Internetwork, Ph.D. Thesis.

Diot, C., Levine, B., Lyles, B., Kassem, H. & Balensiefen, D. (2000). Deployment issues for the ip multicast service and architecture, *IEEE Network magazine special issue on Multicasting* pp. 78–88.

Dondeti, L. R., Mukherjee, S. & Samal, A. (1999). Survey and comparison of secure group communication protocols, *Technical report*, University of Nebraska-Lincoln.

Eriksson, H. (1994). Mbone: the multicast backbone, *Communications of The ACM* pp. 54–60.

Eskicioglu, A. M. (2003). Multimedia security in group communications: Recent progress in management, authentication, and watermarking, *ACM Multimedia Systems Journal* pp. 239–248.

Gemmell, J., Montgomery, T., Speakman, T., Bhaskar, N. & Crowcroft, J. (2003). The pgm reliable multicast protocol, *Technical report*, IEEE Network.

Hardjono, T., Baugher, M. & Harney, H. (2001). Group security association (gsa) management in ip multicast, *Proceedings of SEC*.

Hardjono, T. & Weis, B. (2004). The Multicast Group Security Architecture, RFC 3740.

Harkins, D. & Carrel, D. (1998). The Internet Key Exchange (IKE), RFC 2409. Obsoleted by RFC 4306, updated by RFC 4109.

Harney, H., Colegrove, A., Harder, E., Meth, U., & Fleischer, R. (2003). Tunneled Group Secure Association Key Management Protocol, Internet Draft (work in progress), draft-ietf-msec-tgsakmp-00.

Harney, H., Meth, U., Colegrove, A. & Gross, G. (2006). GSAKMP: Group Secure Association Key Management Protocol, RFC 4535.

Howarth, M. P., Iyengar, S., Sun, Z. & Cruickshank, H. (2004). Dynamics of key management in secure satellite multicast., *IEEE Journal on Selected Areas in Communications* pp. 308–319.

ISO 7498-2 (1989). Information processing systems, Open Systems Interconnection Basic Reference Model, Part 2: Security Architecture, Internation Organization for Standardization.

Jokela, P. (2006). Key management in ip multicast.
URL: *http://www.tcs.hut.fi/Studies/T-79.7001/2006AUT/seminar-papers/Jokela-paper-final.pdf*

Kaufman, C. (2005). Internet Key Exchange (IKEv2) Protocol, RFC 4306. Obsoletes RFC2407, RFC2408, RFC2409, Obsoleted by RFC5996, Updated by RFC5282.

Kent, S. & Atkinson, R. (1998). Security Architecture for the Internet Protocol, RFC 2401. Obsoletes RFC1825, Obsoleted by RFC4301, Updated by RFC3168.

Lotspiech, J., Naor, M. & Naor, D. (2001). Subset-Difference Based Key Management for Secure Multicast, Internet Draft (work in progress), draft-irtf-smug-subsetdifference-00.

Mah, F. (2004). Group key management in multicast security.
URL: *www.tml.tkk.fi/Publications/C/18/mah.pdf*

Maughan, D., Schertler, M., Schneider, M. & Turner, J. (1998). Internet Security Association and Key Management Protocol (ISAKMP), RFC 2408. Obsoleted by RFC 4306.

MBONED (2011). IETF MBONED Working Group.
URL: *http://datatracker.ietf.org/wg/mboned/charter/*

Molva, R. & Pannetrat, A. (1999). Scalable multicast security in dynamic groups., *Proceeding of the 6th ACM Conference on Computer and Communications Security*.

MSEC (2011). IETF Multicast Security Charter (MSec).
URL: *http://datatracker.ietf.org/wg/msec/charter/*

Nonnenmacher, J. & Biersack, E. (1998). Optimal multicast feedback, *Proceedings of INFOCOM*.

NORM (2007). NORM implementation Web Site.
URL: *http://downloads.pf.itd.nrl.navy.mil/norm/*

Noubir, G. & Allmen, L. V. (1999). Security issues in internet protocols over satellite links, *Proceedings of IEEE Vehicular Technology Conference*.

OpenPGM (2011). OpenPGM implementation Web Site.
URL: *http://code.google.com/p/openpgm/*

Orman, H. (1998). The OAKLEY Key Determination Protocol, RFC 2412.

Rafaeli, S. & Hutchison, D. (2003). A survey of key management for secure group communication, *ACM Computing Surveys* pp. 309–329.

Rizzo, L. (1997). Effective erasure codes for reliable computer communication protocols, *SIGCOMM Comput. Commun. Rev.* pp. 24–36.

Rizzo, L. (2000). pgmcc: a tcp-friendly single-rate multicast congestion control scheme, *Proceedings of ACM SIGCOMM*.

Rodeh, O., Birman, K. & Dolev, D. (1999). Optimized group rekey for group communication systems, *Proceedings of ISOC Network and Distributed Systems Security Symposium*.

Sardella, A. (2005). Video Distribution in a Hybrid Multicast-Unicast World, Juniper networks.
URL: *http://www.juniper.net/solutions/literature/white_papers/200107.pdf*

Semeria, C. & Maufe, T. (1997). Introduction to IP Multicast Routing.
URL: *http://www4.ncsu.edu/∼rhee/export/papers/multi1.pdf*

Setia, S., Koussih, S., Jajodia, S. & Harder, E. (2000). Kronos: A scalable group re-keying approach for secure multicast, *Proceedings of IEEE Symposium on Security and Privacy*.

Setia, S., Zhu, S. & Jajodia, S. (2002). A comparative performance analysis of reliable group rekey transport protocols for secure multicast, *Proceedings of Performance Evaluation, special issue on the Proceedings of Performance 2002*.

Speakman, T., Crowcroft, J., Gemmell, J., Farinacci, D., Lin, S., Leshchiner, D., Luby, M., Montgomery, T., Rizzo, L., Tweedly, A., Bhaskar, N., Edmonstone, R., Sumanasekera, R. & Vicisano, L. (2001). PGM Reliable Transport Protocol Specification, RFC 3208.

Tommasi, F. & C.Melle (2011). Large-scale terrestrial relaying of satellite broadcasted real-time multimedia streams, *International Journal of Computer Networks & Communications (IJCNC)* .

Tommasi, F., Molendini, S. & Scialpi, E. (2008). Srdp-sign: a reliable multicast protocol for signaling., *Proceedings of NOMS'08*.

Tommasi, F., Molendini, S. & Scialpi, E. (2009). Reliable key distribution for secure multicast by srdp-sign, *Proceedings of AFIN*.

Tommasi, F., Molendini, S., Scialpi, E. & C.Melle (2010). Charms: Cooperative hybrid architecture for relaying multicast satellite streams to sites without a satellite receiver, *Proceedings of IEEE WCNIS*.

Tommasi, F., Molendini, S. & Tricco, A. (2003). Design of the satellite reliable distribution protocol (srdp), *Proceedings of IEEE Globecom*.

Tommasi, F., Molendini, S. & Tricco, A. (2006). The satellite reliable distribution protocol (srdp), *JCOMSS - Journal of Communications Software and Systems* pp. 152–160.

Wallner, D., Harder, E. & Agee, R. (1999). Key Management for Multicast: Issues and Architectures, RFC 2627.

Widmer, J. & Handley, M. (2006). TCP-Friendly Multicast Congestion Control (TFMCC): Protocol Specification, RFC 4654.

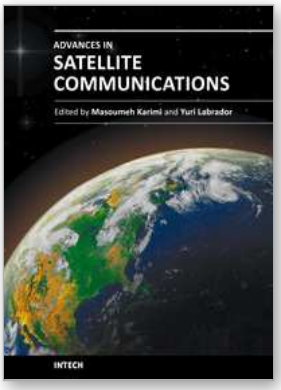
Wong, C. K., Gouda, M. & Lam, S. S. (1998). Secure group communications using key graphs, *Proceedings of IEEE/ACM Transactions on Networking*.

Wong, C. & Lam, S. (2000). Keystone: a group key management system, *Proceedings of International Conference in Telecommunications*.

Yang, Y., Li, X., Zhang, X., & Lam, S. (2001). Reliable group rekeying: a performance analysis, *Proceedings of SIGCOMM*.

Zhang, X. B., Lam, S. S., Lee, D.-Y. & Yang, Y. R. (2003). Protocol design for scalable and reliable group rekeying, *Proceedings of IEEE/ACM Transactions on Networking*.

Zhu, S., Setia, S. & Jajodia, S. (2003). Performance optimizations for group key management schemes for secure multicast, *Proceedings of the 23rd International Conference on Distributed Computing Systems*.



Advances in Satellite Communications
Edited by Dr. Masoumeh Karimi

ISBN 978-953-307-562-4
Hard cover, 194 pages
Publisher InTech
Published online 27, July, 2011
Published in print edition July, 2011

Satellite communication systems are now a major part of most telecommunications networks as well as our everyday lives through mobile personal communication systems and broadcast television. A sound understanding of such systems is therefore important for a wide range of system designers, engineers and users. This book provides a comprehensive review of some applications that have driven this growth. It analyzes various aspects of Satellite Communications from Antenna design, Real Time applications, Quality of Service (QoS), Atmospheric effects, Hybrid Satellite-Terrestrial Networks, Sensor Networks and High Capacity Satellite Links. It is the desire of the authors that the topics selected for the book can give the reader an overview of the current trends in Satellite Systems, and also an in depth analysis of the technical aspects of each one of them.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821