Hierarchical approaches for multicast based on Euclid's algorithm

N. Antequera J.A. López-Ramos *
Departamento de Algebra y Análisis Matemático
Universidad de Almería
04120 Almería, Spain

1 Introduction

Multicast communications allow a host to simultaneously send information to a set of other hosts, avoiding the establishment of point-to-point connections with all of them. There exist many situations where multicast reveals to be the most suitable way to distribute the information such as pay-per-view IPTV or P2PTV, private multiconferences, or any private service that involves several participants or clients. This makes growing the interest in research on appropriate protocols for secure multicast. Some surveys on this field can be found in [2], [9], or more recently in [12].

In [3] the authors made a computational approach to the problem and introduce a solution based on the Chinese Remainder Theorem, the so-called Secure Lock. However, as shown in [4], computational requirements become quickly huge as the number of user grows. To reduce the number of computations, in [10], a divide-and-conquer extension to Secure Lock is introduced. It combines the well-known Hierarchical Tree Approach, [11] and the Secure Lock. The authors propose an arrangement of the members as in a HTA, and use Secure Lock to refresh keys on each tree level.

In [7] the authors introduce a new computational method based on Euclid's algorithm for computing the greatest common divisor of two integers that shows to be adequate in an environment where users are constantly joining and/or leaving the system with low communication overcomes and key storage and gets forward and backward secrecy: new members cannot decrypt information multicasted before their arrival and those leaving the multicast group are not able to access the encrypted information after their departure. The protocol has three parts: a key distribution scheme, an alternative key refreshment authentication and a validation protocol between authorized users. This scheme was object of a cryptoanalysis in [8], but positively addressed in [1]. However, as

*This work has been supported by the Spanish Ministry of Science and Innovation (TEC2009-13763-C02-02) and Junta de Andalucía (FQM 0211).

shown in [7], the length of the rekeying messages grows linearly as the number of users.

The aim of this work is to show how properties of the Euclid's approach combined with the hierarchical tree approach gives rise to a powerful method that allows to multicast messages in environment with huge and highly dynamic audiences with very low communication overcomes, including the length of the rekeying messages. We will show also how the use of Euclid's approach becomes natural in some hierarchical tree situations from the properties of the prime numbers, certainly "the key of this method".

The structure of this paper is as follows. Firstly we recall briefly Euclid's protocol for multicast. Then we discussed rekeying messages in a hierarchical tree distribution of users and in situations where users have different attributes for accessing different services or information. These approaches constitute centralized protocols, i.e., a single entity is in charge of creating and distributing the rekeying messages. Finally, in the last section we introduce a distributed situation where our approach is also suitable because of its nature. Now some detached users, namely, the group managers, are in charge of creating rekeying messages from an original one coming from the Key Server for their corresponding controlled groups. This distributed approach is particularly appropriated when trying to avoid certain type of attacks that can be developed by legal users as we will show.

2 Some background on Euclid's approach

Let us recall from [7] the construction of the key distribution scheme. Every user holds a large prime number, $x_i,\ i = 1, \ldots, n$. Then the Key Server selects

- m and p, large prime numbers, such that p divides $m - 1$.
- k and δ, such that $\delta = k + p$ and $\delta < x_i$, for every $i = 1, \ldots, n$.
- g that verifies $g^p = 1 \ mod\ m$.
- The session key to be distributes is $g^k \ mod\ m$.

Then the Key Server makes the following steps:

1. It calculates $L = \prod_{i=1}^{n} x_i$, which is kept private.
2. It finds u, v, by means of the Extended Euclidean Algorithm, such that

$$u \cdot \delta + v \cdot L = 1$$

To recover the key, each member i calculates $u^{-1} \ mod\ x_i = \delta$ and $g^\delta \ mod\ m = g^k$.

As it was pointed out in [7], every member in the multicast group is able to calculate a multiple of the product of all the tickets, L as follows. Since $u \cdot \delta \ mod\ x_i = 1$ for every $i = 1, \ldots, n$, then $u \cdot \delta - 1$ is a multiple of L, say $v \cdot L$.

In [8, Section 3], the authors propose a "man in the middle" attack using this multiple that is easily avoided by just considering any authentication protocol added to the distribution.

In order to avoid two other attacks considered in [8], the product L should contain some information corresponding to a "fake user", i.e., it is advisable to include in the product at least one prime that is not being used by any user.

3 Join and leaves in HTA+Euclid approach

As in HTA and the Secure Lock + HTA approaches, our proposal uses the divide and conquer strategy. As in the Secure Lock+HTA case, the number of transmissions is reduced with respect to the HTA case, the computational requirements at the Key Server's side are still very low and the length of rekeying messages is considerably reduced, so we can give service to a much bigger number of users without delaying in rekeying operations. The idea is exactly the same as the one introduced in [10]. However let us assume a more general scenario than that considered in [10, Section 3.4]. Consider a hierarchical tree with a depth of 4, i.e., the number of levels below root is 3, and a degree of n, i.e., the number of children below each parent node is n (see Figure 1).

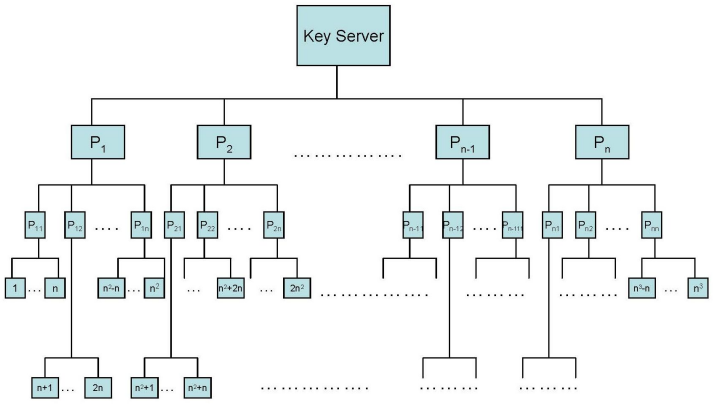

Figure 1: Hierarchical Tree

Let us assume with out loss of generality that user 1 wants to leave. It can be easily observed that we need the following messages to refresh the key and preserve forward secrecy.

1. Key Server $\rightarrow \{2, \ldots, n\}$: E($P_S$, P_1, $P_{1,1}$), being P_S the session key at the Key Server and using the private information hold by users $\{2, \ldots, n\}$
2. Key Server $\rightarrow \{n+1, \ldots, n^2\}$: E($P_S$, P_1,) using the private information $P_{1,2}, \cdots, P_{1,n}$.

3. Key Server $\rightarrow \{n^2+1, \ldots, n^3\}$:E($P_S$) using the private information $P_2, \cdots P_n$.

where P_i, $P_{i,j}$ are prime numbers for every $i = 1, \ldots, n$ and $j = 1, \ldots, n$ and by $E(-)$ we mean the encryption of the corresponding information using the Euclid's approach.

We also detach that if we are dealing with primes of 1024-bit length, then the length of messages 1, 2 and 3 are about $3 \cdot n \cdot 128$, $2 \cdot n \cdot 128$ and $n \cdot 128$ bytes respectively. These means, in case $n = 100$, that we are giving service to one million users are messages are about 37kb, 25kb and 12kb respectively and the computing time to generate them does not depend on the number of users, since operations are just multiply or divide by a prime a product of primes, say L, select a new value k for $\delta = k + p$ and calculate the greatest common divisor of L and δ.

4 Multilevel security and Euclid's method

As it was show in the previous section the Euclid's algorithm allows to deliver efficiently a secret to a huge plurality of users. But this can be also used to rekey in environments with a key hierarchy as the Secure Lock case (cf. [10, Section 4]). The argument is essentially the same although we will describe it since depending on the situations we could simplify it in some way.

Assume first that the audience is composed by a huge amount of users, as in a Pay-Per-View TV broadcasting and that we have four levels of service. In the highest one, T, we get the complete set of services offered, let us say movies, sports, entertainment and general channels. Then we have a reduced version where not all the services are obtained, a packet containing sports, entertainment and general channels, namely R. The third category, E, could be formed by just entertainment and general channels and, finally, the basic packet, B, offering simply general channels. This situation is represented by Figure 2. In this case, the protocol is exactly the same as that introduced [10, Section 4.2] and we do not encounter any problem with computational requirements as outlined in that case, where this method is applicable to just situations where the number of users is reduced due to this fact.

4.1 Enhancing multilevel security through properties of primes

An alternative to this method avoids storing any other key different from the private integer hold by every user. The idea is based on the following fact:

Let x_1 and x_2 be two different prime numbers and consider $L = x_1 \cdot x_2$ and δ an integer. Let $\delta < x_1,\ x_2$, then $u = \delta^{-1}\ mod\ L$ if and only if $u\delta = 1\ mod x_i$, for $i = 1, 2$.

Consider now a situation as that given by Figure 3, where groups corresponds with the same hierarchy as considered above. Then any message sent to the a group, the one with less attributes placed at the top should be decrypted by

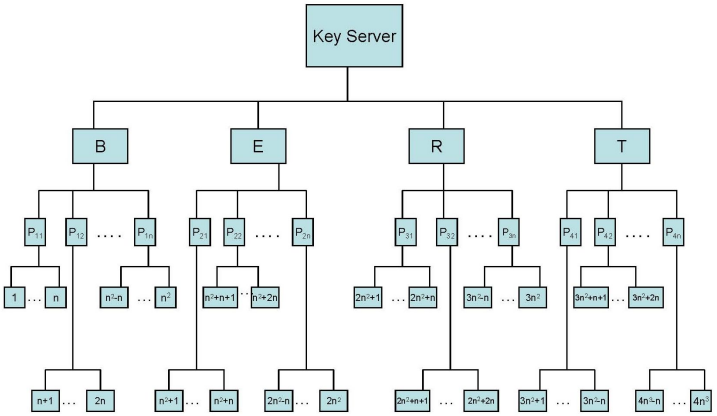

Figure 2: Hierarchical Tree Multilevel Security

the other subgroups of users higher in the hierarchy. Thus we consider sets of primes $P_i = \{p_{i,j}\}$ for $i = 1, \ldots, 4$, $j = 1, \ldots, s_i$ where primes in P_i are used by users in group G_i in the following way:

Every user j in the group G_i is assigned a ticket $x_j = \prod_{h=1}^{i} p_{h,j_h}$, for some j_h.

Now if the Server sends a message to be read just by group k and its corresponding subgroups with higher privileges, the server simply changes the session key, i.e., the key used to encrypt the original message, and encrypts it using the product $L = \prod_{p \in P_i} p$. The process is as follows:

1. m and p, large prime numbers, such that $m - 1 = p \cdot q$.
2. k and δ, such that $\delta = k + p$ and $\delta < p_{h,j}$, for every $p_{h,j} \in P_h$.
3. g that verifies $g^p = 1 \ mod \ m$.
4. The session key to be distributes is $g^k \ mod \ m$.

The encryption is developed as usual now

1. It calculates $L = \prod_{p \in P_i} p$, which is kept private.
2. It finds u, v, by means of the Extended Euclidean Algorithm such that

$$u \cdot \delta + v \cdot L = 1$$

3. The key server broadcasts in the multicast group (g, m, u).

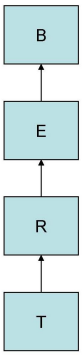

Figure 3: Multilevel Security Groups

To recover the key, each member j calculates $u^{-1}\ mod\ x_j = \delta$ since $\delta < p_{i,j} < x_j$ and $g^{\delta}\ mod\ m = g^k$. From the above fact it is clear that only users holding a ticket with prime $p_{j,i}$ in its factorization are able to get u, and so the session key.

A second version allows to store just one prime of the same length per user. In this case the idea is even more simple. Taking into account the order of the integers we just assign those less integers to the users with a lower level of privileges. So let

$$\{p_1, \ldots, p_{k_1}, \ldots, p_{k_2}, \ldots, p_{k_3}, \ldots p_n\}$$

be the set of tickets hold by users and ordered with respect to the usual order of integers and let k_1, k_2 and k_3 be the corresponding indexes of those users belonging to higher status, i.e., users $u_1, \ldots u_{k_1-1}$ are in the group placed at the top of Figure 3 and users $u_{k_3}, \ldots u_n$ belong to the group placed at the bottom, those with higher privileges.

Then, to rekey messages directed to all the audience, the Key Server should use the product of all the primes, $L = \prod_{i=1}^{n} p_i$. If message is to be read by all users excepting those with less privileges, then the Key Server will use the product $L = \prod_{i=k_1}^{n} p_i$ and, in case the message is directed just to the users with the highest privileges, then the product to be used will be $L = \prod_{i=k_3}^{n} p_i$.

However, as noted at the introduction, if the number of users is big, then the Euclid's approach does not offer an adequate solution due to the length of messages and a distribution of users through a hierarchical tree as above is recommended.

4.2 Using Euclid's method for many-to-many communications

As before, the situation dealt in [10, Section 4.3] is also easily extendable to the Euclid's method. Let us show an example. Assume a situation as that given by

Figure 2, for $n = 4$. The join of user 40 will be accomplished with the following rekeying messages:

1. Root$\rightarrow \{37-40\} : E(K_R, K_C, K_V)$ using the primes (private information) hold by users 37,38,39 and 40.
2. Root$\rightarrow \{33-36,\ 41-48\} : E(K_R,\ K_S)$ using the primes K_{IX}, K_{XI} and K_{XII}.
3. Root$\rightarrow \{1-32\} : E(K_R)$ using the primes K_A, K_B and K_C.
4. Root$\rightarrow \{49-64\} : E(K_S,\ K_R)$ using the primes K_A, K_B and K_C.

5 Group controllers, a distributed protocol

Euclid's approach is also aplicable to a distributed situation, which is usually known as a decentralized protocol. Decentralized architectures divide management of large groups among subgroups with a trusted agent in charge of each subgroup. The best known example is probably Iolus ([6]). The situation we are proposing is as follows. Audience is divided into subgroups, each one managed by a trusted agent or group manager and there is a Server in charge of distributing contents, encrypted with a session key and assigning private information to new users and to group managers. Information provided by the Server, besides the above mentioned encrypted contents consists of the following:

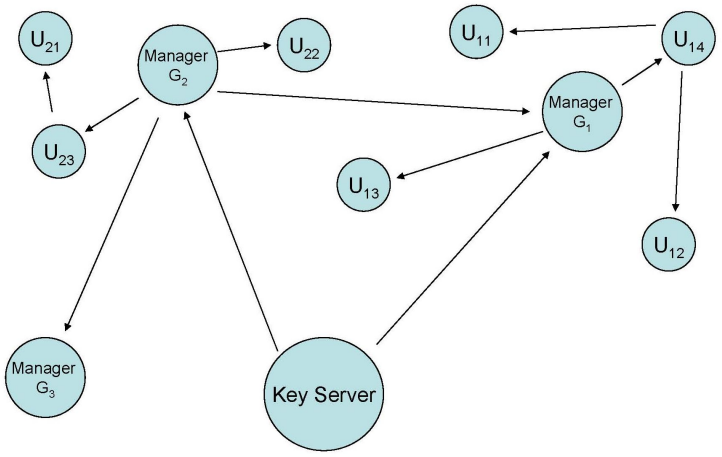

Figure 4: Distributed Protocol Groups

Initialization steps

1. Users are distributed by groups $\{G_1, \ldots, G_n\}$, each one managed by a trusted user or group manager u_j of group G_j, $j = 1, \ldots, n$, as shown in

Figure 4. The Key Server assigns a prime $p_{i,j}$ for every user i in group G_j in the system, including the corresponding primes for the group managers, that we will denote by p_j.

2. The Server calculates the products $L_G = \prod_{i=1}^{n} p_i$ and $L_j = \prod_{i=1}^{k_j} p_{i,j}$.
3. The Server sends individually $\{L_G, L_j\}$ to u_j, the manager of group G_j.

Distributing the information

1. The Server encrypts the information with a session key K_S and sends it to the group managers using L_G.
2. The group manager receives the encrypted key and decrypts it using p_j. Then sends K_S using L_j to users $u_{i,j}$ in group G_j.
3. Each user $u_{i,j}$ gets K_S by using $p_{i,j}$.

Rekeying messages

1. A rekeying message without a user joining or leaving the system just runs as above.
2. If user $u_{i,j}$ leaves or joins the system, then the Server calculates the new L_j.
   (a) The server sends individually L_j to u_j.
   (b) The new session key is distributed as explained above.

5.1 Security on the distributed approach

As pointed out in the introduction, in [8] the authors proposed a "man in the middle attack" against the Euclid's approach that is easily avoided, as for any cryptosystem, by adding some information that provides authentication to distributed messages. In [7] an authentication protocol associated with the key distribution protocol based on the Euclid's approach was introduced. In [8] the authors also show an attack on this authentication protocol using a multiple of the product L_j that can be calculated by any member in G_j. This was addressed in [1] in several manners, but one of them is specially appropriated to the distributed situation. In that case (cf. [1, Section III]), it was shown that the attack, developed by a legal user, can be detected by users whose private information is less than a determined bound, namely some random information that is selected by the attacker. The easiest way to avoid this is, as noted in that case, to assign the group manager a prime p_j that is less than any other prime $p_{i,j}$ in group G_j. In this way, the group manager will detect the attack without making the Server intervene in the authentication process of internal messages in group G_j.

The same attack can be developed in a higher level by one of the group managers. However, there will exist one of them, that one holding the least

prime of all p_j, that will detect the attack. In case groups corresponds to groups with a different status in the hierarchy, what is advisable to avoid this attack is that the group manager of the group with highest priorities holds this detached prime and that the other primes are distributed from the least to the highest in the inverse hierarchy of groups, i.e., those managers corresponding to groups with higher priorities will have the least primes.

5.2 Authentication between users

In [7] an authentication scheme connected with the Euclid's method to distribute secrets was also introduced. It was also cryptoanalyzed in [8] and use to compromise private information of legal users, but it was addressed again in [1, Section IV]. As noted in [7] this authentication between users protocol does not scale as the number of user grows since it can collapse the Server if all users apply for the same service at the same time, but shows to be an alternative if for some reason, maybe due to the characteristics of devices use to decrypt the information, public cryptography is not available and so the use of a certificate becomes difficult. However, this could be more usable in a distributed situation as proposed. So let us show it once the security issue is solved (cf. [1, Section IV]).

Let us assume first, as in the Euclid's approach that m and p are large prime numbers, such that p divides $m-1$ and that all the primes hold by the users, p_j and $p_{i,j}$ verify that are larger than m. Let g be such that verifies $g^p = 1 \ mod \ m$. Then

1. User $u_{i,j}$ in group G_j selects a set of random integer r_h $h = 1, \ldots, s$ such that $1 < r_h < m$ for every $h = 1, \ldots, s$ and sends it to the group manager u_j.
2. The group manager u_j selects one of them such that $inv_h = r_h^{-1} \ mod \ L_j$ is such that $m < inv_h < p_{i,j}$ for every private user's information $p_{i,j}$ in G_j and sends it to user $u_{i,j}$, namely inv.
3. User $u_{i,j}$ sends $\{inv, \ g^{p_{i,j}} \ mod \ m\}$ to user $u_{k,j}$.
4. User $u_{k,j}$ computes $r_k = inv^{-1} \ mod \ p_{k,j}$ and $\beta_k = r_k \cdot (g^{p_{i,j}})^p_{k,j}$ and sends back $\{\beta_k, g^{p_{k,j}}\}$ to $u_{i,j}$.
5. User $u_{i,j}$ computes $\beta_i = r \cdot (g^{p_{k,j}})^{p_{i,j}}$ and authenticates $u_{k,j}$ as a legal user in case $\beta_k = \beta_i$.

5.3 Some additional considerations

In case a user in group G_j wants to multicast a message in his own group, the group controller may authenticate it and user sends the message encrypted using L_j. To do so, every member in G_j should be provided of such a product of private primes.

If the situation is that a user wants to multicast a message to any of the other groups, the user sends the message to the group controller (maybe once he/she was authenticated) and then, the group controller decrypts and authenticates the message and routes it to the desired group G_h using the corresponding L_h. In that case every group manager should be given every product L_j corresponding to every group G_j.

We could also avoid going though the group controller and send the message directly to any of the other groups. In that case, every user should be communicated in some manner on the corresponding product of private information of users and take into account what established previously concerning generation of authentication messages (cf. [1]) in order that the group controller is still able to authenticate these messages.

Different policies for communicating messages could be considered, but always taking into account what concerns security above mentioned.

References

[1] N. Antequera and J.A. Lopez-Ramos. Remarks and countermeasures on a cryptoanalysis of a secure multicast protocol. *Proceedings of 7th International Conference on Next Generation Web Services Practices, Salamanca 2011*, Salamanca (Spain) 201–205 (2011).

[2] K.-C. Chan and S.-H.G. Chan. Key management approaches to offer data confidentiality for secure multicast. *IEEE Network*, 17(5) 30-39 (2003).

[3] G. Chiou and W. Chen. Secure broadcasting using the secure lock. *IEEE Trans. Softw. Eng.*, 15(8) 929–934 (1989).

[4] P.S. Kruus and J. P. Macker. Techniques and issues in multicast security. *Proceedings of Military Communications Conference, MILCOM*, 1028–1032 (1998).

[5] B. Liu, W. Zhang and T. Jiang. A Scalable Key Distribution Scheme for Conditional Access System in Digital Pay-TV System. *IEEE Consumer Electronics*, 50(2) 632–637 (2004).

[6] S. Mittra. Iolus: A framework for scalable secure multicasting. *Proceedings of the ACM SIGCOMM* 27(4) (New York, Sept.) ACM, New York, 277288 (1997).

[7] J.A.M. Naranjo, N. Antequera, L.G. Casado and J.A. Lopez-Ramos. A suite of algorithms for key distribution and authentication in centralized secure multicast environments. To appear in *J. Comp. Appl. Math.*, DOI: 10.1016/j.cam.2011.02.015.

[8] A. Peinado and A. Ortiz. Cryptanalysis of Multicast Protocols with Key Refreshment Based on the Extended Euclidean Algorithm. *Proceedings of CISIS 2011, Lecture Notes in Computer Sciences*, 6694 177-182 (2011).

[9] S. Rafaeli and D. Hutchison. A Survey of Key Management for Secure Group Communication. *ACM Computing Surveys*, 35(3) 309–329 (2003).

[10] O. Scheikl, J. Lane, R. Boyer and M. Eltoweissy. Multi-level secure multicast: the rethinking of secure locks. *Parallel Processing Workshops, 2002. Proceedings. International Conference on*, 17–24 (2002).

[11] D. Wallner, E. Harder and R. Agee. Key management for multicast: Issues and architectures, RFC 2627, 1999.

[12] S. Zhu and S. Jajodia. Scalable group key management for secure multicast: A taxonomy and new directions. *Network Security*. H. Huang, D. MacCallum and D.-Z. Du (eds.) Springer, United States, 57–75 (2010).