COST Action 272

"Packet-Oriented Service Delivery via Satellite"

Security systems for multicast data transfer over satellite

TD-02-024-P

Michael P. Howarth, Sunil Iyengar, Haitham Cruickshank, Zhili Sun
Centre for Communication Systems Research
University of Surrey, Guildford GU2 7XH, United Kingdom

Abstract

Security is an important concern in today's information age, and particularly so in satellite systems where eavesdropping can be easily performed. This paper describes two examples of end-to-end security systems that have been developed within the EU 5th Framework GEOCAST project. The first, SAT-RMTP, provides secure multicast file transfer using a file transfer protocol that is optimised for satellite environments. The second example is optimised for secure multicast data streaming: this involves the integration of GSAKMP, a key management protocol, with LKH, a key management technique that is scalable to the large number of receivers that are expected in satellite multicast.

1. Introduction

Satellite-based broadband IP networks have the potential to deliver multicast services cost-effectively. However, satellites present some significant security challenges:

- Eavesdropping and active intrusion are much easier than in terrestrial fixed or mobile networks because of the broadcast nature of satellites;
- Satellite systems are resource-constrained, particularly in the areas of limited transmission power (and thus channel capacity), and limited processing and switching capability for satellites with on-board processing;
- Satellite channels experience high bit error rates, which can result in packet loss and the loss of security synchronisation.

Security systems for satellite data thus have to take account of these limitations, in particular the need for confidentiality and the requirement to use satellite resources efficiently. Geostationary satellites also suffer from a long propagation delay, and security systems must therefore add only minimal delays to traffic.

The EU IST GEOCAST project is considering the issues associated with multicast over geostationary satellite; one of the issues being addressed is end-to-end security of data, and this paper presents two examples of end-to-end security systems that have been developed within the project. These are secure multicast file transfer and secure multicast data streaming.

This paper is organised as follows. Section 2 provides an overview of the GEOCAST project. Section 3 introduces some definitions associated with electronic security. Section 4 describes the secure multicast file transfer protocol, SAT-RMTP. Section 5 then proceeds to describe two technologies that support secure multicast data streaming: these are GSAKMP and LKH.

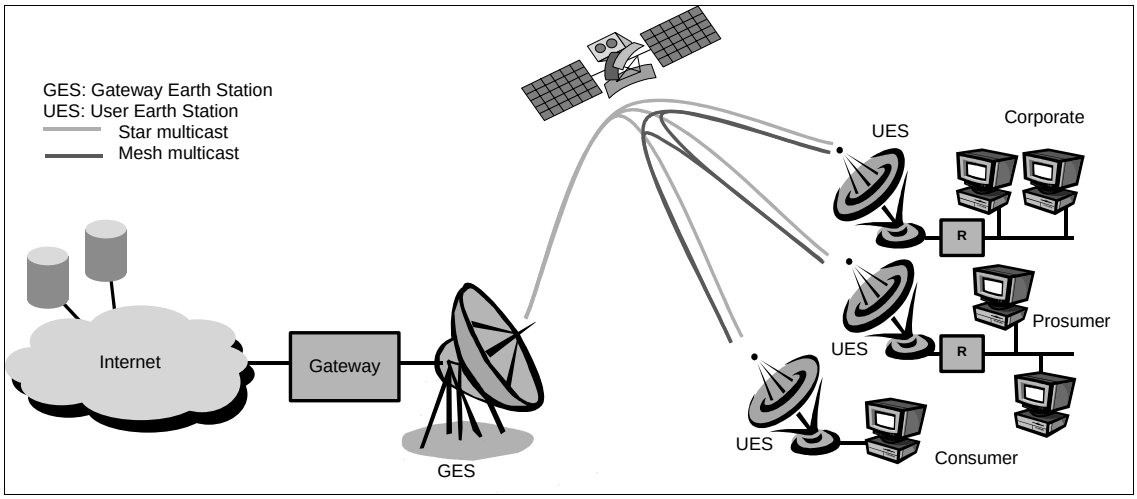

Figure 1: GEOCAST project

2. The GEOCAST project

The GEOCAST project, funded within the EU 5th Framework IST programme, is looking at the issues associated with multicast over geostationary satellite, and aims to develop a satellite system capable of supporting multiple data types (data, video, audio), including multicast applications, on a single network technology (Figure 1) [Geoc].

One of the areas considered in GEOCAST is security: the objective of this work is to investigate and examine issues related to satellite multicast security. To this end, the security component has focused on two areas:

- Core security: authentication of entities (primarily the user earth stations) to the satellite prior to assigning satellite resources (frequency allocation and channel capacity); and data link layer encryption of traffic on the satellite uplinks and downlinks to provide privacy.
- End-to-end security: security, primarily for privacy, provided typically at the network, transport or application layer.

Issues associated with multicast data transfer arise in the end-to-end security, and it is these issues that have been explored in the work described in Sections 4 and 5 of this paper.

3. Electronic security

Algorithms used in electronic security can be divided into two groups. The first group is secret key algorithms, where the encryption key and the decryption key are the same or can be calculated from one another. The second group is public key algorithms, which consist of two matched keys A and B of the same length. Here, a message encoded with one key A can only be decoded with the other key B, and similarly a message encoded with key B can only be decoded with key A. Security systems can be implemented using X.509 digital certificates: these use public key algorithms, and an X.509 certificate contains the name of the owner and their public key, key A of the public key algorithm. The certificate owner maintains key B as a secret at all times and never divulges it to any other entity.

Electronic security can be divided into four principal services: authentication, confidentiality, integrity and non-repudiation, defined as follows:

- *Authentication* is a service used to verify the identity of entities involved in a communication. This is generally achieved by the entities sharing a secret (for example a PIN number stored in hardware) or by a digital certificate.
- *Confidentiality* ensures that only the intended parties have access to communications between two or more entities. This is generally achieved by encryption of communications traffic using a private key known only to the communicating entities. The private key could be either the shared secret described above (e.g. the PIN number), or one created between two entities using e.g. the Diffie-Hellman algorithm. A weaker form of confidentiality may be achieved by restricting the routing of traffic across a network, although this is not applicable to satellite-based communication.
- *Integrity* guarantees that information sent from one party to another is not changed *en route* (either accidentally or maliciously), and that if such changes have occurred then they can be identified. One way of achieving this is by hashing the information to create a short string and then "signing" (=encrypting) the hash with a secret key: this creates a *hashed message authentication code* (HMAC).
- *Non-repudiation* is a service that ensures that the originator of a message is unable to deny sending the message. This is achieved by the originator creating a MAC using a secret known only to the originator (such as the secret key B corresponding to their digital certificate).

In general, these security services are provided between entities because of the existence of a *secure association*: that is, they share one or more secret keys known only to the entities, and they have agreed on a set of algorithms that will provide the security services.

4. Secure multicast file transfer: SAT-RMTP

Satellite Reliable Multicast Transfer Protocol (SAT-RMTP) provides secure multicast transfer of files such as multimedia clips and bulk data transfers (databases etc), Figure 2 [Koya,02]. An application tool has been implemented that consists of three modules (Figure 3):

- Session module;
- SAT-RMTP module;
- Security module.

The first two modules were developed at the University of Aberdeen, and the security module was developed at the University of Surrey.

The session module announces services (file transfers) being offered, and co-ordinates the transmission and scheduling of these file transfers. The session server periodically transmits session announcements to listening receivers. These announcements include session start and end time, media type and other session related parameters. Based on this information, receivers either join or ignore each advertised session.

The SAT-RMTP module provides reliable transmission of encrypted data between the source (transmitter) and those receivers that have registered for a particular file transfer.

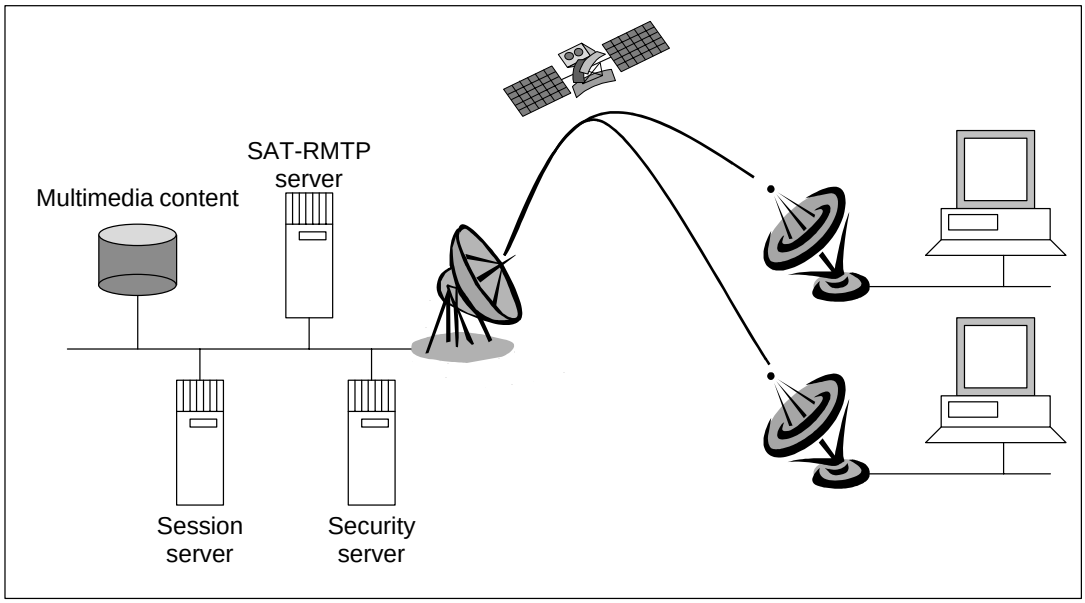

Figure 2: SAT-RMTP overview

The security module implements the following security features:

- Authentication of users (clients) – using a public key algorithm and X.509 digital certificates;
- User access control for each multimedia file;
- Key exchange using public key algorithm;
- Confidentiality, based on file encryption using a secret key algorithm;
- Integrity of the whole file using HMACs;
- Detection of replay attacks using random numbers.

The security module works as follows. Prior to the file transfer, the module generates a secret key and encrypts the entire file using this key. After transfer of the encrypted file (using the SAT-RMTP protocol) each security client requests the key from the security server. The security server authenticates each client using the X.509 digital certificate sent by the client, and then checks the access permissions of the client (user access control). If the client has the correct permissions, the security server unicasts the key (encrypted with the security client's public key, contained in their certificate) to the authorised client. The client can then decrypt the file, and check its integrity using the HMAC.

The entire system has been implemented in C and Linux, and demonstrated running over a satellite emulator.

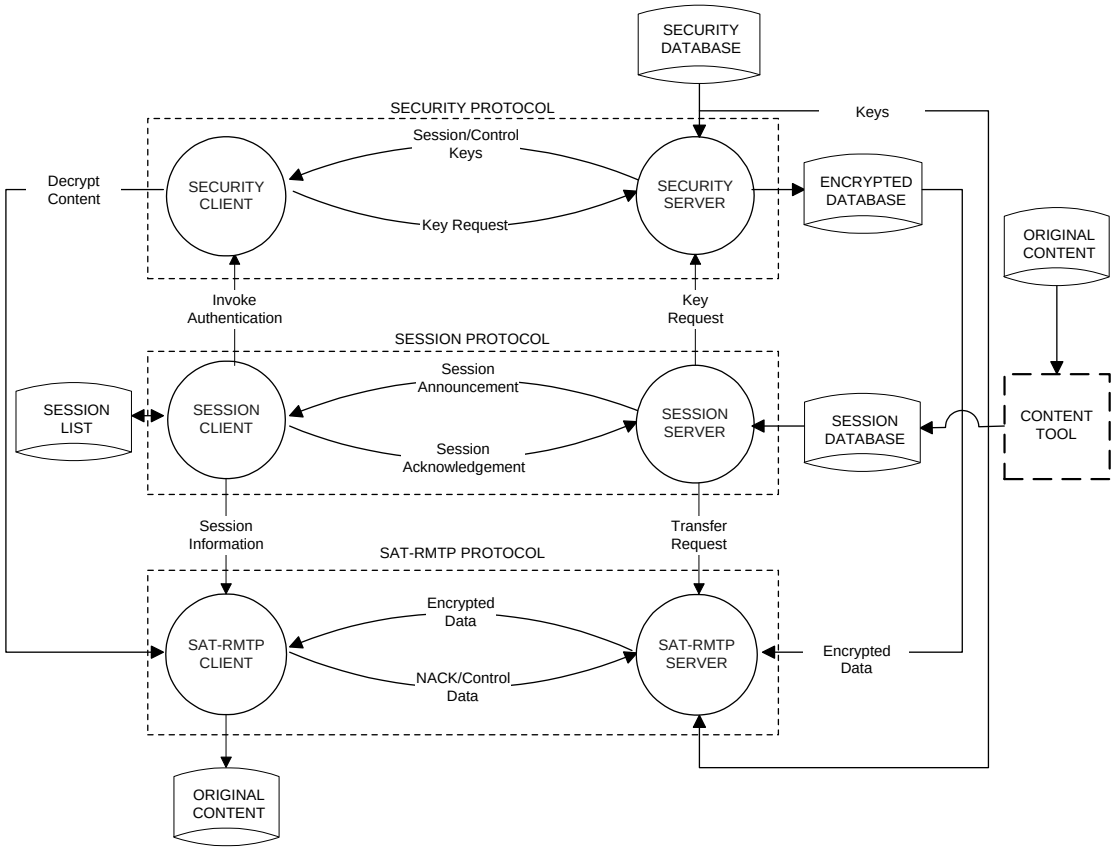

Figure 3: SAT-RMTP application tool architecture

5. Secure data streaming: GSAKMP and LKH

The process of securing a unicast connection is well understood [Maug,98], [Hark,98], [Orma,98], but multicast security is more complex. In principle, a multicast connection can be regarded as a set of unicast connections, but this approach does not scale well for large groups, especially at the scales expected in satellite systems. Protocols that manage the process of distributing keys in a multicast environment are under development [Harn,02], [Arrk,02], [Baug,02].

The principal actors in multicast key management are the group controller (GC) and group members (GMs). The former is responsible for creating and distributing keys and rekeying (to maintain security) as appropriate; the group members are entities with access to the group keys. The GC need not be co-located with the multicast data source. Each group member has an initial one-to-one secure association with the group controller (using techniques such as Diffie-Hellman to create a shared secret known only to the two parties; or a pre-shared secret; or secret exchange using a public key system [Schn,96]). These one-to-one secure associations are then used to create and share information about a *group secure association* between the group controller and the group members. The ultimate aim of the group secure association is to ensure that a single key, usually called the group traffic encryption key (GTEK), is known to all group controller and group members, and to no entity outside the group: this key can then be used to encrypt the data multicast within the group. We now discuss one of the key distribution protocols currently under development within the IETF, GSAKMP Light.

5.1. GSAKMP Light

The lightweight Group Secure Association Key Management Protocol [Harn,02] is a scaled-down version of the original IETF work, GSAKMP. The lightweight version assumes that group members (GMs) have been previously notified of the security mechanisms (i.e. the algorithms used for authentication, encryption, integrity and non-repudiation) used in the group during the group

announcement or invitation. GSAKMP Light provides mechanisms to perform the following key management tasks:

- Disseminate group policy;
- Distribute group keys;
- Rekey the group (e.g. if a member is compromised).

The life cycle of a GSAKMP group secure association can be divided into three phases, and these are now briefly discussed. The discussion is illustrated with the message flows shown in Figure 4; the left side of the diagram represents the actions of the GC, and the right side of the diagram represents the actions of the GMs.

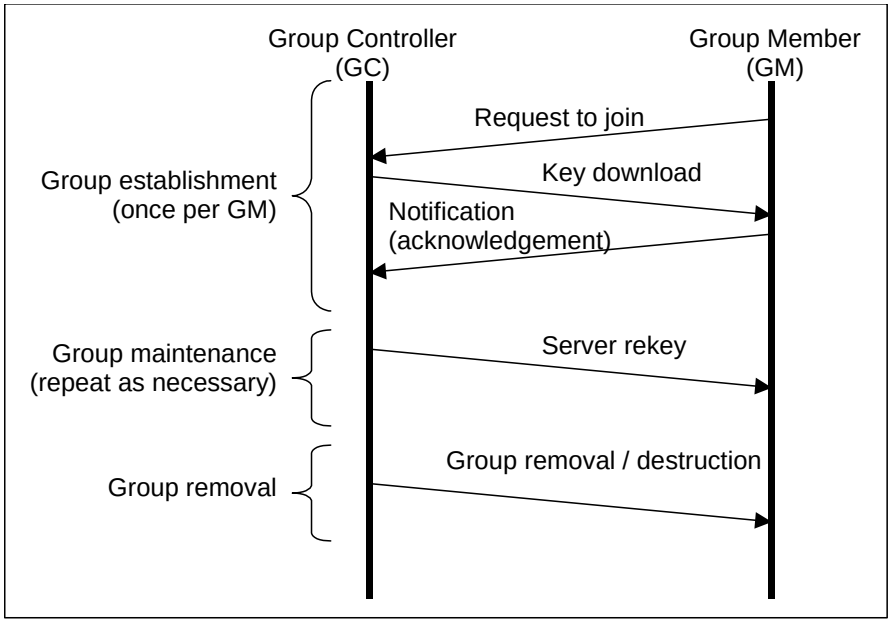

Figure 4: GSAKMP Light message exchange

GSAKMP group establishment

Potential GMs may join a group in one of two ways:

- Invitation (push);
- Request (pull).

Figure 4 illustrates a Request to Join Group (RTJ), a "pull" message sent from a potential GM. On receiving the RTJ, the GC must either accept or deny the request. If accepted the GC checks the RTJ message and following successful authorisation and verification creates the key download payload. This includes the policy token payload, the traffic encryption key payload and also the rekey event payload which helps in scalable group maintenance. The GM returns an acknowledgement to the GC on successful receipt of the key download message.

GSAKMP group maintenance

The Group Maintenance phase includes:

- Member joins and leaves: a group member that elects to voluntarily leave the group is responsible for destroying their own key(s). Any further action for a voluntary leave must be specifically addressed in the group's security policy;

- Group rekey activities: the GC creates and sends a new group key and a rekey array (such as a hierarchy of keys from LKH, see Section 5.3) as illustrated in Figure 4. Subject to authentication, GMs then use these keys for successive decrypting. The reasons for rekeying are discussed further in Section 5.2.

GSAKMP group removal / destruction

The final phase in the group's life cycle is group removal. If a decision is made to destroy the group, the notification may either be broadcast on a key management channel (as shown in Figure 4) or through a directory service.

5.2. Group rekeying

The multicast group may need to be rekeyed for any of a number of reasons:

(1) The group key is usually updated regularly (typically every few seconds or minutes) to reduce the probability of successful cryptanalysis of the encrypted traffic.

(2) The group key may also need to be changed on demand if it is determined that the key has been compromised.

(3) Rekeying may be required when a new user joins the multicast group. This ensures that the user cannot decrypt encoded traffic that was sent prior to their joining (this is called backward secrecy).

(4) Rekeying may be required when an existing user departs from the multicast group. This ensures that the user cannot decrypt encoded traffic that is sent after they leave (this is called forward secrecy).

For large multicast groups that have frequent membership changes the cost of rekeying can be significant, since satellite resources are expensive. Scalable rekeying is therefore an important problem that needs to be considered in order to support secure communications for large dynamic groups. We now proceed to investigate rekey techniques for each of the four functions listed above.

Several techniques exist for rekeying (1) and (3) above: two options are for the new group key to be encrypted with either (a) the old group key, or (b) a separate "control" key negotiated during session establishment. For (2) and (4) above a different rekeying approach is required since the old key is known by at least one user who is no longer to be a recipient of the multicast transmission. We now consider options for this rekeying, focusing in particular on one mechanism, LKH.

5.3. Logical key hierarchy (LKH)

Logical Key Hierarchy (LKH) [Wong,00], [RFC2627], described in more detail below, uses a set of keys arranged in a tree structure to reduce the cost of rekeying. For a tree of outdegree *k*, the number of rekeys transmitted on a member compromise is reduced from $N = k^d$ (for a flat system) to $k \log_k N - 1$.

Improvements to LKH for the specific case of binary trees (*k*=2) have also been proposed in one-way function trees [Bale,99] [Moye,99], and by [Cane,99]: both these approaches reduce the number of rekeys required in the event of compromise of a user from $2\log_2 N - 1$ to $\log_2 N$. Here we focus on the basic mechanism, which is receiving wide support in the research community. We now briefly review the working of LKH, repeating material from [Anno,02].

We initially consider the simple flat key management system. Consider *N* pairwise keys each shared between the group controller and one of the *N* group members (Figure 5a): this represents the flat system described in Section 2. The pairwise key associations are represented by the circles and the group key is represented by the box labelled 'A'. If the group key is changed the new group key has to be encrypted with each user's unique pairwise key and then unicast to that user; each of these

encrypted keys is represented by one of the lines drawn in Figure 5a. Thus for *N* users a total of *N* encrypted keys are generated and transmitted across the satellite network.

We contrast this with LKH, where a tree of keys is used: in Figure 5b the keys are labelled A through O, the circles again represent the pairwise keys, and the lines each represent encrypted keys sent across the network, as we shall now see. Suppose that User 11 needs to be deleted from the multicast group. Then all of the keys held by User 11 (keys F, K, N, O) must be changed and distributed to the users who need them, without permitting User 11 to obtain them or anyone else who is not entitled to them. To do this, we must replace the keys held by User 11, proceeding from the bottom up.

The server chooses a new key for the lowest node (not the leaf, for which a unicast secure association exists between the GC and the GM), and then transmits it encrypted with the appropriate daughter keys. Thus for this example, the first key replaced is Key F, and this new key will be sent encrypted with User 12's unique pairwise key. The second key replaced is Key K, which is sent encrypted with the newly replaced Key F (for User 12) and also sent encrypted with key E (for Users 9 and 10). Key N is then sent encrypted in the newly replaced Key K (for Users 9, 10, and 12) and also encrypted in key L (shared by Users 13 through 16). Finally, Key O is replaced, and this new key is sent encrypted in the newly replaced Key N (for Users 9, 10, and 12 through 16) and also separately is encrypted in key M (shared by Users 1 to 8). Since we are proceeding from the bottom up, each of the replacement keys will have been replaced before it is used to encrypt another key.

The seven keys sent represent a significant saving on the 16 keys that would need to be transmitted using the flat key system of Figure 5a. In general, the number of transmissions required is the sum of the degrees of the replaced nodes. In a *k*-ary tree of depth *d*, this is a total of $kd - 1 = k \log_k N - 1$ transmissions.

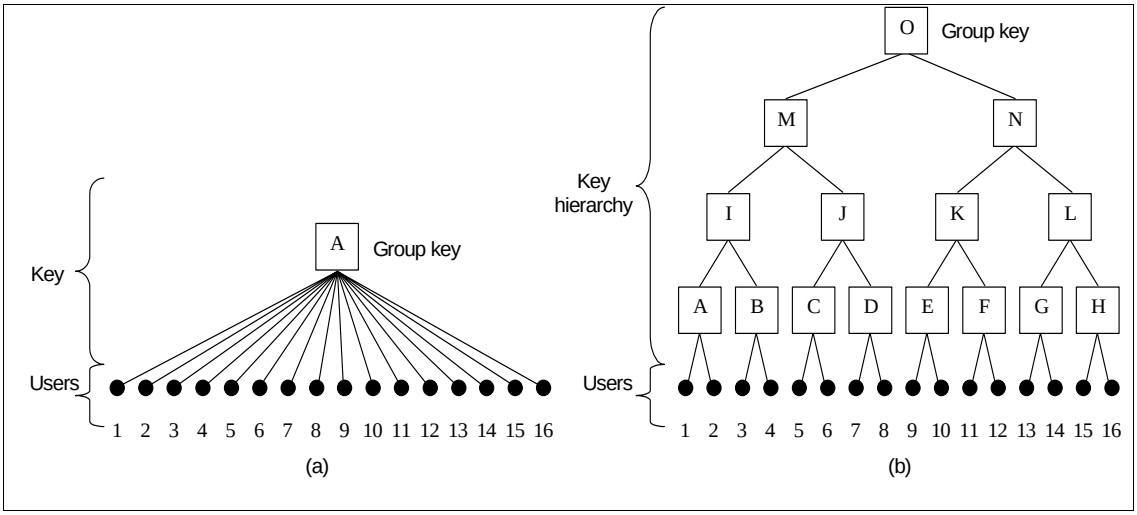

Figure 5: Key hierarchies: (a) *N* pairwise keys (left); (b) logical key hierarchy (right)

The GTEK, used to encrypt data traffic, may, depending on the group security policy, either be key O (Figure 5b), or it may be separately encrypted using key O and transmitted to all group members.

The system is robust against collusion, in that no set of users together can read any message unless one of them could have read it individually. Alternatively, multiple keys can be sent in one message, provided that there exists a means for each user to determine which key in the message corresponds to which node of the hierarchy (for example, [Harn,02]). Taking into account the per-message overheads, a single multicast message uses the fewest bits to transmit the new keys.

5.4. GSAKMP and LKH integration

LKH is currently supported as the default rekey mechanism in GSAKMP. Support for other rekey mechanisms (such as one-way function trees, [Bale,99],[Moye,99]) will be included in the future

versions of GSAKMP. We have implemented GSAKMP and LKH in C and Linux, and have successfully demonstrated key transfer between a group controller and multiple group members.

6. Summary

In this paper we have described two examples of end-to-end security systems. The first, SAT-RMTP, provides secure multicast file transfer using a file transfer protocol that is optimised for satellite environments. The second example is optimised for secure multicast data streaming: this involves the integration of GSAKMP, a key management protocol, with LKH, a key management technique that is scalable to the large number of receivers that are expected in satellite multicast. GSAKMP / LKH integration and interworking has been demonstrated in a laboratory set-up: this technology can be used at application level, transport level or network level.

References

[Anno,02] M. Annoni, G. Boiero, N. Salis, H.S. Cruickshank, M.P. Howarth and Z. Sun, "Interworking between multi-layer IPSEC and secure multicast services over GEO satellites," COST 272, document TD-02-016-P, Thessaloniki Greece, 20-21 June 2002.

[Arrk,02] J. Arrko et al., "MIKEY: Multimedia Internet keying,", IETF Internet Draft, work-in-progress, draft-ietf-msec-mikey-05.txt, Oct 29 2002, expires Apr 2003.

[Bale,99] D. Balenson et al., "Key management for large dynamic groups: one-way function trees and amortized initialization", IETF Draft, work-in-progress, draft-balenson-groupkeymgmt-oft-00.txt, Feb 1999.

[Baug,02] M. Baugher et al., "The group domain of interpretation," IETF Draft, work-in-progress, draft-ietf-msec-gdoi-06.txt, Oct 2002, expires April 2003.

[Cane,99] R. Canetti et al., "Multicast security: a taxonomy and some efficient constructions," *Proc. IEEE INFOCOM* 1999, pp. 708-716.

[Geoc] GEOCAST project home page, http://www.geocast-satellite.com/

[Hark,98] D. Harkins and D. Carrel, "The Internet key exchange," IETF RFC2409, Nov. 1998

[Harn,02] H. Harney, A. Schuett and A. Colegrove, "GSAKMP Light," IETF Internet Draft, work-in-progress, draft-ietf-msec-gsakmp-light-sec-01.txt, Jul 2002, expires Dec 2002.

[Koya,02] M. Koyabe, A. Matthews, G. Fairhurst, S. Iyengar, M.P. Howarth and H. Cruickshank, "A network tool for multimedia file distribution," GEOCAST project paper ref. GEOC-UOA-2400-3, 1 May 2002.

[Maug,98] D. Maughan et al., "Internet Security Association and Key Management Protocol (ISAKMP)", IETF RFC2408, Nov. 1998.

[Moye,99] M.J. Moyer et al., "A survey of security issues in multicast communications," *IEEE Network*, Nov 1999, pp.12-23.

[Orma,98] H. Orman, "The OAKLEY key determination protocol," IETF RFC2412, Nov. 1998.

[RFC2627] D. Wallner, E. Harder and R. Agee, "Key management for multicast: issues and architectures," IETF RFC2627, June 1999.

[Schn,96] B. Schneier, "Applied cryptography," John Wiley & Sons, 1996.

[Wong,00] C.K. Wong, M. Gouda and S.S. Lam, "Secure group communications using key graphs," *IEEE/ACM Trans. Networking*, Vol. 8 No. 1, 2000, pp.16-30.