Scalable Delay-constrained Multicast Group Key Management

Saïd Gharout, Yacine Challal, Abdelmadjid Bouabdallah

Scalable Delay-constrained Multicast Group Key Management

Saïd Gharout[1], Yacine Challal[2], and Abdelmadjid Bouabdallah[2]
(Corresponding author: Saïd Gharout)

Computer Science Department, Bejaia University[1]
Targa Ouzemmour 06000 Bejaia, Algeria (Email: gharout@gmail.com)
Networking Group of the Heudiasyc Lab, Compiègne University of Technology[2]
BP 20529, 60205, Compiègne Cedex, France

(Received May 7, 2006; revised and accepted Sep. 19, 2006)

Abstract

In the last few years, multicasting is increasingly used as an efficient communication mechanism for group-oriented applications in the Internet. Some multicast applications require confidentiality for transmitted data. So, a traffic encryption key is used to assure this confidentiality and has to be changed and distributed to all valid members whenever a membership change (join or leave) occurs in the group. The bandwidth used for re-keying operations could be high when the group size is large. To cope with this limitation, many solutions propose to organize group members into subgroups that use independent traffic encryption keys in order to mitigate the scope of key management and thereby to scale better to large groups. Unfortunately, these solutions require the decryption and re-encryption of multicast messages whenever they pass from one subgroup to another. Moreover, the decryption / re-encryption operations induce delays in packet delivery throughout the delivery path. In order to avoid delays in packet delivery and perturbations caused by re-keying, we propose in this paper an adaptive solution for key management which organizes group members into dynamic and homogeneous clusters according to the application level requirements. First, we show that partitioning the group into clusters of subgroups that use independent traffic encryption keys can be formulated as tree partitioning problem. Then, we propose a protocol to solve the problem with respect to the application requirements and membership behavior. We conducted several simulations of the proposed protocol and the obtained results show that our solution is efficient and achieves better performance trade-offs compared to other schemes.

Keywords: Dynamism, key management, multicat, scalability, security

1 Introduction

The advantages of IP multicast in multi-party communications, such as saving bandwidth, simplicity and efficiency, are very interesting for new services combining voice, video and text over Internet [18, 44]. This urges the effective large scale deployment of multicasting to satisfy the increasing demand for multicasting from both Internet Service Providers (ISPs) and Content Distributors [3, 19]. Unfortunately, the strengths of IP multicast are also its security weaknesses. Indeed, the open and anonymous membership and the distributed nature of multicasting are serious threats to the security of this communication model. Much effort has been conducted to address the many issues relating to securing multicast data transmission, such as: *access control, confidentiality, authentication and watermarking* [5, 34]. The MSEC Working Group of the Internet Engineering Task Force (IETF) worked on the securing and deployment of IP Multicast [9].

Group communication confidentiality requires that only valid users could decrypt the multicast data even if the data is broadcast to the entire network. We consider in what follows that data is encrypted to ensure confidentiality using a symmetric cryptosystem (such as DES [25], 3DES [4] or AES [27]). Public key encryption algorithms such as RSA [48] or DSA [26] are rarely used to encrypt exchanged data in multicast because they are too late in relation to symmetric algorithms. Thus, a symmetric key is used to encrypt data by the source and to decrypt it by receivers. This key is generally called Traffic Encryption Key (TEK). The confidentiality requirements can be translated into two key distribution rules [53]:

- Forward secrecy: users which left the group should not have access to any future exchanged data. This

ensures that a member cannot decrypt data after it leaves the group.

- Backward secrecy: a new user that joins the session should not have access to any old key. This ensures that a member cannot decrypt data sent before it joins the group.

In order to meet the above requirements, a re-keying process should be triggered after each join/leave to/from the secure group. It consists in generating a new TEK and distributing it to the group members including the new one in case of a join or to the residual members in case of a leave. This process ensures that a new member can not decrypt previously exchanged messages and prevents a leaving member from eavesdropping future messages. A critical problem with any re-key technique is scalability: as the re-key process should be triggered after each membership change, the number of TEK update messages may be important in case of frequent join and leave operations. Some solutions propose to organize the secure group into subgroups with different local TEKs. This reduces the impact of the key updating process, but needs decryption and re-encryption operations at the borders of subgroups. These operations may decrease the communication quality.

The rest of the paper is organized as follows: In Section 2, we present a taxonomy of group key management protocols. We present principles of our protocol and model it in Section 3. In Section 4 we present some required definitions. In Section 5, we study costs induced by our protocol. In Section 6, we formalize the protocol. In Section 7, We present simulation results and comparison of our protocol with other protocols from literature. We conclude the paper in Section 8.

2 Group Key Management Protocols

Group key management has been extensively studied in the literature. Judge and Ammar [34], Rafaeli and Hutchison [46], Zhu and Jajodia [60] surveyed some group key management solutions. In this section we present relevant group key management protocols. In table 1 we summarize some existing solutions in group key management. Existing key management solutions could be classified into three categories: centralized, decentralized and distributed architectures.

2.1 Centralized Architectures

In this approach, the key distribution function is assured by a single entity which is responsible for generating and distributing the traffic encryption key (TEK) whenever required. The most proposed centralized protocols in the literature use a common *Traffic Encryption Key (TEK)* for group members. Two techniques are used to ensure *Key Management*: *Pairewise Keys* and *hierarchy of keys*. In the **pairewise keys** the *key server* shares a secret key with each group member. These pairewaise secret keys are generally called *Key Encryption Keys (KEK)* and are used to establish secure channels between the key server and each group member in order to re-distribute the *TEK* securely whenever required. In dynamic groups, a new *TEK* is sent to valid group members encrypted with their respective *KEKs*, including the new member in case of a join (to ensure *backward secrecy*), and excluding the leaving member in case of a leave (to ensure *forward secrecy*). A typical solution that fits into this category is the *Group Key Management Protocol (GKMP)* proposed by Harney and Muckenhirn in [29, 30]. Similar solutions are those proposed by Dunigan and Cao in [24] and Poovendram et al. in [43]. This scheme has the drawback to require a high number of update messages (in the order of $O(n)$ with n being the number of valid group members) to transmit the new TEK after membership changes.

The aim of the **hierarchy of keys** approach is to reduce the required number of *TEK update messages* induced by re-keying after membership changes. Therefore, in contrast to the *pairewise keys* approach, the key server shares secret keys with subgroups of the entire secure group in addition to the individual secured channels. The *Logical Key Hierarchy (LKH)* protocol proposed at same time by Wong et al. in [57, 58] and Wallner et al. in [56], is a typical solution fitting into this category. The intermediate keys shared with different combinations of subgroups form a hierarchy (generally a binary tree) of keys. The number of update messages induced by this protocol is in the order of $\log(n)$ with n being the number of valid group members. *One-way Function Trees (OFT)* [6, 36], the *One-way function chain tree* [12], and the *Efficient Large group Key distribution (ELK)* [42] are variants of *LKH* protocol that allow to save some update message transmissions of intermediate keys of the hierarchy by replacing them with *one-way function* computations. In *Centralized Flat table Key Management (CFKM)* [55], the key hierarchy is replaced by a *flat table* and allows hence to reduce the number of keys maintained by the *Key Server*.

2.2 Decentralized Architectures

In this category, a hierarchy of key managers share the labor of distributing the TEK to group members in order to avoid bottlenecks and single point of failure.We can distinguish protocols that use common TEK for the whole group and protocols that use a common TEK per subgroup.

2.2.1 Common TEK Protocols

Ballardie proposed in RFC1949 [7] the *Scalable Multicast Key Distribution (SMKD)*; a protocol where the *main core* of the multicast tree (constructed using *CBT* routing protocol [8]) mandates the *secondary cores* and other trusted routers to *propagate* the distribution of the TEK.

Table 1: Group key management protocols

Centralized	*Decentralized*	*Distributed*
GKMP [29, 30]	SMKD [7]	Ingemarson et al. [33]
Poovendran et al. [43]	IGKMP [28, 16]	GDH [54]
Dunigan and Cao [24]	Hydra [45]	DH-LKH [41, 35]
LKH [57, 58]	MARKS [11]	D-LKH [49]
OFT [36, 6]	Kronos [51]	D-OFT [20]
Canetti et al. [12]	DEP [23]	D-CFKM [55]
ELK [42]	Iolus [37]	FTGDH [50]
CFKM [55]	KHIP [52]	
	Cipher Sequences [38]	
	Yang et al. [59]	
	SAKM [13]	
	SIM-KM [40]	
	Mykil [31]	

This protocol has the drawback to be routing dependent. DeCleene et al. [16, 28] proposed the *Intra-domain Group Key Management Protocol (IGKMP).* In this architecture, the network is organized into administratively scoped areas in which a Domain Key Distributor (DKD) and many Area Key Distributors (AKD) are defined. Each AKD is responsible for one area. The DKD generates the TEK and multicasts it to the set of AKDs. When an AKD receives the TEK it propagates it to the group members of its area. In *Hydra* protocol (Rafaeli et al. [45]), the group is organized into subgroups, and each subgroup i is controlled by a server called the Hydra server (HS_i). If a membership change occurs at subgroup i, the corresponding HS_i generates the group TEK and sends it to the other HS_js involved in that session. Setia et al. proposed the *Kronos* protocol [51], where the whole domain is divided into smaller areas managed by different *Area Key Distributors (AKDs).* After each specific period of time, each AKD generates a new TEK and distributes it to the members of its area. The AKDs share some secret parameters that allow them to generate the same TEK after each time period. In MARKS [11], Briscoe suggests slicing the time length to be protected into small portions of time and using a different key for encrypting each slice. The encryption keys are the leaves in a binary hash tree that is generated from a single seed. A blinding function, such as MD5 [47] is used to create the tree nodes. Dondeti et al. proposed the Dual Encryption Protocol (DEP) in [21, 22, 23]. DEP considers the case where intermediaries may be not trusted, and thereby proposes to use a double encryption scheme in TEK distribution in order to prevent those intermediaries from having access to propagated TEKs.

2.2.2 TEK Per Subgroup Protocols

The *common TEK* protocols has the drawback to require that all group members commit to a new TEK, whenever a membership change occurs in the group, in order to ensure *perfect backward and forward secrecy.* This is commonly called *1-affects-n* phenomenon. In order to mitigate the *1-affects-n* phenomenon, another decentralized approach consists in organizing group members into subgroups. Each subgroup uses its own independent TEK. Indeed, in this scheme when a membership change occurs in a subgroup, it affects only the members of the same subgroup. Mittra proposed in [37] the *Iolus* architecture which is a framework of a hierarchy of multicast subgroups. Each subgroup is managed by a Group Security Agent (GSA) which is responsible for key management inside the subgroup. A main controller called the Group Security Controller (GSC) manages the GSAs. Figure 1 illustrates a hierarchy with six subgroups. Each of them uses its own TEK.

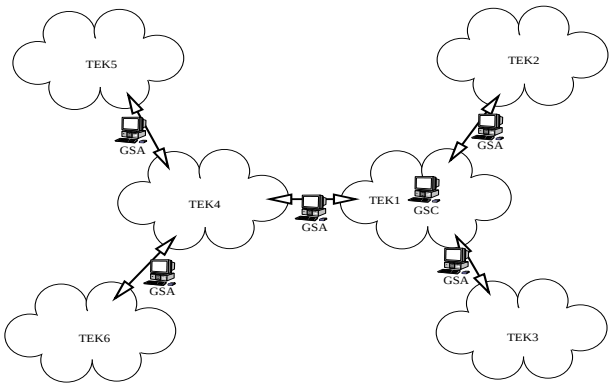

Figure 1: An example of a Iolus architecture

When a membership change occurs in a subgroup, only that subgroup is involved in a re-key process. This way, Iolus scales to large groups and mitigates *1-affects-n* phenomenon. However, Iolus has the drawback of affecting the *data path.* Indeed, there is a need for translating the data that goes from one subgroup, and thereby one key, to another. Shields et al. proposed another protocol that uses the same concept, called the *Keyed HIerarchical multicast Protocol(KHIP)* [52]. *KHIP* operates at the routing level where core routers ensure the translation of the

packets. Instead of translating data itself, the protocol translates only the headers of the packets that contain a random key with which data is encrypted. In the case of the framework proposed by Molva and Pannetrat in [38], each time multicast messages pass through special nodes on the multicast tree, they are transformed using special functions called *Cipher Sequences*. Another decentralized solution with TEK per subgroup is the architecture proposed by Yang et al. in [59] where multicast group is organized into a set of subgroups, and each subgroup is managed by a Key Server wich redistribute periodically a new TEK to its subgroup members. Challal et al. proposed in [13] SAKM protocol. The idea of SAKM is to organize dynamically over the time the multicast group into clusters of subgroups that use the same TEK. Recently, Mukherjee and Atwood proposed the SIM-KM protocol in [40] which uses *proxy encryption* [39] to transform data at the border of a subgroup. Proxy functions convert cipher text for one key into cipher text for another key without revealing secret decryption keys or clear text messages. This allows SIM-KM to do subgrouping with data transformation in order to limit the impact of rekeying, even thought intermediaries are not trusted entities. Huang and Mishra [31] proposed the Mykil protocol with fault tolerance and mobility support [32]. The Mykil protocol combines the TEK per subgroup and common TEK approaches.

2.3 Distributed Key Agreement

In this approach, the group members cooperate to establish a group key. This improves the reliability of the overall system and reduces the bottlenecks in the network in comparison to the centralized approach. Ingemarson et al. [33] and Steiner et al. [54] proposed to extend Diffie-Hellman key agreement protocol [17] to group communication. In their schemes, group members perform intermediate DH exchanges, through the ring, and finally culminate into the *common group key*. Perrig et al. proposed in [35, 41] DH-LKH a distributed Diffie-Hellman implementation of LKH (cf. Section 2.1) through hierarchical collaboration. Similarly, Rodeh et al. [49], Dondeti et al. [20] and Waldvogel et al. [55] propose distributed versions of LKH (D-LKH), OFT (D-OFT) and CFKM (D-CFKM) protocols respectively following a hierarchical cooperation of group members. Seba et al. [50] proposed the *Fault-Tolerant Group Diffie-Hellman* (FTGDH) protocol where the key agreement is established only with members which are supposed correct. For that, failure detectors of Chandra and Toueg [15] are used. Indeed, the member M_i does not send its contribution to the successor M_{i+1} but to correct (non faulty) successor. In FTGDH, each member can start the key agreement.

2.4 Discussion and Motivation

We notice that proposed solutions in the literature suffer from great concerns depending on group dynamism: protocols with *common TEK* suffers from the *1-affects-n* phenomenon, where a single group membership change (join or leave) results in a re-keying process that disturbs all group members to update the TEK. Moreover, centralized protocols are not scalable, and distributed ones bring new challenges such as synchronization, conflict resolution and required time to construct the key. On the other hand, decentralized protocols with *TEK per subgroup approach* reduces the *1-affects-n* problem. This is advantageous for highly dynamic multicast groups. However, this approach requires translation of sent messages whenever they pass from a subgroup to another, and this may not be supported in applications which are sensitive in transmission time and do not tolerate **delays** in packet delivery. Besides, this approach would not be worthy with relatively static groups because the multiple translations would induce avoidable delays and useless computation overheads. These shortcomings are due to the lack of *dynamism awareness* in existing group key management schemes.

We propose a decentralized architecture for group key management which takes into consideration the dynamic aspect of group membership. We call it: ***S**calable **D**elay-constrained Multicast Group **K**ey **M**anagement (SDKM).*

3 Overview and Principles of Our Solution

In our solution, the multicast group members are organized into multiple subgroups (as in Iolus [37]), where each subgroup is managed by an agent called ***S**ubgroup **M**anager* (SM). The set of agents form a tree structure rooted at a subgroup whose the agent is the source of traffic. The subgroup i is managed by SM_i. The *Subgroup Manager* is responsible for the local key management process and packet delivery in the subgroup and can be in two possible states: *active* or *passive*. An active SM uses an independent TEK for the subgroup under its control and thus it has to decrypt and re-encrypt received messages before forwarding them to local members. A passive SM uses the same TEK as its parent subgroup, and hence forwards received messages to local members without decryption/re-encryption. So, the whole SMs' states induces a partition of the subgroups into a set of clusters. Each cluster is composed of a set of subgroups that share the same TEK. The root cluster SM is active and all internal/leaf SMs are passive in a cluster. The root SM of the SDKM tree is always active but does not do decryption/re-encryption (see Figure 2).

Each SM is member in two subgroups: its own subgroup and its parent subgroup. Thus, each SM_i knows two TEKs, its TEK and the TEK of its parent subgroup. If a subgroup i is in the same cluster with its parent subgroup j then $TEK_i = TEK_j$, otherwise they are in two different clusters then $TEK_i \neq TEK_j$. Initially, all subgroups use the same TEK. In Figure 2 for example, subgroups belonging to the Cluster 3 use the

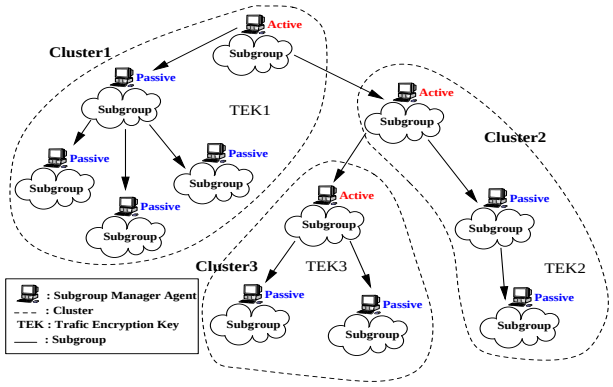

Figure 2: The SDKM architecture

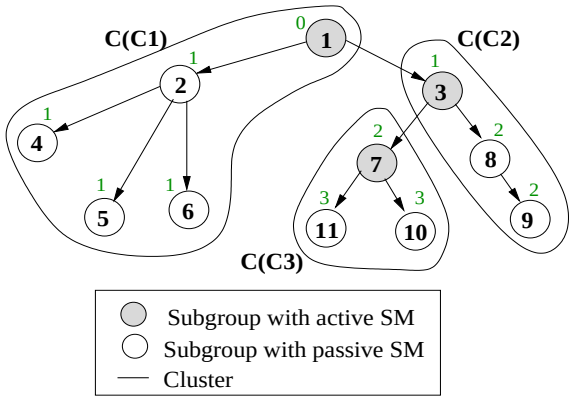

Figure 3: Formalization of a *SDKM* architecture

same traffic encryption key TEK_3 which is different from $TEK2$ the traffic encryption key of Cluster 2.

Our objective is to find a partition of multicast group (a hierarchy of subgroups) into hierarchy of clusters $\{C_1, C_2 \ldots, C_i, \ldots\}$ where each cluster C_i contains a set of subgroups in hierarchical mode (see Figure 2). To do a best partitioning, we consider in our solution delays in packet delivery and the *1-affects-n* phenomenon. We aim to gather (merge) in the same cluster subgroups which are stable, i.e., membership changes rarely occur. So, SMs exchange periodically information about their subgroups. According to this information they decide to be either active or passive, what induces a partitioning to the SDKM hierarchy. These information will be detailed in subsequent sections. In following section, we formalize the partitioning problem in graph theory.

3.1 Formalization

We model the *SDKM* hierarchy by a tree structure $T = (G, V)$, where G is the set of *SDKM* SMs (Subgroup Mangers) and V the set of edges, where each edge connects two subgroups. The problem of affecting a state (*active* or *passive*) to the SMs is equivalent to partitioning the whole hierarchy of subgroups into clusters , where the root of each cluster is an *active* SM and internal SMs are *passive*. We note by $T_i = (G_i, V_i)$ the subtree of T where T_i is a tree such that $G_i \subset G$, $G_i \neq \Phi$ and $V_i \subset V$. Let $E = \{T_1, T_2, \ldots T_i, \ldots\}$ be the family of all possible subtrees of T. These subtrees constitute all the *SDKM* possible clusters $\{C_1, C_2 \ldots, C_i, \ldots\}$ of the given *SDKM* hierarchy. To each cluster C_i, we associate a cost $C(C_i)$ that evaluates the different overheads induced by subgroups of this cluster. We formalize this cost function in a subsequent section. Figure 3 illustrates the tree structure associated to the *SDKM* hierarchy of Figure 2.

Finding a sub-family of E: $F = \{T_{i_1}, T_{i_2}, \ldots, T_{i_i}, \ldots\}$, such the total cost of the corresponding *SDKM* clusters $\{C_{i_1}, C_{i_2}, \ldots, C_{i_i}, \ldots\}$ is minimal could be formalized as follows:

$$\begin{cases} \min \sum_{C_k/T_k \in F} C(C_k) \\ \text{with } F \subset E \\ \text{and } \bigcup T_k = T \\ \text{and } \forall T_j, T_k \in F, T_j \cap T_k = \Phi \text{ if } j \neq k. \end{cases} \quad (1)$$

The objective of this *optimization* problem is to *minimize* the overall overhead induced by partitioning the *tree* into *clusters of sub-trees*. Each cluster induces a new delay in packet delivery because of data translation at its root. In addition, it induces a rekeying overhead that depends on the dynamism of the clustered subgroups. In what follows, we present our *key translation* and re-keying strategies, and show how they reduce the overall overheads. Then, we evaluate the dynamism homogeneity inside a cluster, and the delay in packet delivery induced by clustering. Then, we formalize the *cluster cost function*: $C(C_i)$. Once the optimization problem is completely defined, we propose a heuristic, and hence a protocol that solves it.

3.2 SDKM Key Translation Technique

In SDKM, we adopted the key translation proposed by Shields et al. in [52]. This technique can be summarized as follows: When the sender wants to send a message M to group members: It generates randomly a secret symmetric key K_s; It encrypts the message M with K_s to obtain $\{M\}_{K_s}$; and finally It sends by multicast to its members the message $(\{K_s\}_{TEK_r}, \{M\}_{K_s})$, where TEK_r is the TEK of the SDKM root subgroup. When an agent SM_j receives the message from its parent agent SM_i, it just forwards the message to its members if it is passive. If it is in active state, it decrypts the first part of the received message $(\{K_s\}_{TEK_i})$ using parent's subgroup TEK (TEK_i) to get K_s. Then, it encrypts K_s with its own TEK (TEK_j) and sends by multicasting to its members the message $(\{K_s\}_{TEK_j}, \{M\}_{K_s})$. When a member of subgroup i receives the message from SM_i, it decrypts the first part $(\{K_s\}_{TEK_i})$ using TEK_i to obtain K_s. Then it decrypts the second part $(\{M\}_{K_s})$ with

K_s to obtain the data M. With the presented key translation technique we reduce the decryption/re-encryption overhead from many bytes (the size of the message M) to just few bits (the size of the key K_s).

3.3 SDKM Re-keying Technique

As described above, the common TEK protocols suffer from the *1 affects n* phenomenon. Indeed, after each membership change (*join* or *leave*), a new TEK is generated for the whole cluster and must be distributed for all cluster members. In our approach we use a re-key strategy that allows decreasing the overhead caused by membership changes. We propose an approach that minimizes the number of re-keying messages in a cluster. In SDKM, each subgroup has a Subgroup Key Encryption Key $SKEK$ which is different from $SKEK$s of other subgroups. Each SM_i shares secretly its $SKEK_i$ with only valid members of subgroup i. The $SKEK$ is used to encrypt the new TEK before sending it to subgroup members. In what follows, we detail our re-keying strategy in case of *join/leave* operations.

3.3.1 Join Re-keying

When a new member joins the subgroup i, a new TEK must be generated for the cluster C_k containing subgroup i in order to ensure the backward secrecy. Let us note by m_i^j the member j of subgroup i. SM_i sends securely by unicast to m_i^j a symmetric key K_i^j which will be shared only between the subgroup manager (SM_i) and the member m_i^j. The root SM of cluster C_k generates a new TEK_k that will be distributed to all cluster members. The root SM of cluster (SM of root subgroup) sends the new TEK encrypted with the old one ($\{\text{new } TEK_k\}_{\text{old } TEK_k}$). When an internal SM of cluster C_k receives the new TEK it just forwards to its members the message ($\{\text{new } TEK_k\}_{\text{old } TEK_k}$). The SM_i responsible for subgroup i, where the *join* occurred, forwards also to its members the received message ($\{\text{new } TEK_k\}_{\text{old } TEK_k}$), but it sends the new TEK to the new member encrypted with the secret key they share ($\{\text{new } TEK_k\}_{K_i^j}$). Thus, when a join occurs in subgroup i, SM_i sends two messages of re-keying and the other subgroups send one message of re-keying.

3.3.2 Leave Re-keying

When a member m_i^j leaves a subgroup i, a new TEK is generated by the cluster's root. All SMs distribute to their members the new TEK in order to ensure the forward secrecy. SM_i agent of subgroup i where the leave occurred generates a new $SKEK$ for subgroup i and sends it with the new TEK by unicast to each member m_i^l (with $l \neq j$) by unicast encrypted with K_i^l ($\{\text{new } TEK_k, \text{ new } SKEK_i\}_{K_i^l}$). The other SMs in the same cluster C_k send the new TEK by multicast to their members encrypted with their $SKEK$s ($\{\text{new } TEK_k\}_{SKEK_t}$). With the use of $SKEK$s in our approach, we reduce the overhead due to re-keying a cluster. Indeed, SMs of other subgroups have to send just one message of re-keying when a membership change occurs in another subgroup of the same cluster.

3.4 SDKM Problem Statement

The partitioning of group members into subgroups aims to reduce the *1-affects-n* phenomenon. The principle of our approach is based on merging in the same cluster subgroups having homogeneous membership dynamism. Initially all subgroups constitute a single cluster and use the same TEK (Figure 4). Internal SMs take the decision to become either active or passive. An active SM decrypts/re-encrypts messages (see Paragraph 3.2) before sending them to its members. The impact of decryption/re-encryption operations will be discussed in Section (4.1).

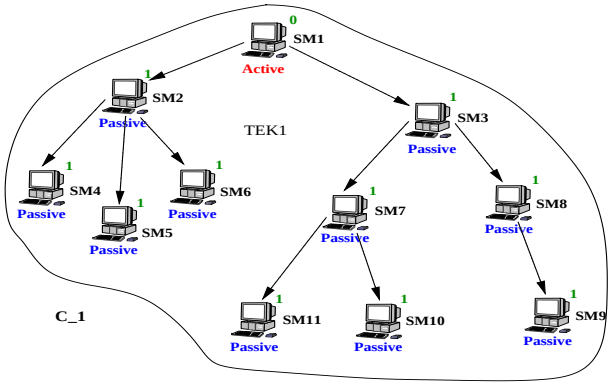

Figure 4: Initial state of the SDKM hierarchy

Our aim is to find a partition of SDKM hierarchy into clusters of subgroups with similar membership change frequencies which minimizes delays in transmission. When a SM_i becomes active, a new cluster will be created with SM_i as a root. All passive SMs which are in the subtree rooted at SM_i will belong to the newly created cluster. The impact of a change in a SM_i status can be summarized as follows:

- If SM_i becomes active, then a new cluster rooted at SM_i will be created.
- If SM_i becomes passive, then the SM_i's cluster merges with the SM_i's parent cluster.

Let us explain the behavior of SMs by an illustrative example of SDKM hierarchy where each subgroup is represented by its SM. In Figure 4, suppose that Subgroup 3 (represented by SM_3) is stable and its parent Subgroup 1 is very dynamic. So, it is better that SM_3 becomes active to protect its subgroup from re-keyings caused by frequent membership changes at Subgroup 1. If SM_3 becomes active, it would create a new cluster with an independent

TEK and SM_3 as root. Then, Subgroup 1 would no longer disturb Subgroup 3.

4 Model and Definition Requirements

In Table 2 we summarize the nomenclature used through the paper.

Table 2: Nomenclature

Symbol	Signification
C_i	A cluster in the SDKM hierarchy
SM_i	Subgroup Manager Agent of subgroup i
$C(C_i)$	The total cost associated to a cluster C_i
$DR(i)$	Delay overhead at subgroup i
$\tau(i)$	Key translation overhead of SM_i
$DD(i,j)$	*dynamism distance* between subgroups i and j
α	The weight given to *key translation* overhead
β	The weight given to dynamism overhead
λ_i	Members' arrival frequency at subgroup i
θ	Period of time after which *subgroup manager agents* re-execute SDKM protocol
$d(i)$	Delay Distance of the SM of subgroup i
$p(i)$	Parent subgroup of subgroup i
$r(C_k)$	The root of cluster C_k

In order to approximate the impact of re-keying due to the clustering, we need a multicast dynamism model since this re-keying is due to changes in the subgroups membership. Almeroth et al. showed in [1, 2] that the dynamism of some multicast sessions over the MBone can be modeled as follows: the users arrive in a multicast group according to a Poisson process with rate λ (arrivals/time unit), and the membership duration of a member in the group follows an exponential distribution with a mean duration $\frac{1}{\mu}$ time units. In our case, we apply this model to each subgroup. Unlike [14], we do not suppose that the subgroups are likely to be joined by the members. Instead, each subgroup is characterized by its own parameters λ and μ. Moreover, we suppose that the parameters λ and μ change over time and thus each SM adjusts its estimations of λ and μ of its subgroup every θ time units in order to approximate better the re-keying overhead.

4.1 Impact of Decryption/Re-encryption Overhead

When a SM becomes active it will do decryption/re-encryption of the secret key K_s which we described in Section 3.2. The decryption/re-encryption is an operation which is costly in time despite it concerns only few bits (the secret key K_s). Especially in applications which don't tolerate delay in delivery such as real time applications. Now, let us consider this definition:

Definition 1. *The Delay Distance (d(i)) of a subgroup agent SM_i is given by the number of times that packets are decrypted/re-encrypted before it receives them. It is equal to the number of active SMs in the branch connecting SM_i to the SDKM tree root SM.*

The delay distance of SM_i reflects the delay due to decryption/re-encryption at active SM_i. For a passive SM we can estimate the delay distance it will have if it takes the decision to become active. So, we associate to a passive SM the *delay distance* it will have if it becomes active. Thus, when a SM changes its state it has the same value of *delay distance*. The *delay distance* of a SM is changed only if any of SMs connecting it to SDKM root SM changes its state (ancestor SMs).

4.2 Impact of Dynamism Overhead

In our approach we aim to put in the same cluster subgroups which are homogeneous in membership dynamism in order to reduce *1-affects-n* phenomenon. This means subgroups which have similar membership change frequencies. Now, let us consider some definitions in order to simplify subsequent concepts.

Definition 2. *The Dynamism Distance between two subgroups i and j (DD(i,j)) is the difference of dynamism frequencies between them. DD measures if one of the subgroups is more dynamic than the other.*

Two subgroups which have similar dynamism frequencies will have reduced DD. The *dynamism distance* between two subgroups i and j will be expressed only by their members' arrival frequencies λ_i and λ_j. In Section 6.2 we give the formulation of the *dynamism distance*.

5 Cluster's Cost Function

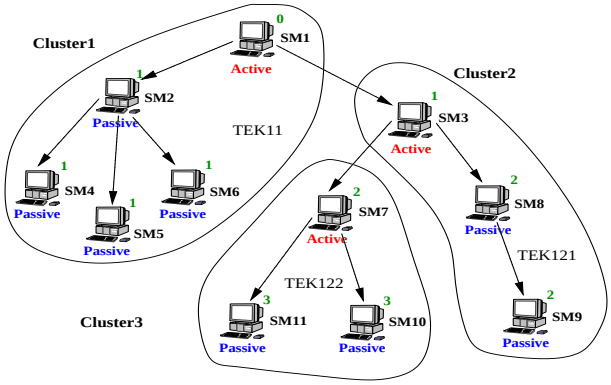

Figure 5: Example of SDKM partition

Active SMs decrypt/re-encrypt the secrete key K_s each time they receive packets from upper SM (see Paragraph 3.2). Decryption/re-encryption operations are costly in time. We associate a cost due to decryption/re-encryption operations for an active SM_i. This cost can be expressed by the frequency that SM_i decrypts/re-encrypts the key K_s and the processing power of the agent. This frequency is the arrival rate of multicast flow packets. Let us denote this by $\tau(i)$. In Section 7, we give a detailed formulation of $\tau(i)$. Then, all SMs will have the

same value of $\tau(i)$. But if there is a lot of active SMs in a branch, delays would accumulate as we go down in the hierarchy. For example, in Figure 5, the delay created at SM_7 is greater than the delay at SM_3. In fact, the agent SM_3 is the parent of SM_7 in the hierarchy, and both are active. Thus, the two agents decrypt/re-encrypt received messages before forwarding them to their members. Since SM_7 receives packets from SM_3, then the delay created by SM_3 has an impact on SM_7's delay. So, the delay at SM_7 is bigger then SM_3's one as shown in Figure 6.

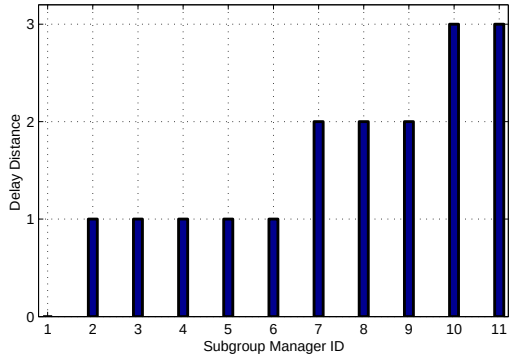

Figure 6: Illustration of the delay impact

Definition 3. *The Delay Cost due to decryption/re-encryption operations for a subgroup i with an active SM_i is given by*

$$DR(i) = d(i) \times \tau(i)$$

with $d(i)$ Delay Distance of SM_i.

We know now that when a SM is passive it has a *dynamism distance* with its parent SM, and when a SM is active it creates a delay in transmission. For a cluster C_k we associate a cost function ($C(C_k)$) that expresses the overheads due to the dynamism in the cluster and delays in transmission. Let us consider for a subgroup i:

$$st(i) = \begin{cases} 1, \text{ if } SM_i \text{ is active} \\ 0, \text{ if } SM_i \text{ is passive.} \end{cases}$$

Let us note by $p(i)$ the parent subgroup of the subgroup i in the SDKM hierarchy. We defined two costs: the *Dynamism Distance* (Definition 2) and the *Delay Cost* (Definition 3) in order to evaluate the *dynamism homogeneity* and the *key translation* overheads, respectively. We know that only active SMs (root nodes of clusters) do decryption/re-encryption operation and they are not disturbed by re-keying messages of parent subgroups because they belong to different clusters. Let α and β be the weight factors given to the *Dynamism Distance* and the *Delay Cost*, respectively. The importance and the role of these parameters will be discussed later in this paper. For a cluster C_k we associate a cost function $C(C_k)$ that expresses the *dynamism homogeneity* and the *delay* due to *key translation*. In general case this function is given, for a cluster C_k, by:

$$\begin{aligned} C(C_k) &= \sum_{i \in C_k} (st(i) \times \alpha \times DR(i) \\ &\quad + (1 - st(i)) \times \beta \times DD(i, p(i))). \end{aligned} \quad (2)$$

Only the SM of the root subgroup does decryption/re-encryption in a cluster and has not *dynamism distance* with its parent subgroup because they are in two different clusters. Let us denote by $r(C_k)$ the root subgroup of cluster C_k. Equation 2 can be rewritten as follows:

$$C(C_k) = \alpha \times DR(r(C_k)) + \beta \times \sum_{i \in C_k}^{i \neq r(C_k)} DD(i, p(i)). \quad (3)$$

The optimization Problem (1) becomes:

$$\begin{cases} \min \left(\sum_{C_k / T_k \in F} \alpha \times DR(r(C_k)) \right. \\ \left. + \beta \times \sum_{i \in C_k}^{i \neq r(C_k)} DD(i, p(i)) \right) \\ \text{with } F \subset E \\ \text{and } \bigcup T_k = T \\ \text{and } \forall T_j, T_k \in F, T_j \cap T_k = \Phi \text{ if } j \neq k. \end{cases} \quad (4)$$

6 Protocol

Now, we will associate to each member the values that it compares in order to take the decision to become either active or passive.

Recall that *dynamism* distribution over a multicast session is not uniform neither over space nor over time [1, 2]. We assume that periodically (after each θ time units) each SM_i of the SDKM hierarchy sends dynamism information of its subgroup (λ_i and $d(i)$) to its child subgroups.

When a SM_i receives dynamism parameters of its parent SM_j, it updates its estimations and evaluates if it has similar membership dynamism with its parent and the overhead induced by delivery delay given respectively by the two formulas:

$$\beta \times DD(i, p(i))$$

and

$$\alpha \times DR(i).$$

Then SM_i takes the decision to become *active* or *passive* depending on the comparison between these two overheads:

if $\beta \times DD(i, p(i)) \leq \alpha \times DR(i)$
then SM_i becomes passive;
if $\beta \times DD(i, p(i)) > \alpha \times DR(i)$
then SM_i becomes active;

Parameters α and β can be seen as the importance given respectively to split, merge subgroups. We can fix them according to the application type. They play a key role in the operation and the performance of the proposed scheme.

Proposition 1. ***Let*** m ***be the number of subgroups (SMs) in SDKM tree. The objective function of the optimization Problem (4) can be rewritten as follow:***

$$\boldsymbol{Cost} = \sum_{i=1}^{m} C_{sub}(i) \tag{5}$$

with,

$$C_{sub}(i) = \begin{cases} \alpha \times DR(i), \textbf{ if } SM_i \textbf{ is active} \\ \beta \times DD(i,p(i)), \textbf{ if } SM_i \textbf{ is passive}. \end{cases}$$

Proof. Let m be the number of subgroups in the SDKM hierarchy. Let n be the number of clusters in a given partition of the SDKM hierarchy. According to Equation 3, the total cost of SDKM hierarchy is given by:

$$\begin{aligned} \boldsymbol{Cost} &= \sum_{k=1}^{n} (\alpha \times DR(r(C_k)) \\ &\quad +\beta \times \sum_{i \in C_k}^{i \neq r(C_k)} DD(i,p(i))) \\ &= \sum_{k=1}^{n} (\sum_{i \in C_k} (st(i)\alpha \times DR(i) \\ &\quad +(1-st(i))\beta \times DD(i,p(i)))) \\ &= \sum_{i=1}^{m} (st(i)\alpha \times DR(i) \\ &\quad +(1-st(i))\beta \times DD(i,p(i))) \\ &= \sum_{i=1}^{m} C_{sub}(i). \end{aligned} \tag{6}$$

From Equation 6, we can easily see that the cost of the overall hierarchy depends on SM agents' states. In fact, the cost function of a cluster C_k can be formulated as the sum of subgroups' costs. □

This reformulation of cost function is used to prove the lemmas which follow.

6.1 Split/Merge Subgroups

In this section, we discuss and evaluate the performance consequences of the split/merge decisions taken according to the proposed heuristic.

6.1.1 Merge Subgroups

Lemma 1. *For an agent* SM_i*, if* $(\beta \times DD(i,p(i))) \leq (\alpha \times DR(i))$*, then* SM_i *becomes passive and the decision will be optimal. The resulted total cost of the partition after merging less than the total cost of the partition before merging.*

In other words, if the cost that a SM has when it is active is greater than the cost it would have when it becomes passive, then it is more interesting that this SM becomes passive.

Proof. Suppose that we have a partition of n clusters $\{C_1, C_2, ..., C_k, C_{k+1}, ..., Cn\}$. Suppose that cluster C_k contains t *Subgroup Managers* with SM_i being the root of C_k. Suppose that $(\beta \times DD(i,p(i))) \leq (\alpha \times DR(i))$. When SM_i becomes passive, the cluster (C_k) will be merged with its parent cluster. In the cost of the partition after merging, only $C_{sub}(i)$ (see proposition 1) will be changed from $(\alpha \times DR(i))$ to $(\beta \times DD(i,p(i)))$. Costs of active SMs which are in the descendants of SM_i will decrease because their *delay distances* decrease. According to Equation 5, the new partition has a cost lower than the old one. □

6.1.2 Split Subgroups

When a SM is passive, it does not induce a decryption/re-encryption overhead but when it calculates its DR, it obtains the estimation of its DR in the case that it would be active.

For a SM_i, if $(\alpha \times DR(i)) < (\beta \times DD(i,p(i))))$, then the cost of *key translation* operations is less than the cost when SM_i uses the same TEK as its parent subgroup. It is more interesting that SM_i becomes active because the delay that it would incur to flow transmission would be lower than the dynamism overhead it induces when it is merged with its parent subgroup.

Lemma 2. *Let* $SM_i \in C_k$ *be an agent with* SM_j *the root of cluster* C_k*. If* $(\alpha \times DR(i)) < (\beta \times DD(i,p(i))))$*, then* SM_i *can become active. Cluster* C_k *will be split to two clusters,* C_{k1} *with* SM_j *as root and* C_{k2} *with* SM_i *as root. We will have* $C(C_{k1}) + C(C_{k2}) < C(C_k)$*.*

Proof. Before the split operation the cost of SM_i in C_k was $(\beta \times DD(i,p(i))))$. Let us write the cost of C_k as $(B + (\beta \times DD(i,p(i)))))$, with B the costs' sum of the other SMs in C_k. According to Equation 5, when SM_i becomes active $C(C_{k1}) + C(C_{k2}) = B + (\alpha \times DR(i)))$. We have an unchanged value of B because only SM_i changed its state to active, and the other SMs stay passive with the original root of C_k which stay active also. Here, the decision is optimal if we compare only with the original cluster and the created clusters because the *delay distance* of SM_i stays the same, and we have $(\alpha \times DR(i)) < (\beta \times DD(i,p(i))))$. Let us remind that the delay distance of SM_i changes only if an ascendant SM (ancestor of SM_i in the SDKM tree) changes state. □

Note that the decision that a SM becomes active may be not optimal for the whole SDKM partition, because when SM_i becomes active it increases the *delay distance* of SMs which are below in the hierarchy. So, for active SMs the value of DR will increase and the global cost (optimization Problem 4) of the partition may increase. Even though our approach does not guarantee optimality, it has the advantage of the flexibility guaranteed by proposed heuristic. When the value of DR increases for an SM because another SM above becomes active means that there is now an important delay in flow transmission, and in the next decision, the SM may become passive if

it notes that its DR is greater than its DD. It is more interesting that SM supports the overhead due re-keying messages than creating delay in multicast flow transmission if we know that one of the characteristics of multicast communications is the routing delays.

6.1.3 Steady State of SDKM Tree

When subgroups reach a stable state with stable membership frequencies, they will have unchanged values of their DD. In this state, SMs takes the decision to stay in current state. At each comparison for an SM_i it will have the same value in comparison between $\alpha \times DR(i)$ and $\beta \times DD(i, p(i)))$ and thus maintains its state. Therefore, the SDKM tree will be partitioned into a stable configuration.

6.2 Implementation

We have defined the *dynamism distance* between two agents by the difference of dynamism frequencies between subgroups. In this section we give two possible formulations to express the *dynamism distance*.

6.2.1 SDKM.1

In a first time, we formulate the *dynamism distance* between two subgroups by the difference of membership change frequencies. According to our model presented in Section 4, the members arrive in a subgroup i according to a Poisson process with rate λ_i (arrivals/time unit). This means that the members leave a subgroup according to the same law and the same parameter λ_i [10]. So, we obtain the membership change frequency in a subgroup i equal to $2 \times \lambda_i$. We define the *dynamism distance* between two subgroups i and j by

$$\begin{aligned} DD(i,j) &= (2 \times \lambda_i - 2 \times \lambda_j)^2 \\ &= 4 \times (\lambda_i - \lambda_j)^2. \end{aligned}$$

Then SM_i takes the decision to become *active* or *passive* depending on the comparison between the two overheads *dynamism distance* ($\beta \times 4 \times (\lambda_i - \lambda_{p(i)})^2$) and *key translation* ($\alpha \times d(i) \times \tau(i)$). Since α and β are weight factors, the agent SM_i compares between ($\beta \times (\lambda_i - \lambda_{p(i)})^2$) and ($\alpha \times d(i) \times \tau(i)$). Then it takes its decisions as follows:

if $\beta \times (\lambda_i - \lambda_{p(i)})^2 \leq \alpha \times d(i) \times \tau(i)$
then SM_i becomes passive;
if $\beta \times (\lambda_i - \lambda_{p(i)})^2 > \alpha \times d(i) \times \tau(i)$
then SM_i becomes active;

The protocol SDKM.1 ensures that each cluster contains subgroups which are homogeneous in membership change frequencies.

6.2.2 SDKM.2

The weakness of SDKM.1 is that it merges in a same cluster very dynamic subgroups because they have similar membership change frequencies. So, in a second stage, we are interested in keeping the advantages of SDKM.1 which are merging in the same cluster subgroups which have similar membership change frequencies, but with taking into account the dynamism of the subgroups. Thus, if a subgroup is very dynamic, it's more interesting that we isolate it because it disturbs other subgroups. For this reason we propose to multiply the distance between the two frequencies by the membership change frequency of subgroup i ($2 \times \lambda_i$). In this case, *dynamism distance* will be given by

$$\begin{aligned} DD(i,j) &= 2 \times \lambda_i \times (2 \times \lambda_i - 2 \times \lambda_j)^2 \\ &= 2 \times \lambda_i \times 4 \times (\lambda_i - \lambda_j)^2 \\ &= 8 \times \lambda_i \times (\lambda_i - \lambda_j)^2. \end{aligned}$$

Then, in SDKM.2 the agent SM_i takes the decision to become *active* or *passive* depending on the comparison between the two overheads:

if $\beta \times \lambda_i (\lambda_i - \lambda_{p(i)})^2 \leq \alpha \times d(i) \times \tau(i)$
then SM_i becomes passive;
if $\beta \times \lambda_i (\lambda_i - \lambda_{p(i)})^2 > \alpha \times d(i) \times \tau(i)$
then SM_i becomes active;

7 Simulation

In this section, we present our simulation model and some results of the carried out simulations in which we compared the two alternatives of SDKM (SDKM.1 and SDKM.2) with two protocols: Iolus [37] which is a decentralized approach with *independent TEK per subgroup*, and the *Group Key Management Protocol (GKMP)* proposed by Harney and Muckenhirn in [29, 30] which is a centralized solution with *common TEK* for the whole group. We study the *1 affects n* behavior of each simulated protocol and the number of decryption / re-encryption operations required for the communication. Let's remind that in SDKM only the key K_s (see Paragraph 3.2), used to encrypt the multicast data, is decrypted / re-encrypted by *active* SMs.

7.1 Simulation Model

In our simulations, we use a SDKM hierarchy composed of 5 subgroups as illustrated in Figure 7. Unless specified otherwise, we considered a session of 3 hours, where members arrive following a Poisson law with an average inter-arrival equal to 20 seconds, and remain in the session 30 minutes in average. After each $\theta = 15min$, the subgroup managers reconsider the dynamism information and decide to become *active* or *passive*. The affectation of arriving members to the different subgroups is random

with percentages varying over time. In table 3, we illustrate the percentage distribution of arrivals to the different subgroups during the whole session.

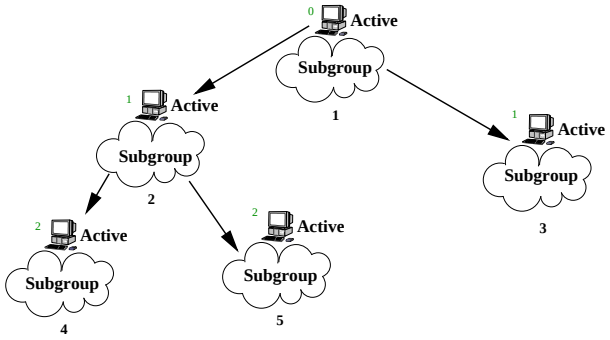

Figure 7: Simulated SDKM example

Table 3: Percentage distribution of arrivals to the different subgroups

Period	1	2	3	4	5
0h-1h	20%	20%	40%	15%	5%
1h-2h	40%	20%	20%	15%	5%
2h-3h	15%	20%	5%	40%	20%

In definition 3 we expressed the cost of *key translation* with of a SM_i by $DR(i) = d(i) \times \tau(i)$. For simplicity reasons, we assume that the subgroups' manager agents (SMs) have the same computation power. They use a symmetric encryption system such as AES [27] or 3DES [4]. We denote by Alg, the computation time per data size unit to do encryption using the agents' computation power. We assume also that packets arrive with a static rate r. Therefore, the *key translation* overhead can be written as

$$\tau(i) = 2 \times r \times Alg.$$

So, we will have $DR(i) = d(i) \times 2 \times r \times Alg$.

7.2 Simulation Results

Recall that every θ time units the agents update their status according to SDKM protocol depending on the actual dynamism. Recall also that the two compared quantities: *Delay overhead* due to *Key translation* and *dynamism distance* are weighted with the two factors α and β.

7.2.1 Impact of the Weight Factors (α, β)

In a first stage, we were interested in the impact of these weight factors on the performance of our protocol, and how they allow to adapt the behavior of our protocol to the requirements of the application level in terms of synchronization and tolerance to latencies in packet delivery. For this purpose, we varied the value of $\frac{\alpha}{\beta}$ and we carried out intensive simulations for each value. Figure 8 shows the impact of $\frac{\alpha}{\beta}$ on the *number of affected members by membership changes*: *1-affects-n*. We notice that when $\frac{\alpha}{\beta}$ increases, the *1-affects-n* phenomenon increases also. Initially SDKM has similar behavior to Iolus because all SMs were active and it increases over the time.

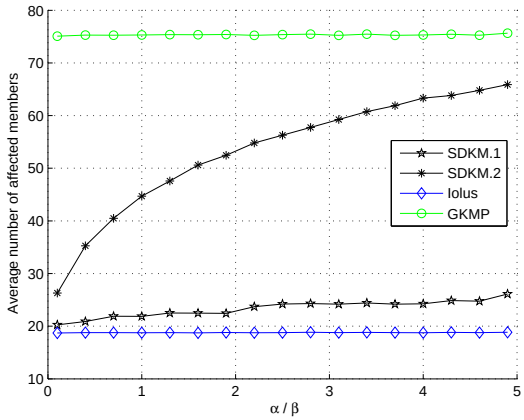

Figure 8: Impact of $\frac{\alpha}{\beta}$ variation on *1-affects-n* overhead. Average inter-arrival=20s, average membership duration=30min, $\theta = 15min$

This is due to the fact that when we increase $\frac{\alpha}{\beta}$, this means that we increase α and/or decrease β. Thus we give more importance to merging subgroups with their parent subgroups. Consequently, the number of affected members increases. In SDKM.2 the number of affected members increases faster than in SDKM.1. Obviously, as we can see in Figure 9, the number of decryption / re-encryption operations decreases, because of the tendency of subgroup managers to merge instead of splitting. We can easily note that SMs becomes passive faster in SDKM.2.

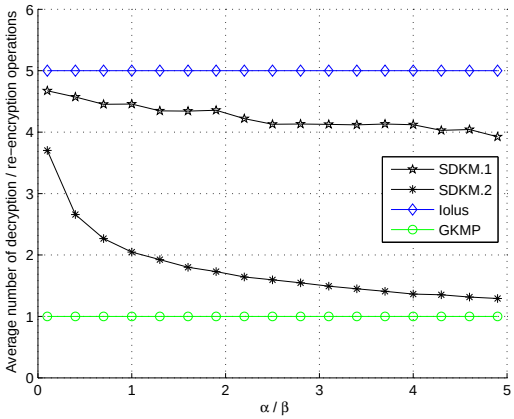

Figure 9: Impact of $\frac{\alpha}{\beta}$ variation on decryption / re-encryption overhead. Average inter-arrival=20s, average membership duration=30min, $\theta = 15min$

In conclusion, the two weight factors give a large spectrum of flexibility for our approach to meet the requirements of the application level in term of tolerance to latencies in packet delivery. So, smallest values of $\frac{\alpha}{\beta}$ suit better for applications that are not sensitive to delays in packet delivery (such as off-line software updates). Greater values of $\frac{\alpha}{\beta}$ (greater than 3 for SDKM.2) suit better for applications that are critical in packet delivery as real-time applications.

7.2.2 Impact of the Group Size

In a second stage, we were interested in the size scalability of SDKM compared to the other approaches in what relates to *1-affects-n* and decryption /re-encryption overheads. Two parameters of our simulation model control the size of the group: the *average inter-arrival* of members into the session, and the *average membership duration* of the members in the session. In Figures 10 and 11 we varied the average inter-arrival from 1 second to 65 seconds and we measured the *1-affects-n* and the decryption / re-encryption overheads, respectively.

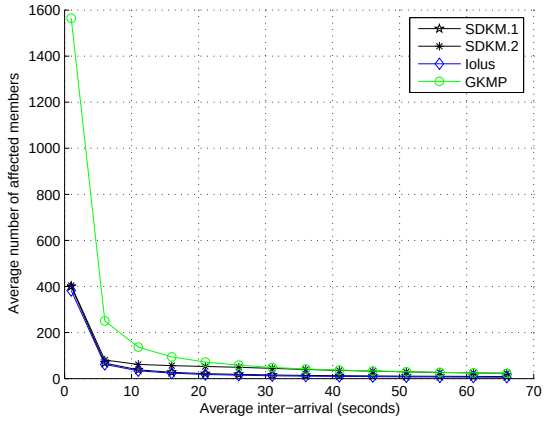

Figure 10: Impact of inter-arrival variation on *1-affects-n* overhead. $\frac{\alpha}{\beta} = 2$, average membership duration=30min, $\theta = 15min$

The smallest values of the inter-arrival correspond to the largest sizes of the group. With the GKMP protocol, all the members are affected and hence suffers from the *1-affects-n* phenomenon. Typically, we notice in Figure 10 that for the inter-arrival in $[1s : 5s]$, the number of affected members for the GKMP protocol is comprised between 1600 and 400. For the same range of inter-arrival values, Iolus, SDKM.1 and SDKM.2 reduce the number of affected members to the minimum (between 400 and 70). Therefore, our approaches and Iolus scale better to large and dynamic groups. Moreover, notice in Figure 11 that when the inter-arrival increases, and hence the group size decreases, Iolus performs always the same number of decryption / re-encryption operations (5 operations). However, SDKM reduces this overhead while maintaining

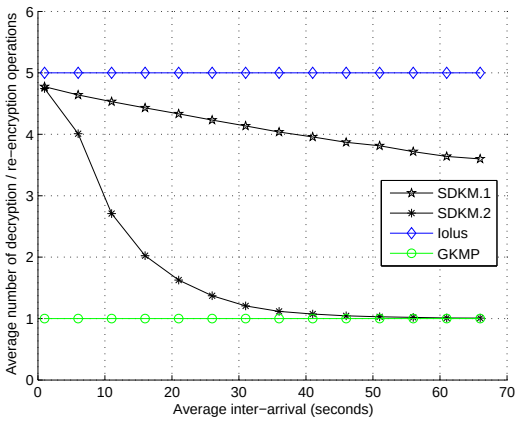

Figure 11: Impact of inter-arrival variation on decryption / re-encryption overhead. $\frac{\alpha}{\beta} = 2$, average membership duration=30min, $\theta = 15min$

low *1-affects-n* overhead. For the smallest groups, SDKM reaches the same performance of the GKMP protocol in performing a single encryption at the source and decryption at receivers.

We can notice the same phenomenon by varying the average membership duration of the members in the session. In Figure 12, we remark that SDKM.2 and Iolus scale better to large groups. We remark also that SDKM.1 scales better than the GKMP protocol.

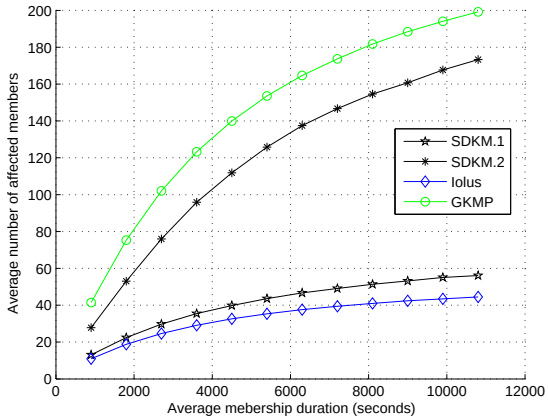

Figure 12: Impact of membership duration variation on *1-affects-n* overhead. $\frac{\alpha}{\beta} = 2$, average inter-arrival=20s, $\theta = 15min$

In Figure 13, we remark again that SDKM.2 has the advantage over SDKM.1 and Iolus of decreasing the decryption / re-encryption without dramatically increasing the *1-affects-n* overhead. When membership duration increases, SMs have tendency to be passive, and thus the number of decryption/re-encryption operations dereases.

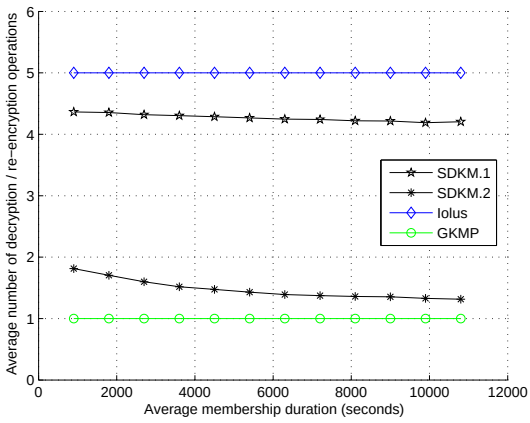

Figure 13: Impact of membership duration on decryption / re-encryption overhead. $\frac{\alpha}{\beta} = 2$, average inter-arrival=20s, $\theta = 15min$

7.3 Discussion

We demonstrated through simulations that SDKM, especially SDKM.2, provides a great flexibility and many parameters to tune the whole architecture to suit better the application level requirements. We showed also that SDKM scales better to large and dynamic groups while taking into account the delay in packet delivery at each subgroup level. We notice also that SDKM signaling does not increase dramatically the complexity of key management, because most of the signaling can be incorporated in usual messages, such as requesting a new TEK, delivery of a new TEK. The new signaling relates mainly to split / merge operations which happen rarely compared to re-keying due to membership changes. We can add in conclusion that SDKM.1 is a suitable solution if we want to gather in the same cluster subgroups with similar membership change frequencies. SDKM.2 is an efficient and scalable solution that completes SDKM.1 with offering the possibility to isolate very dynamic subgroups. However, SDKM.1 can be improved by offering a mechanism which controls the affectation of members to subgroups.

Our proposed protocol can be enhanced to tolerate the failure of some SMs: One solution can consist in using failure detectors of Chandra and Toueg [15]. Another solution can consist in using a list of SMs to manage the same subgroup, and if a SM is suspected faulty it will be replaced by the following SM in the list. The size of the list is fixed according to the quality of hosts used as Subgroup Managers.

8 Conclusion

The distributed nature of multicasting has a huge impact on security efficiency. On one hand, a multicast distribution tree can span large networks where members may be tremendously distant from each other. On the other hand, the efficiency of multicast security mechanisms can be severely affected by some phenomena that depend on their occurrence location in the network. One keystone phenomenon in the efficiency and scalability of group key management is the *dynamism of group members* which is likely to be different from a location to another, and from a moment to another during the whole multicast session. Given this group communication feature, we proposed in this paper a new group key management approach that takes into consideration the distributed nature of multicasting. The *dynamism awareness* of our approach allows to reach high levels of scalability and better performance trade-offs. Indeed, simulations demonstrated that the SDKM scheme scales to large and highly dynamic groups without tremendously affecting data path while mitigating the *1-affects-n* phenomenon. Moreover, parameters of the proposed scheme allow to customize it to support applications with different features such as synchronization requirements and tolerance to latencies in packet delivery.

References

[1] K. Almeroth and M. Ammar, "Collecting and modelling the join/leave behaviour of multicast group members in the Mbone," in *Proceedings of the High Performance Distributed Computing (HPDC'96)* , pp. 209-216, 1996.

[2] K. Almeroth and M. Ammar, "Multicast group behaviour in the internet's multicast backbone (Mbone)," *IEEE communications Magazine*, vol. 35, no. 6, pp. 124-129, 1997.

[3] K. Almeroth, "The evolution of multicast: From the MBone to inter-domain multicast to Internet2 deployment," *IEEE Network*, vol. 14, pp. 10-20, Jan.-Feb. 2000.

[4] American National Standards Institute, *Triple Data Encryption Algorithm Modes of Operation*, ANSI X9.52-1998, 1998.

[5] J. P. Avognon and Z. T. Li, "New multicast technology 'Survey and Security Concerns'," *Information Technology Journal*, vol. 3, no. 1, pp. 95-105, 2004.

[6] D. Balenson, D. McGrew, and A. Sherman, *Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization*, Internet-Draft, draft-balenson-groupkeymgmt-oft-00.txt, Feb. 1999.

[7] A. Ballardie, *Scalable Multicast Key Distribution*, RFC 1949, May 1996.

[8] A. Ballardie, *Core Based Trees (CBT version 2) Multicast Routing protocol specification*, RFC 2189, Sep. 1997.

[9] M. Baugher, B. Weis, T. Hardjono, and H. Harney, *The Group Domain of Interpretation*, RFC 3547, Jul. 2003.

[10] G. Belch, S. Greiner, and K. S. Trivedi H. d. Meer, *Queueing Networks and Markov Chains,Modeling and Performance Evaluation with Computer Science Applications*, Wiley & sons edition, 1998.

[11] B. Briscoe, "MARKS: Multicast key management using arbitrarily revealed key sequences," in *1st International Workshop on Networked Group Communication*, pp. 301-320, Nov. 1999.

[12] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas, "Multicast Security: A taxonomy and efficient constructions," in *IEEE INFOCOM*, pp. 708-716, Mar. 1999.

[13] Y. Challal, H. Bettahar, and A. Bouabdallah, "SAKM: a scalable and adaptive key management approach for multicast communications," *ACM SIGCOMM Computer Communications Review* , vol. 34, no. 2, pp. 55-70, 2004.

[14] K. C. Chan and S. H. G. Chan, "Distributed servers approach for large-scale secure multicast," *The IEEE Journal On Selected Areas in Communications*, vol. 20, no. 8, pp. 1500-1510, Oct. 2002.

[15] T. D. Chandra, and S. Toueg, "Unreliable failure detectors for reliable distributed systems," *Journal of the ACM*, vol. 43, no. 2, pp. 225-267, Mar. 1996.

[16] B. DeCleene, L. Dondeti, S. Griffin, T. Hardjono, D. Kiwior, J. Kurose, D. Towsley, S. Vasudevan, and C. Zhang, "Secure group communications for wireless networks", in *MILCOM'01*, pp. 113-117, June 2001.

[17] W. Diffie, and M.E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, pp. 644-654, Nov. 1976.

[18] C. Diot, "Reliability in multicast services and protocols ; A survey," in *Proceedings Conference LAN & MAN*, pp. 295-313, Dec. 1994.

[19] C. Diot, B. N. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," *IEEE Network*, vol. 14, no. 1, pp. 78-88, 2000.

[20] L. Dondeti, S. Mukherjee, and A. Samal, *A Distributed Group Key Management Scheme for Secure Many-to-Many Communication*, Technical Report PINTL-TR-207-99, 1999.

[21] L. R. Dondeti, S. Mukherjee, and A. Samal, "Comparison of hierarchical key distribution schemes," *IEEE Globcom Global Internet Symposium*, pp. 1774-1778, 1999.

[22] L. R. Dondeti, S. Mukherjee, and A. Samal, *Survey and Comparison of Secure Group Communication Protocols*, Technical Report, University of Nebraska-Lincoln, 1999.

[23] L. R. Dondeti, S. Mukherjee, and A. Samal, "Scalable secure one-to-many group communication using dual encryption," *Computer Communications*, vol. 23, no. 17, pp. 1681-1701, Nov. 2000.

[24] T. Dunigan and C. Cao, *Group Key Management*, Technical Report ORNL/TM-13470, 1998.

[25] Federal Information Processing Standards Publication, *Data Encryption Standard (DES)*, FIPS PUB 46, Dec. 1993.

[26] Federal Information Processing Standards Publication, National Institute of Standards and Technology (NIST), *Digital Signature Standard (DSS)*, FIPS PUB 186-2, Jan. 2000.

[27] Federal Information Processing Standards Publication, *Advanced Encryption Standard (AES)*, FIPS PUB 197, Nov. 2001.

[28] T. Hardjono, B. Cain, and I. Monga, *Intra-domain Group Key Management for Multicast Security*, IETF Internet draft, Sep. 2000.

[29] H. Harney, and C. Muckenhirn, *Group Key Management Protocol (GKMP) Architecture*, RFC 2093, July 1997.

[30] H. Harney and C. Muckenhirn, *Group Key Management Protocol (GKMP) Specification*, RFC 2094, July 1997.

[31] J. H. Huang and S. Mishra, "Mykil: A highly scalable and efficient key distribution protocol for large group multicast," *IEEE GLOBECOM*, pp. 1476-1480, 2003.

[32] J. H. Huang and S. Mishra, "Support for mobility and fault tolerance in mykil," in *Proceedings of the 2004 International Conference on Dependable Systems and Networks (DSN04)*, pp. 537-246, 2004.

[33] I. Ingemarson, D. Tang, and C. Wong, "A conference key distribution system," *IEEE Transactions on Information Theory*, vol. 28, no. 5, pp. 714-720, Sep. 1982.

[34] P. Judge and M. Ammar, "Security issues and solutions in multicast content distribution: A survey," *IEEE Network*, vol. 17, no. 1, pp. 30-36, Jan.-Feb. 2003.

[35] Y. Kim, A. Perrig, and G. Tsudik, "Simple and fault-tolerant key agreement for dynamic collaborative groups," in *7th ACM Conference on Computer and Communications Security*, pp. 235-244, Nov. 2000.

[36] D. A. McGrew, and A. T. Sherman, "Key establishement in large dynamic groups using one-way function trees," *IEEE Transactions On Software Engeneering*, vol. 29, no. 5, pp. 444-458, May 2003.

[37] S. Mittra, "Iolus: A framework for scalable secure multicasting," in *ACM SIGCOMM*, pp. 277-288, 1997.

[38] R. Molva, and A. Pannetrat, "Scalable multicast security in dynamic groups," in *6th ACM Conference on Computer and Communication Security*, pp. 101-112, Nov. 1999.

[39] R. Mukherjee, and J.W. Atwood, "Proxy encryptions for secure multicast key management," in *IEEE Local Computer Networks (LCN'03)*, pp. 377-384, Oct. 2003.

[40] R. Mukherjee and J. W. Atwood, "SIM-KM: Scalable infrastructure for multicast key management," in *IEEE Local Computer Networks (LCN'04)*, pp. 335-342, Nov. 2004.

[41] A. Perrig, "Efficient collaborative key management protocols for secure autonomous group communication," in *International Workshop on Cryptographic techniques and E-commerce*, pp. 192-202, 1999.

[42] A. Perrig, D. Song, and J.D. Tygar, "ELK, a new protocol for efficient large-group key distribution," in *IEEE Security and Priavcy Symposium*, pp. 247-262, May 2001.

[43] R. Poovendram, S. Ahmed, S. Corson, and J. Baras, "A scalable extension of group key management protocol," in *2nd Annual ATRIP Conference*, pp. 187-191, Feb. 1998.

[44] B. Quinn, and K. Almeroth, *IP Multicast Applications: Challenges and Solutions*, RFC 3170, Sep. 2001.

[45] S. Rafaeli and D. Hutchison, "Hydra: a decentralized group key management," in *11th IEEE International WETICE: Enterprise Security Workshop*, pp. 62- 67, June 2002.

[46] S. Rafaeli and D. Hutchison, "A survey of key management for secure group communication," *ACM Computing Surveys*, vol. 35, no. 3, pp. 309-329, Sep. 2003.

[47] R. Rivest, *The MD5 Message-Digest Algorithm*, RFC 1321, Apr. 1992.

[48] R. L. Rivest, A. Shamir, and L. M. Adelman,"A method for obtaining digital signatures and public-key cyptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.

[49] O. Rodeh, K. Birman, and D. Dolev, "Optimized group rekey for group communication systems," in *Network and Distributed System Security*, pp. 39-48, Feb. 2000.

[50] H. Seba, A. Bouabdallah, and N. Badache, "A new approach to scalable and fault-tolerant group key management protocols," *Journal of High Speed Networks*, vol. 13, no. 4, pp. 283-296, 2004.

[51] S. Setia, S. Koussih, S. Jajodia, and E. Harder, "Kronos: A scalable group re-keying approach for secure multicast," in *IEEE Symposium on Security and Privacy*, pp. 215-228, May 2000.

[52] C. Shields and J. J. Garcia-Luna-Aceves, "KHIP-A scalable protocol for secure multicast routing," *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 4, pp. 53-64, Oct. 1999.

[53] J. Snoeyink, S. Suri, and G. Vorghese, "A lower bound for multicast key distribution," in *IEEE INFOCOM'01*, pp. 422-431, 2001.

[54] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman key distribution extended to group communication," in *3rd ACM Conference on Computer and Communications Security*, pp. 31-37, Mar. 1996.

[55] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B. Plattner, "The versaKey framework: Versatile group key management," *IEEE Journal on Selected Areas in Communications (Special Issues on Middleware)*, vol. 17, no. 8, pp. 1614-1631, Aug. 1999.

[56] D. Wallner, E. Harder, and R. Agee, *Key Management for Multicast: Issues and Architecture*. RFC 2627, National Security Agency, Jun. 1999.

[57] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," in *ACM SIGCOMM*, pp. 68-79, 1998.

[58] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 16-30, Feb. 2000.

[59] Y. R. Yang, X. S. Li, X. B. Zhang, and S. S. Lam, *Reliable Group Rekeying: A Performance Analysis*, in *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 27-38, 2001.

[60] S. Zhu and S. Jajodia, "Scalable group rekeying for secure multicast: A survey," in *Proceedings of 5th International Workshop on Distributed Computing*, vol. 2918, pp. 1-10, 2004.

Saïd Gharout received the engineer degree from the National Institute of Computer Science (INI-Algeria) in 2003 and the magister degree from the University of Bejaia (Algeria) in 2005. Currently, he is a PhD student at the University of Bejaia. His research interests include group communication security and Mobile IP.

Yacine Challal is a research assistant at Compiegne University of Technology (UTC-France). He got his master degree and PhD from the UTC respectively in 2002 and 2005. He graduated and received the engineer degree from the National Institute of Computer Science (INI-Algeria) in 2001. His research interests include: group communication security, security and routing in ad hoc and wireless sensor networks.

Abdelmadjid Bouabdallah received the engineer diploma in computer science from university of technology of Algiers (USTHB) in 1986, and received the Master (DEA) degree and Ph.D. from university of Paris-sud Orsay (France) respectively in 1988 and 1991. From 1992 to 1996, he was Assistant Professor at university of Evry-Val-d'Essonne, France. Since 1996, he is Professor in the department of Computer Engineering at University of Technology of Compiegne (UTC) where he is leader of Networking and Optimization research group. His research Interest include Internet QoS and security, unicast/multicast communication, and fault tolerance in wired/wireless networks and distributed systems. He conducted severeal important research project founded by Motorola, France telecoms, RNRT,etc.