INTERNATIONAL JOURNAL OF SATELLITE COMMUNICATIONS
Int. J. Commun. Syst. Network 2008; **26**:269–289
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/sat.912

Maximizing service availability for secure satellite broadcasting

Roberto Di Pietro[1,2,*,†] and Stefano Iannitti[3]

[1]*Department of Computer Engineering and Maths, Universitat Rovira i Virgili, UNESCO Chair in Data Privacy, Av. Països Catalans 26, E-43007 Tarragona, Catalonia*
[2]*Dipartimento di Matematica, Università di Roma Tre, Largo S. Murialdo n.1, 00146 Roma, Italy*
[3]*Agenzia Spaziale Italiana (ASI), Viale Liegi n.26, 00198 Roma, Italy*

SUMMARY

Cryptography is the most suitable technique to protect the access to subscribed value-added services for mobile applications provided through satellite broadcasting (e.g. localization and mobile TV). However, if a temporary loss of signal is experienced by receivers, not only the data streaming is compromised, but also the key management-related messages. Hence, when the signal is received again, it is impossible for the receivers to decrypt it. One way to overcome this issue is the re-transmission of keys to guarantee that the largest set of legitimate receivers has the updated (set of) keys.

In this paper we analyze the problem of key update in a mobile user context in order to maximize the service availability. In particular, we provide the following contributions: we propose a mathematical model that captures all the relevant features of mobile users. Then, we formally prove a few novel results on the structure of the admissible rekeying sequence. These results are used to design a scheduling algorithm for key broadcasting that, according to our model, minimizes the number of mobile users that are prevented from updating their crypto keys, hence continuing accessing the service. Finally, simulation results support our theoretical findings. Copyright © 2008 John Wiley & Sons, Ltd.

KEY WORDS: secure multicast; key management protocol; group communications; mobility; scheduling algorithm

1. INTRODUCTION

Mobile devices have been receiving attention in the latest years as more and more applications are conceived for a mobile use. Typical examples of such applications are: position location,

*Correspondence to: Roberto Di Pietro, CNR - ISTI, via G. Moruzzi 1, 56124 Pisa, Italy.
†E-mail: roberto.dipietro@urv.cat,dipietro@mat.uniroma3.it

Contract/grant sponsor: Spanish Ministry of Science and Education
Contract/grant sponsor: Government of Catalonia; contract/grant number: 2005 SGR 00446
Contract/grant sponsor: European Satellite Communications Network of Excellence; contract/grant number: IST-27393
Contract/grant sponsor: PERSONA; contract/grant number: 045459
Contract/grant sponsor: INTERMEDIA; contract/grant number: 38419

mobile TV, and satellite phones. Access to a specific value-added service will require a subscription by the user, while cryptography is the most suitable technique to guarantee the service only to subscribers [1]. Owing to device resource scarceness, symmetric key cryptography seems a better option than asymmetric key cryptography. Moreover, services can be delivered to mobile receivers through satellite broadcasting, which allows the provision even in remote locations (i.e. where no similar terrestrial service is present). However, small mobile devices (receivers) could be resource constrained, and hence could be not equipped with an up-link to communicate with the satellite (i.e. to the source). Further, even if in principle the receivers could be able to send feedback to the source, one could chose not to send feedback to avoid the management burden of the feedback implosion problem [2,3], that is, when a large number of receivers send feedback to the source.

One method for enabling secure multicast communication is the periodic distribution of a new key (called a session key) to group members. All messages exchanged within the group during a fixed interval of time, or session, are communicated securely through encryption under this session key. We assume that prior to the start of each session, the group manager broadcasts a packet containing that session's key (and perhaps, other information) to the group. Because group membership is dynamic (that is, users may be added and removed from the group periodically), the key distribution broadcast targets only the relevant group members. The problem of distributing keys over a reliable channel has received much attention (see, for example, [4,5]). However, this situation is also complicated in specific locations where the satellite signals cannot be received for a number of reasons, like *shadowing* (due to the presence of physical obstacles, for instance, in urban canyons or indoor environments) or fading [6]. In these cases, not only the data streaming is compromised, but also the key management-related messages. In fact, when a receiver loses one (or more) rekeying steps, it loses synchronization with the cryptographic keys used by the broadcaster to encrypt the data payload. A way to overcome this issue is the re-transmission of keys to let a greater number of receivers to have the keys updated. The key transmission order is evidently important to guarantee that the largest set of legitimate receivers has the updated (set of) keys.

In the following, we address the problem of maximizing the access to services for mobile users belonging to the same security group, with no access to a feedback link. The solution provided is the design of a scheduling algorithm that provides the rekeying sequence that maximizes the set of users with the most updated rekeying packet (RP). To reach this goal, we first provide a population model to be used for the design of the scheduling algorithm. On the basis of the user population model analysis, we prove a few formal results, that is, the optimal rekeying sequences show a special structure that can be used by the scheduling algorithm to reduce the set of admissible sequences; a following result allows to further prune the set of optimal sequences and, consequently, to reduce the computational burden of the scheduler. The theoretical results have been supported by simulations, thus proving the effectiveness of our approach, and showing the scheduler behavior in several interesting cases. This work paves the way for future works along different directions, such as adopting different user population models to detail specific user behavior.

The rest of this paper is organized as follows. Section 2 introduces related work in the field; Section 3 sets the framework for Section 4, where we introduce the analytical model and the proposed scheduling algorithm; Section 5 reports on the simulation, while Section 6 highlights some concluding remarks.

2. RELATED WORK

Based on the interdependency of rekeying messages, group key management algorithms can be classified into *stateless* and *stateful* rekeying schemes. In particular, the stateless approach was introduced in the seminal work of Naor *et al.* [7], where the authors proposed the subset difference revocation (SDR) scheme. As for stateful rekeying, we will consider logical key hierarchy (LKH) [8], that will be revisited in Section 3. Readers interested in the differences between the two approaches can refer to [9]. In particular, in the latter paper the authors compare the storage cost and the rekeying cost (number of encrypted keys) of LKH and SDR in immediate and batch rekeying scenarios. The simulation studies show that LKH performs better in immediate rekeying and small batch rekeying, whereas stateless rekeying performs better as membership changes are processed in larger batches. In this paper we will deal with the stateful approach only.

The scalability problem of group key management for a large group with frequent joins and leaves was addressed also by Mittra with the Iolus system [10], and Wang with the LKH [8]. Both Iolus and LKH solve the scalability problem by making use of a hierarchy. However, the system architectures are very different in the two approaches. In the Iolus system the main idea is to establish a geographical hierarchy of keys using intermediaries called group security agents (GSA). In the key graph, the main idea is to balance the cost of additions and deletions, arranging keys in a logical hierarchy (unlike the physical hierarchy of Iolus), with the root key being the session key and the individual user keys being the leaves. LKH will be revisited in Section 3. Among the approaches that use batch techniques, it is worth referring to Kronos [4]. In Kronos the rekeying operation is driven by time rather than membership changes, that is, a new key is generated locally after a certain period of time, without incurring in the overhead of the new key distribution related to a centrally generated key. Kronos has been described in detail within the distributed approach proposed in [11]. An interesting comparison between the dvb conditional access implemented on satellite IP secure multicast can be found in [1].

For several years, the theory community has studied the problem of allowing only legitimate users to access multicast data. Starting with [12], and continuing with [13,14], the theory community has studied this problem using different models and different assumptions from those used in the Internet community. We refer the reader to [15] where these differences are well treated. For example, Fiat and Naor [12] introduced the concept of k resilient broadcast. In their approach, a secret distributed to a subset of n destinations is resilient to collusion by up to k other destinations. Finally, in [16], it has been shown that while key distribution schemes can trade addition cost for deletion cost, for any scheme there is a sequence of $2n$ insertions and deletions whose total cost is $\Omega(n \log n)$. Thus, any key distribution scheme has a worst case of $\Omega(\log n)$ either for adding or for deleting a user.

As for recent contributions, in [17] a methodology to establish the minimal key length that guarantees a specified level of confidentiality is proposed. Reducing the keys length reduces the communication cost, the computation cost, the total rekeying completion time, the storage required by the users, and enhances the reliability of rekeying communications. The proposed methodology scales with the number of users in the multicast group. Further, in [18] the authors focus on key authentication problem for the LKH multicast model.

As for resilience to packet loss, in [19,20] two key distribution protocols that are resistant to packet loss through non-interactive means are described. Both are motivated by the single sender content distribution setting and take a tree-based approach to key distribution and achieve resistance to loss by appending additional key update information to the packets that follow a key distribution broadcast. In [20], resistance to packet loss is ensured by using error-correcting codes to generate information about past group keys. If a certain fixed number of packets is received after a lost one, it is possible for the user to reconstruct the lost information. In [19], these ideas are built upon to allow resistance to correlated packet loss. Depending on the membership change(s) that trigger the rekeying, 'hints' for updated keys are attached to subsequent data packets. The hints can be as small as half the size of the keys themselves, but leave the user with significant work to do in order to recover missing keys.

The approaches in [19,20] differ significantly from the innovative approach based on self-healing techniques introduced in [21,22]. The self-healing property requires that any pair of preceding and following packets be sufficient for recovering the lost key. With this self-healing requirement, it is possible to communicate with all group members through short broadcasts even though the underlying set of personal keys is flat rather than hierarchical (each key is stored by one user), which has the advantage of permitting traceability. However, the self-healing property can be obtained only for a certain number of consecutive packets (*window*). If this window is exhausted, the source needs to distribute to receivers auxiliary key material to enable another self-healing window. Hence, the auxiliary key material distribution phase could be affected by packet loss.

In [23] it is shown how to perform an efficient rekeying of large groups with dynamic membership: minimizing the overall time it takes for the key server and the group members to process the rekey message. Specifically, the authors concentrate on rekey algorithms based on the LKH [8], and minimize the longest sequence of encryptions and decryptions that need to be done in a rekeying operation. In [6] the transmission scheduling with faded channel and constrained devices is analyzed, whereas in [24] the dynamics of key management in satellite communications is analyzed.

Finally, note that a five pages preliminary workshop version of this work appeared in [25]. In this paper we extend [25] in several ways. First, we provide formal results and extensive simulations supporting our analytical findings; second, the paper is enriched with a comprehensive survey of the related literature, and third we highlight further, effective research directions paved by this paper.

3. SYSTEM ASSUMPTIONS AND DEFINITIONS

In this work, we consider the following assumptions:

- The data broadcasted by satellites (referred to as *data stream*) are encrypted.
- Only authorized-legitimate devices/users can decrypt the data.
- The set of authorized users may change frequently (i.e. joins and evictions of users are frequent) and keys need to be changed in a suitable way to preserve backward and forward confidentiality.
- The key management is performed through 'Over The Air Rekeying' (OTAR), i.e. the keys are transmitted via rekey packets broadcasted by the same satellites broadcasting the encrypted data.
- Only a fixed amount of bandwidth is allocated to key management data (OTAR operations).
- Receivers cannot send any feedback to the source (i.e. to the broadcasting satellites).

- Owing to shadowing and other interference effects, receivers are not always able to receive the encrypted data stream as well as the OTAR data.

Delivering keys to a large group of users or to different groups of users via over-the-air rekeying may face scalability problems that can be improved by adopting a *LKH* approach (see [8]). Indeed, a secure group can be formalized as a triple (U, K, R), where U denotes a set of users, K denotes a set of keys, and $R \subset U \times K$ denotes a *user-key* relation, which specifies keys held by each user in U. In particular, each user is given a subset of keys that includes the user's individual key and a group key. Group key management can be performed by organizing the keys in K into a hierarchy and giving users additional keys. A trusted server is responsible for group access control and key management (i.e. for the distribution of keys to group members and maintenance of the userkey relation).

A key graph is a directed acyclic graph G with two types of nodes: *u-nodes* representing users and *k-nodes* representing keys. Each u-node has one or more outgoing edges but no incoming edge. Each k-node has one or more incoming edges. If a k-node has incoming edges only and no outgoing edge, then this k-node is called a *root*. Given a key graph G, it specifies a secure group (U, K, R) as follows:

(1) There is a one-to-one correspondence between U and the set of u-nodes in G.
(2) There is a one-to-one correspondence between K and the set of k-nodes in G.
(3) (u, k) is in R if and only if G has a directed path from the u-node that corresponds to u to the k-node that corresponds to k.

As an example, the key graph in Figure 1 specifies the following secure group:

$$\begin{aligned}
U &= \{u_1, u_2, u_3, u_4\}\\
K &= \{k_1, k_2, k_3, k_4, k_{12}, k_{234}, k_{1234}\}\\
R &= \{(u_1, k_1), (u_1, k_{12}), (u_1, k_{1234}),\\
&\quad (u_2, k_2), (u_2, k_{12}), (u_2, k_{234}), (u_2, k_{1234}),\\
&\quad (u_3, k_3), (u_3, k_{234}), (u_3, k_{1234}),\\
&\quad (u_4, k_4), (u_4, k_{234}), (u_4, k_{1234})\}
\end{aligned}$$

In our case, the users are represented by mobile devices that receive encrypted data stream from a satellite. The satellite also transmits the rekeying messages needed by the users to decrypt the broadcasted data stream, as in Figure 2. Anytime a user join/leave occurs, a RP containing all the keys to be delivered to the user group is computed and transmitted by the key server. Consistently, the main data stream is encrypted with the new key provided by means of the last RP itself. In practice, both the data stream and the rekeying messages are generated by a ground control station and transmitted to the satellite for broadcasting to the mobile users. In the following, for the sake of simplicity, we will assume that the satellite will provide the functions of the key server, although it is not necessarily the true originator of data. We also assume that users cannot transmit any data to the satellite, i.e. they do not have any feedback link. The latter assumption makes the key distribution more complicated, since users cannot request retransmission for not received RPs.

To be more specific, if an RP sequence, say $r_1, r_2, \ldots, r_m$, has been broadcasted, due to a number m of joins/leaves, any user needs to receive and decrypt all the m RPs to decrypt the

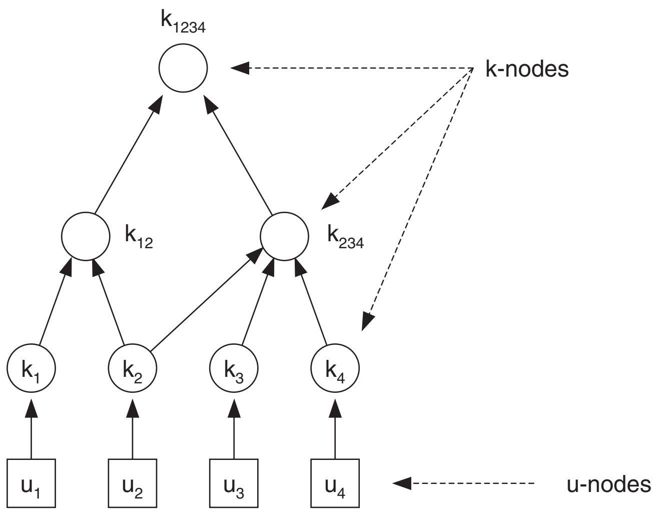

Figure 1. A key graph.

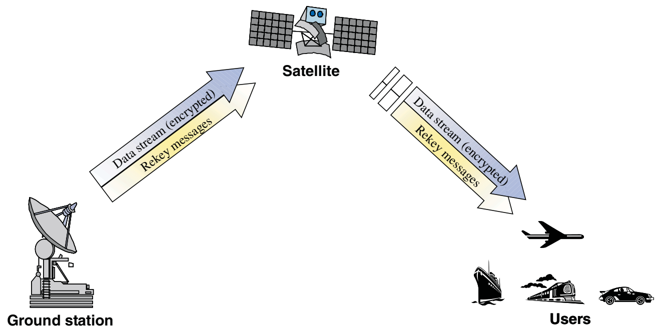

Figure 2. Data flow form the ground station to the users.

data broadcasted afterwards (and encrypted with the key provided by r_m). Therefore, if a user loses an intermediate RP, it is not able to decrypt all the next RPs, nor to decrypt any other broadcasted encrypted data.

Therefore, when a new RP (say, r_i) is broadcasted, we assume that only a subset of mobile users receives and decrypts it. In fact, for mobile users reception of data and RP are limited by several reasons like interference or shadowing due to physical obstacles. In order to increase the number of users that receive the new RP, the only solution is to broadcast it again several times, due to the absence of any feedback from the receivers, in a total open-loop fashion.

When another RP (r_{i+1}) is generated and broadcasted, only the users that correctly received packet r_i are able to get packet r_{i+1} and decrypt it, whereas all the others are not able to process it in any way. Moreover, among all the potential users that could process packet r_{i+1} (since they already processed packet r_i) only a subset of them will receive r_{i+1} due to location problems and transmission errors. As a consequence, packet r_{i+1} needs to be broadcasted again, to let more users receive it. In other words, both packets (r_i and r_{i+1}) need to be broadcasted more than once to increase the number of *updated users* (defined as the legitimate users that have received and decrypted all the RPs so that they are correctly assigned the up-to-date keys). Suitable 'statistics' (recall that a real-time monitoring of the situation is not possible due to the absence of feedback from the receivers) could represent the evolution of the user set when new RPs are broadcasted, showing the average percentages of legitimate users that have received and decrypted packets $r_i, r_{i+1}, r_{i+2}, \ldots$. On the basis of these statistics, one can address the problem of finding an optimal schedule to maximize the number of users that have received and decrypted all the rekeying data packets up to date (i.e. maximize the size of the set of *updated users*).

It is presumable that a percentage of receivers 'get lost'. Some of them, in fact, are not able to receive and decrypt so many RPs (i.e. a long time has passed since they received any RP) that they are not able to decrypt the broadcasted data any longer. When this happens, a recovering procedure is needed, which is out of the scope of this paper. Indeed, we consider the problem of finding an optimal schedule for the transmission of RPs in order to maximize the size of the updated users set.

4. THE ANALYTICAL MODEL

4.1. The scheduler

In this section we describe the system in charge of deciding which RP must be transmitted to users at a given time (i.e. the RP scheduler). This system is part of the key server (denoted by s) that generates two types of messages: the encrypted data (referred to as *main data stream*) and the rekeying messages (contained into the RPs) that the users need to decrypt the main data stream. Server s is actually hosted in a ground station transmitting all data to the satellite for broadcasting to mobile users, as shown in Figure 2. However, since the satellite is used only as a relay for data broadcasting, we can simplify the model assuming—as already mentioned in previous section and without losing of generality—that the server s broadcasts the main data stream, as well as RPs, directly to users.

A sketch of the scheduler, as well as its functional concept is shown in Figure 3. An *RP Generator* is responsible for the production of the RPs any time that a join or a leave occurs. RPs generated by the RP generator are stored in an *RP Buffer* that contains M slots (denoted by $b_1, b_2, \ldots, b_M$). When a new RP is generated, the RP in slot b_1 is dropped and all the RP stored in slots $b_2, \ldots, b_M$ are shifted by one slot in order to empty slot b_M, where the new RP is stored.

On the basis of a scheduling algorithm, the *Scheduler* picks one of the M RPs currently available in the RP buffer and transmits it (i.e. sends it to the satellite for broadcasting to the users). Recall that users are not able to provide any feedback link to the scheduler. Therefore, the scheduler algorithm is based on a user model referred to as *User population model* in Figure 3.

Note that this system acts in a complete open-loop fashion, since the feedback is not provided by the real set of user, but by a model. Nevertheless, leveraging this approach it is possible to elaborate a scheduling algorithm on the basis of the values provided by the user population model. The more accurate the model, the more accurate will be the results obtained by applying the scheduling algorithm.

In the following, we study the problem of computing a whole RP sequence aiming at maximizing the set of updated users, on the basis of the RPs currently stored in the buffer, and

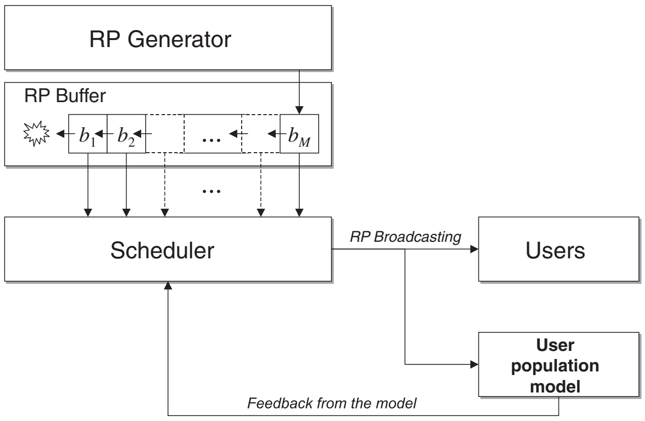

Figure 3. The scheduler and its functional concept.

on the modelled evolution of the user population model. When the transmission of the RP sequence is completed and a new RP is generated, a new optimal RP sequence can be computed. Clearly, the new sequence will not contain the RP previously stored in slot b_1 (dropped when a new RP is stored in slot b_M). This means that all users needing to receive the dropped RP will no longer be able to decrypt any other RP. We will refer to these users as *dead* users. In this case, a special intervention is needed to recover the situation and provide the dead users with the updated set of keys contained in the last generated RP. However, since the optimal RP sequence is computed between the generation of two RPs, we focus on the behavior of the scheduler (and of the user population model) during the transmission of an RP sequence. In order to derive the scheduling algorithm, it is fundamental to define the user population model.

4.2. The user population model

User population can be divided into $M+2$ classes $(x_{-1}, x_0, x_1, \ldots, x_M)$ denoting the state of the user receivers. The receiver state represents the number of the RP that the receiver has correctly received and processed (i.e. decrypted). Thus, a user is in state x_i if it decrypted RP r_i, and it is able to receive and decrypt (only) RP r_{i+1}, when it is broadcasted. We assume that the set of available RPs is constituted only by M elements, denoted by $r_1, r_2, \ldots, r_M$. A receiver being in state x_M has decrypted RP r_M (i.e. the most updated RP), and cannot evolve to any other state, unless a new RP r_{M+1} is generated by the scheduler. A user being in state x_0 is 'going to die', whereas a user being in state x_{-1} is 'dead', i.e. it is not able to decrypt any RP, since the set of available RPs contains only $r_1, \ldots, r_{M-1}, r_M$, which can be received only by users that are, respectively, in states $x_0, \ldots, x_{M-1}$.

For the sake of simplicity, we assume that RPs are transmitted at regular time intervals t_j $(j = 0, 1, 2, \ldots)$ with constant length[‡] T. Therefore, time instants t_j satisfy the following rules:

[‡]Note that this assumption can be easily removed since the proposed model is valid even in case the time intervals present variable length.

$t_j = jT$, and $t_{j+1} = t_j + T$, $\forall j = 0, 1, 2, \ldots$. Transmission of RP r_ℓ is denoted by the variable v_ℓ, with $\ell = 1, 2, \ldots, M$. Therefore, we let:

$$v_\ell(t_j) = \begin{cases} 1 & \text{if RP } r_\ell \text{ is transmitted at time } t_j \\ 0 & \text{otherwise} \end{cases}$$

Since one RP is always transmitted at every t_j, we have:

$$\sum_{\ell=1}^{M} v_\ell(t_j) = 1 \quad \forall t_j \tag{1}$$

thus showing that at each instant t_j only one variable $v_\ell(t_j)$ is equal to 1, while all the others are equal to zero. In the following, we denote that r_ℓ is transmitted at time t_j by using the compact notation $v(t_j) = \ell \in \mathrm{RPS}(M)$, where $\mathrm{RPS}(M)$ denotes the RP *set* containing the numbers corresponding to the available M RPs, that is,

$$\mathrm{RPS}(M) = \{\ell | \ell = 1, 2, \ldots, M\}$$

Furthermore, when it is not relevant to consider the instant t_j, we simply write $v = \ell$ to mean that RP r_ℓ is transmitted.

Generally, only an average fraction $p \in (0, 1)$ of the users being in state x_i is able to acquire RP r_{i+1} when it is transmitted.

Indeed, in practical satellite communications, several operating conditions must be taken into account, like varying fading and channel conditions, as well as different types of users, applications, and environments (e.g. hand-held, car or aircraft receivers, etc.). We assume that p (which is referred to as the *transition parameter*) represents the average percentage of the different users that are able to acquire RP r_{i+1}. We also assume that it can be statistically determined and it is the same for each state x_i ($i = 0, 1, \ldots, M$).

Therefore, at each instant t_j, when RP r_{i+1} is transmitted ($v_{i+1}(t_j) = 1$), only a fraction of the users being in state x_i evolves to state x_{i+1}, while the remaining fraction $(1 - p)$ remains in state x_i. As a consequence, for the generic state x_i, ($i = 1, \ldots, M - 1$), we have:

$$x_i(t_{j+1}) = x_i(t_j) - p x_i(t_j) v_{i+1}(t_j) + p x_{i-1}(t_j) v_i(t_j) \tag{2}$$

Equation (2) states that the number of receivers being in state x_i increases (decreases) if RP r_i (r_{i+1}) is transmitted.

Since no RP r_{M+1} is defined, for $i = M$, Equation (2) becomes:

$$x_M(t_{j+1}) = x_M(t_j) + p x_{M-1}(t_j) v_M(t_j) \tag{3}$$

whereas, since no RP r_0 is either defined, for $i = 0$, Equation (2) reads as follows:

$$x_0(t_{j+1}) = x_0(t_j) - p x_0(t_j) v_1(t_j) \tag{4}$$

Finally, for receivers in state x_{-1}, we have:

$$x_{-1}(t_{j+1}) = x_{-1}(t_j) \tag{5}$$

showing that users being in state x_{-1} have no possibility to evolve to any other state, independently of the transmitted RP. Note that, according to Equations (2)–(4), no user can move to state x_{-1} during the transmission of the RP sequence. In fact, this situation may happen only when the RP generator generates a new RP.

A block diagram of the user population model representing the state transitions defined by Equations (2)–(4) is shown in Figure 4, where transitions between user states on the basis of the

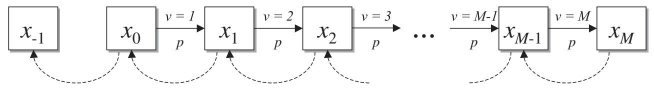

Figure 4. State transitions in the user population model.

value of variable v are represented by solid arrows. It is evident that only a fraction p of users being in any state can evolve to the next state. On the other hand, transitions occurring when a new RP is generated are represented with dashed lines, reproducing a state shift similar to the corresponding shift implemented in the RP buffer. Note also that when a user is in state x_{-1} (i.e. it is dead), no transitions to any other state is possible; therefore, state x_{-1} will not be taken into account in the analysis of the evolution of the user population model during the transmission of an RP sequence.

Let x denote the vector of states $x = [x_0, \ldots, x_M]^{\mathrm{T}}$; we can write the system represented by Equations (2)–(4) in the form:

$$x(j+1) = A(j)x(j) \tag{6}$$

where the transition matrix $A(j)$ is defined as follows:

$$A(j) = \begin{bmatrix} 1 - pv_1(j) & 0 & \cdots & 0 & 0 \\ pv_1(j) & 1 - pv_2(j) & & & \vdots \\ 0 & pv_2(j) & \ddots & 0 & \\ \vdots & & & 1 - pv_M(j) & 0 \\ 0 & 0 & \cdots & pv_M(j) & 1 \end{bmatrix}$$

where the instants t_j have been substituted by their index j (this compact notation will be maintained in the following).

From Equation (6) it is evident that transmission of RP r_i at instant j (i.e. $v_i(j) = 1$) affects the dynamics of the sole states x_{i-1} and x_i, thus having a 'local' effect in the evolution of the state vector. Note that the sum of the terms in each column of matrix $A(j)$ is equal to 1. This shows that only a fraction p of the users can evolve from one state to the next one. Equation (6) describes a discrete-time, switched, and positive dynamic system [26], where the switches are represented by variables $v_\ell(\cdot)$.

4.3. Optimization on the scheduler

In this section we consider the problem of finding an optimal scheduling for the transmission of RPs, by exploiting the scheduler concept, as well as the user population model.

Given a fixed number M of available RPs, the sequence length R, and given an initial condition $x(0) = [x_0(0), x_1(0), \ldots, x_M(0)]^{\mathrm{T}}$ for the user population, we want to find the RP transmission sequence $\{v(0), v(1), \ldots, v(R-1)\}$ in order to maximize the number of users in the state x_M at the end of the sequence. Therefore, we consider the following problem.

Optimization Problem

Given the user population model (6), the initial conditions $x(0)$, the number of states (and the related number M of RPs), the transition parameter p, and the number of instants (or, sequence steps) $R > M$, maximize the function:

$$\max_{\{v(j)\}_{j=0,1,\ldots,R-1}} x_M(R) \tag{7}$$

computing the optimal sequence $\{v^*(\cdot)\} = \{v^*(0), v^*(1), \ldots, v^*(R-1)\}$. □

Equation (7) could be solved by enumeration starting from Equation (6), finding off-line the optimal sequence and the value of the state $x_M(R)$ at instant R. However, the number of iterations is exponential and equal to M^R, as it is possible to choose among M available RPs at each of the R instants. Further, note that since no external intervention is envisaged in the dynamic system (6) other than the choice of the RP to be transmitted at each instant, the optimal sequence is affected by the initial conditions $x(0)$, as well as by the values of the parameters p, M, and R.

In order to solve the optimization function (7), it is necessary to devise an algorithm with low complexity. Therefore, an analysis of the optimal sequence characteristic is useful to reduce the number of admissible solutions.

The following theorem provides the *existence* as well as the *structure* of the optimal sequence (to ease reading, the proof is in the Appendix).

Theorem 1

Given the user population model (6), the number of states (and the related number M of RPs), and the number of instants (or, sequence steps) $R > M$, the sequence $\{v^*(\cdot)\} = \{v^* \times (0), v^*(1), \ldots, v^*(R-1)\}$ that maximizes function (7) is such that $v^*(R-1) = M$, and, for each $j = 2, \ldots, R$, $v^*(R-j) = \alpha$, with $\alpha \in \{v^*(R-j+1), v^*(R-j+1) - 1\} \subset \text{RPS}(M)$. □

Theorem 1 not only guarantees the existence of the optimal solution, but it also defines its structure. This result is not irrelevant, since it reduces dramatically the number of *admissible sequences* (i.e. optimal sequence candidates), thus simplifying the search for the optimal one. Indeed, a consequence of Theorem 1 is that the number of admissible sequences is reduced from M^R to 2^{R-1}, since all the admissible sequences $\{v(\cdot)\}$ can be represented by means of a binary tree (see Figure 5) which is constructed starting from the RP to be transmitted at the last step (i.e. starting from the value of $v^*(R-1) = M$).

Furthermore, since $R > M$, the number of admissible sequences is far less than 2^{R-1}, since the RP set RPS(M) is limited to M elements only. As an example, refer to Figure 6, where the tree of admissible sequences has been drawn considering $M = 3$ RPs and a sequence length $R = 5$. As it can be seen, since $M = 3$, RPs corresponding to $v(\cdot) = (M-3) = 0$ and $v(\cdot) = (M-4) = -1$ are not defined, since they are not included in RPS(M), thus reducing the total number of admissible sequences from 16 to 11. Generally speaking, the admissible sequences cannot contain RPs r_{M-j}, with $j \geqslant M$, which are not defined.

It is also straightforward to see that the set of admissible sequences is further reduced if $R \gg M$. Given M and R, the exact number of admissible sequences is provided by the following theorem (to ease reading, the proof is in the Appendix).

Theorem 2

Given the user population model (6), the number of states (and the related number of RPs M) and the number of instants (or, sequence steps) $R > M$, the total number of admissible sequences having the structure defined in Theorem 1 is

$$\sum_{i=0}^{M-1} \frac{(R-1)!}{i!(R-i-1)!} \ll 2^{R-1}$$ □

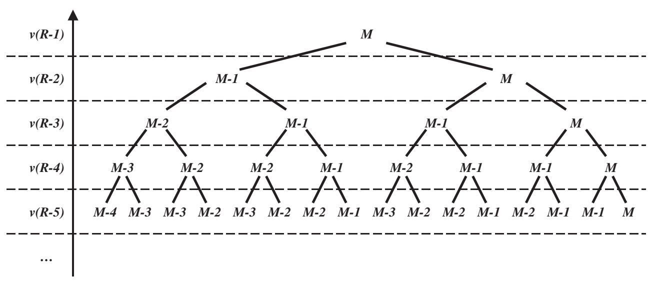
v(R-1)
v(R-2)
v(R-3)
v(R-4)
v(R-5)
...
M
M-1 M
M-2 M-1 M-1 M
M-3 M-2 M-2 M-1 M-2 M-1 M-1 M
M-4 M-3 M-3 M-2 M-3 M-2 M-2 M-1 M-3 M-2 M-2 M-1 M-2 M-1 M-1 M

Figure 5. The admissible sequence tree in the general case.

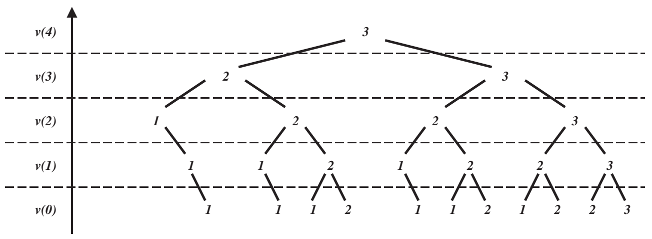

Figure 6. The admissible sequence tree when $M = 3$ and $R = 5$.

Thanks to Theorem 2, it is possible to evaluate the number of admissible sequences among which to find the optimal solution. Table I shows how Theorem 2 simplifies the complexity of the problem for some values of sequence length R and number M of RPs. On the basis of Theorem 2, a branch-and-bound algorithm can be implemented to optimize the search of the optimal sequence in the admissible sequence tree.

5. SIMULATION RESULTS AND DISCUSSION

In this section we present how the theoretical results described in the previous sections apply to specific situations by simulating the population model and implementing the scheduling algorithm. We used the MatLab® environment to this purpose.

In the following we show the results obtained for different scenarios characterized by different values of the *number M of RPs*, the *sequence length R*, and the transition parameter p, showing how the total number of possible sequences is dramatically reduced by the optimization algorithm on the basis of Theorem 2. Further, for each scenario, we present a set of simulations with respect to different initial conditions of the user population (i.e. the values of the states $x_0(0), x_1(0), \ldots, x_M(0)$), providing the computed optimal sequence, and the value of $x_0^*(R), x_1^*(R), \ldots, x_M^*(R)$ obtained after the transmission of the optimal sequence.

Table I. Number of total sequences (M^R) and admissible sequences (as from Equation (13)) with respect to different cases having sequence length R and RP number M.

		Number of RPs (M)			
(R)		2	4	8	10
4	Total seq.	16	256	4096	10 000
	Reduced seq.	4	8	—	—
8	Total seq.	256	65 536	16.7×10^6	10^8
	Reduced seq.	8	64	128	—
10	Total seq.	1024	1 048 576	1.1×10^9	10^{10}
	Reduced seq.	10	130	502	512
16	Total seq.	65 536	4.3×10^9	2.8×10^{14}	10^{16}
	Reduced seq.	16	576	16 384	27 824
20	Total seq.	1 048 576	1.1×10^{12}	1.15×10^{18}	10^{20}
	Reduced seq.	20	1160	94 184	262 144
40	Total seq.	1.1×10^{12}	1.2×10^{24}	1.3×10^{36}	10^{40}
	Reduced seq.	40	9920	1.9×10^7	2.9×10^8

For the sake of simplicity, and in order to ease the presentation of the results, we assume that the total number of users n is constant and normalized to 1. We have therefore the following assumption:

$$n = \sum_{i=0}^{M} x_i(j) = 1 \quad \forall j = 0, 1, \ldots, R$$

5.1. Scenario 1

In the first case we consider the behavior of the optimizing algorithm in the presence of the following parameters:

# of RPs (M)	Sequence length (R)	Transition parameter (p)
4	16	0.97

characterized by a very high value of the transition parameter p. With these parameters, we have the following possible sequences, reduced by the optimizing algorithm:

Total number of sequences (M^R)	4.294967×10^9
Number of possible sequences (2^{R-1})	32768
Reduced number of admissible sequences, Equation (13)	576

This shows how, in this specific case, the number of admissible cases is reduced from $4.294\ 967 \times 10^9$ to 576, as from Equation (13). The complexity of the search for the optimal sequence is thus reduced by 1.34×10^7.

5.1.1. Scenario 1.1. In the first scenario we show the effectiveness of our approach in the case most of the users have the chance to decrypt each transmitted RPs (i.e. we have a high value of $p = 0.97$).
In this situation we assume that the user population has the following initial distribution:

$x_0(0)$	1
$x_1(0)$	0
$x_2(0)$	0
$x_3(0)$	0
$x_4(0)$	0

Transmitting the optimal sequence computed by the algorithm:

$$v^*(\cdot) = \{1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4\}$$

yields to the optimal final state:

$x_0^*(R)$	0.00000081000000
$x_1^*(R)$	0.00000080999934
$x_2^*(R)$	0.00000080999869
$x_3^*(R)$	0.00000080999803
$x_4^*(R)$	0.99999676000394

thus showing how the algorithm is able to transfer at least the 99.9996% of the population from state x_0 to state x_4, thus proving the effectiveness of our approach. This is considerably remarkable when compared with the solution obtained by transmitting the trivial sequence $\{v(\cdot)\} = \{1, 2, 3, 4\}$. In the latter case, in fact, the following final state would be attained:

$x_0(M)$	0.03000000
$x_1(M)$	0.02910000
$x_2(M)$	0.02822700
$x_3(M)$	0.02738019
$x_4(M)$	0.88529281

thus showing that, using our optimizing algorithm, an 11.5% gain in the amount of users transferred from state x_0 to state x_4 is obtained.

5.1.2. Scenario 1.2. In this second case, we consider the behavior of the algorithm in the same hypothesis of Scenario 1.1, but with $p = 0.25$, and the user population having the following initial distribution:

$x_0(0)$	1/64
$x_1(0)$	0
$x_2(0)$	63/64
$x_3(0)$	0
$x_4(0)$	0

Transmitting the optimal sequence computed by the algorithm:

$$v^*(\cdot) = \{3,3,3,3,3,3,3,3,4,4,4,4,4,4,4,4\}$$

yields to the optimal final state:

$x_0^*(R)$	0.01562500000000
$x_1^*(R)$	0
$x_2^*(R)$	0.09854865074158
$x_3^*(R)$	0.08868265804267
$x_4^*(R)$	0.79714369121575

This second case shows how the low value of parameter p forces the algorithm to focus only on the users initially located in state x_2. In fact, transferring even a small amount of users initially located in state x_0 would require too much effort, which would compromise the majority of users. This explains why the optimal sequence starts with the transmission of r_3. Note, however, that, although parameter $p = 0.25$, about 80% of the users are finally transferred to state x_4.

5.2. Scenario 2

This scenario is characterized by the following parameters:

# of RPs (M)	Sequence length (R)	Transition parameter (p)
3	20	0.8

In this case, we have the following possible sequences, reduced by the optimizing algorithm:

Total number of sequences (M^R)	3.4868×10^9
Number of possible sequences (2^{R-1})	524288
Reduced number of admissible sequences, Equation (14)	191

This shows how, in this specific case, the number of admissible cases is reduced from 3.4868×10^9 to 191, as from Equation (13). The complexity of the search for the optimal sequence is thus reduced by 5.48×10^8.

In the following we show the application of our algorithm with respect to different initial conditions of the user population.

5.2.1. Scenario 2.1. In this situation we assume that the user population has the following initial distribution:

$x_0(0)$	1
$x_1(0)$	0
$x_2(0)$	0
$x_3(0)$	0

Transmitting the optimal sequence computed by the algorithm:

$$v^*(\cdot) = \{1,1,1,1,1,1,2,2,2,2,2,2,2,3,3,3,3,3,3,3\}$$

yields to the optimal final state:

$x_0^*(R)$	0.00006400000000
$x_1^*(R)$	0.00001279918080
$x_2^*(R)$	0.00001279901697
$x_3^*(R)$	0.99991040180223

This case is similar to Scenario 1.1. In fact all the users are initially distributed in state x_0. This yields to the pretty balanced optimal sequence computed by the algorithm. Different from Scenario 1.1, however, sequence length R is not a multiple of the number M of RPs, and this explains why RP r_1 is retransmitted only six times whereas r_2 and r_4 are transmitted seven times each.

5.2.2. Scenario 2.2. In this situation we assume that the user population has the following initial distribution:

$x_0(0)$	1/256
$x_1(0)$	255/256
$x_2(0)$	0
$x_3(0)$	0

Transmitting the optimal sequence computed by the algorithm:

$$v^*(\cdot) = \{1,1,1,1,2,2,2,2,2,2,2,2,3,3,3,3,3,3,3,3\}$$

yields to the optimal final state:

$x_0^*(R)$	0.00000625000000
$x_1^*(R)$	0.00000255998400
$x_2^*(R)$	0.00000255997745
$x_3^*(R)$	0.99998863003855

This case is similar to Scenario 1.2. In fact, since the majority of users are initially distributed in state x_1, RP r_1 is transmitted only four times. Indeed, the algorithm focuses on transferring the users from state x_1 to state x_3 transmitting a higher number of RP r_2 and r_3.

5.2.3. Scenario 2.3. In this situation we assume that the user population has the following initial distribution:

$x_0(0)$	1/256
$x_1(0)$	0
$x_2(0)$	255/256
$x_3(0)$	0

Transmitting the optimal sequence computed by the algorithm:

$$v^*(\cdot) = \{1,1,1,1,1,1,2,2,2,2,2,3,3,3,3,3,3,3,3,3\}$$

yields to the optimal final state:

$x_0^*(R)$	0.00000025000000
$x_1^*(R)$	0.00000124992000
$x_2^*(R)$	0.00000051199923
$x_3^*(R)$	0.99999798808077

This scenario is pretty unrealistic due to the initial absence of users in state x_1, while most of them are distributed in state x_2. Nevertheless, from an algorithmic point of view, it is particularly interesting. In fact, one could expect that the optimal sequence could focus on the transmission of RP r_3, thus following the same principle shown in Scenario 1.2. The fact that the majority of users is initially located in state x_2, i.e. just one step away from the goal state x_3, lets the algorithm focus also on the users initially located in state x_0. As a result, a larger number of users (when compared with the previous scenario) are finally in state x_3.

6. CONCLUSIONS

In this work, we have addressed the problem of maximizing the access to services for mobile users belonging to the same security group, with no access to a feedback link. The solution provided is the design of a scheduling algorithm that computes the rekeying sequence that maximizes the set of users with the most updated RP. To this end, we analyzed the problem by defining a user population model to be used for the design of the scheduling algorithm. On the basis of the user population model analysis, optimal rekeying sequences have been proven to show a special structure that can be used by the scheduling algorithm to reduce the set of admissible sequences. Thanks to the structure of the optimal sequences, the sequence tree has also been proven to be further prunable, thus considerably reducing the set of optimal sequences and, consequently, the computational burden of the scheduler. The theoretical results have been supported by simulations, thus proving the effectiveness of our approach, and showing the scheduler behavior in several interesting cases.

This work paves the way for future works along different directions: the problem can be further specialized to the logic key hierarchy approach. In fact, scheduling of single rekeying messages instead of RPs can be considered, thus providing a finer resolution scheduling. Finally, a different user population model could be adopted in order to detail user behavior. For instance, the set of users could be subdivided into different classes on the basis of their characteristics, e.g. taking into account different values for the transition parameter p.

ACKNOWLEDGEMENTS

The authors are solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organization. The authors would like to thank the anonymous reviewers for their detailed comments that helped to substantially improve the presentation of this paper. This work was partly supported by the Spanish Ministry of Science and Education through projects TSI2007-65406-C03-01 'E-AEGIS' and CONSOLIDER CSD2007-00004 'ARES', and by the Government of Catalonia under grant 2005 SGR 00446; the EU FP6 projects: 'European Satellite Communications Network of Excellence' (SatNEx II, IST-27393); PERSONA (contract N. 045459); INTERMEDIA (contract N.38419).

APPENDIX

Theorem 1

Given the user population model (6), the number of states (and the related number M of RPs) and the number of instants (or, sequence steps) $R > M$, the sequence $\{v^*(\cdot)\} = \{v^*(0), v^* \times (1), \ldots, v^*(R-1)\}$ that maximizes function (7) is such that $v^*(R-1) = M$ and, for each $j = 2, \ldots, R$, $v^*(R-j) = \alpha$, with $\alpha \in \{v^*(R-j+1), v^*(R-j+1) - 1\} \subset \mathrm{RPS}(M)$.

Proof

First, we show that $v^*(R-1) = M$. In fact, if it is not the case, there would exist another optimal sequence $\{v(\cdot)\} \neq \{v^*(\cdot)\}$, with length shorter than R.

Going backwards, it is possible to show that $v^*(R-2)$ can only assume either the value of $v^*(R-1) = M$ or of $v^*(R-1) - 1 = M - 1$. In fact, since $v^*(R-1) = M$, from model (6), it follows that:

$$x_M(R-1) = p x_{M-1}(R-2) + x_M(R-2)$$

Therefore, to maximize $x_M(R-1)$, an RP should be transmitted to increment the contribution given either by the term $x_{M-1}(R-2)$ (i.e. choosing $v^*(R-2) = M-1$) or by the term $x_M \times (R-2)$ (i.e. choosing $v^*(R-2) = M$). These considerations may be extended, in a backward fashion, to all instants $j = 3, \ldots, R$, thus proving the special structure of the optimal sequence $\{v^*(\cdot)\}$. The existence of the optimal sequence is trivially guaranteed by the assumption $R > M$. In fact, this allows that all RPs may be transmitted at least once, thus yielding to a value of $x_M(R) > x_M(0)$ for any initial condition $x(0)$. □

Theorem 2

Given the user population model (6), the number of states (and the related number of RPs M) and the number of instants (or, sequence steps) $R > M$, the total number of admissible sequences having the structure defined in Theorem 1 is

$$\sum_{i=0}^{M-1} \frac{(R-1)!}{i!(R-i-1)!} \leqslant 2^{R-1}$$

Proof

In order to compute the number of admissible sequences, let $n_{M-i}(R-j)$, $i = 0, \dots, M-1$, $j = 1, \dots, R$, be the number of possible RPs r_{M-i} to be transmitted at step $R-j$.

On the basis of Theorem 1, the number n_{M-i} of RPs that can be transmitted at each instant $(R-j)$ depends on the number n_{M-i} and n_{M-i-1} of RPs that can be transmitted at the next step $(R-j+1)$. Therefore, n_{M-i} can recursively be expressed as follows:

$$n_{M-i}(R-j) = n_{M-i}(R-j+1) + n_{M-i-1}(R-j+1) \tag{A1}$$

with $n_{M-i}(R-1) = 0\ \forall i = 1, \dots, M-1$. Note also that $n_M(R-j) = 1\ \forall j = 1, \dots, R$, since the term $n_{M+1}(R-j+1)$ in Equation (A1) is equal to zero (in fact, RP r_{M+1} is not defined).

The way the terms in Equation (A1) are defined suggests that they are computed by means of binomial coefficients. Indeed, they follow the principle of the Tartaglia triangle, where each term (starting from 1) is defined as the sum of the two terms above itself. As a matter of fact, it is

$$n_{M-i}(R-j) = \binom{j-1}{i} = \frac{(j-1)!}{i!(j-i+1)!} \tag{A2}$$

As a consequence of Theorem 1, and of the binary structure of the admissible sequence tree shown in Figure 5, the total number of possible RPs to be transmitted at the generic step $(R-j)$ is

$$\sum_{i=0}^{j-1} n_{M-i}(R-j) = 2^{j-1}, \quad j = 1, \dots, R \tag{A3}$$

which is also consistent with the rules regarding the sum of the elements in any jth row in the Tartaglia triangle.

Nevertheless, taking into account that $n_{M-i}(R-j) = 0$ for $i = M, M+1, \dots, R-1, R$, Equation (A3) can be re-written as follows:

$$\sum_{i=0}^{M-1} n_{M-i}(R-j), \quad j = 1, \dots, R \tag{A4}$$

or, considering Equation (A2),

$$\sum_{i=0}^{M-1} n_{M-i}(R-j) = \sum_{i=0}^{M-1} \binom{j-1}{i} = \sum_{i=0}^{M-1} \frac{(j-1)!}{i!(j-i+1)!}, \quad j = 1, \dots, R \tag{A5}$$

Equation (A5) represents the number of RPs that can be transmitted at each step $(R-j)$, $j = 1, \dots, R$. In order to compute the total number of admissible sequences of length R, it suffices to let $j = R$. Therefore, it is

$$\sum_{i=0}^{M-1} n_{M-i}(0) = \sum_{i=0}^{M-1} \binom{R-1}{i} = \sum_{i=0}^{M-1} \frac{(R-1)!}{i!(R-i-1)!} \ll 2^{R-1} \tag{A6}$$

The right-hand-side of Equation (A6) is justified by the fact that $M < R$ and that, by definition of binomial coefficients and from Equation (A3), the following holds (for $M = R$):

$$\sum_{i=0}^{R-1} \frac{(R-1)!}{i!(R-i-1)!} = 2^{R-1} \qquad \square$$

REFERENCES

1. Cruickshank H, Howarth M, Iyengar S, Sun Z. A comparison between satellite dvb conditional access and secure ip multicast. *14th IST Mobile and Wireless Communications Summit*, 19–22 June 2005, Dresden, Germany.
2. Wu S, Banerjee S, Hou X. A comparison of multicast feedback control mechanisms. *ANSS '05*: *Proceedings of the 38th Annual Symposium on Simulation*. IEEE Computer Society: Washington, DC, U.S.A., 2005; 80–87.
3. Zhang X, Shin KG. Delay analysis of feedback-synchronization signaling for multicast flow control. *IEEE/ACM Transactions Network* 2003; **11**(3):436–450.
4. Setia S, Koussih S, Jajodia S, Harder E. Kronos: a scalable group re-keying approach for secure multicast. *SP '00*: *Proceedings of the 2000 IEEE Symposium on Security and Privacy*. IEEE Computer Society: Washington, DC, U.S.A., 2000; 215.
5. Setia S, Zhu S, Jajodia S. A comparative performance analysis of reliable group rekey transport protocols for secure multicast. *Performance Evaluation* 2002; **49**(1–4):21–41.
6. Fu A, Modiano E, Tsitsiklis JN. Optimal transmission scheduling over a fading channel with energy and deadline constraints. *IEEE Transactions on Wireless Communications* 2006; **5**(3):630–641.
7. Naor D, Naor M, Lotspiech JB. Revocation and tracing schemes for stateless receivers. *CRYPTO '01*: *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*. Springer: London, U.K., 2001; 41–62.
8. Wang CK, Gouda M, Lam SS. Secure group communications using key graphs. *IEEE/ACM Transactions Network* 2000; **8**(1):16–30.
9. Chen W, Dondeti LR, Sun Y. Performance comparison of stateful and stateless group rekeying algorithms. *International Journal of Computer Science and Network Security* 2008; **8**(3):186–195.
10. Mittra S. Iolus: a framework for scalable secure multicasting. *SIGCOMM Computer Communication Review* 1997; **27**(4):277–288.
11. Hardjono T, Cain B, Monga I. Intra-domain group key management protocol. *IETF Internet Draft* (November 2000).
12. Fiat A, Naor M. Broadcast encryption. *Advances in Cryptology—CRYPTO'93*. *13th Annual International Cryptology Conference*. *Proceedings*. Lecture Notes in Computer Science, vol. 773, Springer: Berlin, 1993.
13. Luby M, Staddon J. Combinatorial bounds for broadcast encryption. *Advances in Cryptology*, *EUROCRYPT 1998*. Lecture Notes in Computer Science, vol. 1403. Springer: Berlin, 1998.
14. Blundo C, Frota Mattos LA, Stinson D. Trade-offs between communication and storage in unconditionally secure schemes for broadcast encryption and interactive key distribution. *Advances in Cryptology—CRYPTO'96*. *16th Annual International Cryptology Conference*. *Proceedings*. Lecture Notes in Computer Science, vol. 1109. Springer: Berlin, 1996.
15. Canetti R, Malkin T, Nissim K. Efficient communication-storage tradeoffs for multicast encryption. *Advances in Cryptology*, *EUROCRYPT 1999*. Lecture Notes in Computer Science, vol. 1592. Springer: Berlin, 1999.
16. Snoeyink J, Suri S, Varghese G. A lower bound for multicast key distribution. *Proceedings of IEEE INFOCOM 2001*: *Conference on Computer Communications*, 22–26 April 2001, Anchorage, Alaska.
17. Di Pietro R, Mancini LV, Mei A. Hierarchies of Keys in Secure Multicast Communications. *Journal of Computer Security*. To appear.
18. Di Pietro R, Durante A, Mancini LV. A reliable key authentication scheme for secure multicast communications. *Proceedings of the 22nd IEEE Symposium on Reliable and Distributed Systems*, Florence, Italy. IEEE Press: New York, 6–8 October 2003; 231–240.
19. Perrig A, Song DX, Tygar JD. A new protocol for efficient large-group key distribution. *SP '01*: *Proceedings of the 2001 IEEE Symposium on Security and Privacy*. IEEE Computer Society: Silver Spring, MD, 2001; 247–262.
20. Wong K, Gouda M, Lam SS. A group key management service. *Proceedings of International Conference on Telecommunications*, *ICT 2000*, May 2000, Acapulco, Mexico.
21. Staddon J, Miner S, Franklin M, Balfanz D, Malkin M, Dean D. Self-healing key distribution with revocation. *SP '02*: *Proceedings of the 2002 IEEE Symposium on Security and Privacy*. IEEE Computer Society: Washington, DC, U.S.A., 2002; 241.
22. Liu D, Ning P, Sun K. Efficient self-healing group key distribution with revocation capability. *CCS '03*: *Proceedings of the 10th ACM Conference on Computer and Communications Security*. ACM Press: New York, NY, U.S.A., 2003; 231–240.
23. Di Pietro R, Mancini LV, Mei A. Key management for high bandwidth secure multicast. *Journal of Computer Security* 2004; **12**(5):693–709.
24. Howarth M, Iyengar S, Sun Z, Cruickshank H. Dynamics of key management in secure satellite multicast. *IEEE Journal on Selected Areas in Communications* 2004; **22**(2):308–319.
25. Di Pietro R, Iannitti S. Optimal key scheduling for secure satellite broadcasting to mobile users. *IEEE International Workshop on Satellite and Space Communications* (*IWSSC'06*), 14–15 September 2006, Universidad Carlos III de Madrid, Leganés, Spain; 34–38.
26. Luenberger DG. *Introduction to Dynamic Systems—Theory, Models, and Applications*. Wiley: New York, 1979.

AUTHORS' BIOGRAPHIES

Roberto Di Pietro is currently an Assistant Professor at the Department of Mathematics of Università di Roma Tre, Roma, Italy. He received the PhD in Computer Science from the Università di Roma 'La Sapienza', Italy, in 2004. In 2004 he also received from the Department of Statistics of the same University a Specialization Diploma in Operating Research and Strategic Decisioning. He received a Laurea degree in Computer Science from the Universitá di Pisa, Italy, in 1994. Since 1995 he has been working for the technical branch of the Italian Army and the Internal Affairs Ministry.

His main research interests include: security and privacy for mobile, *ad hoc*, and underwater wireless networks; security and privacy for distributed systems, secure multicast, applied cryptography, and computer forensics.

Stefano Iannitti received the Laurea degree in Computer Engineering in 1998 and the PhD in Systems Engineering in 2002, both from the Università di Roma 'La Sapienza', Roma, Italy. In 2002 and 2003 he has been a visiting fellow at the Mechanical Engineering Department of NorthWestern University, Evanston (IL) U.S.A., where he held a post-doc position. Since 2003 he is with the Italian Space Agency (ASI), where he works on Italian and European space projects on Satellite Telecommunication and Navigation. In 2006 he received a Master in Information Security from the Università di Roma 'La Sapienza', Roma, Italy.

His main research interests are in the area of nonlinear control systems. Recent interests and activities include the analysis and design of satellite navigation systems, with an emphasis on security issues, as well as the application of nonlinear control to enhance network and space system security.