HAL
open science

Modélisation des propriétés de sécurité de protocoles de groupe

Najah Chridi, Laurent Vigneron

To cite this version:

Najah Chridi, Laurent Vigneron. Modélisation des propriétés de sécurité de protocoles de groupe. 1er Colloque sur les Risques et la Sécurité d'Internet et des Systèmes, CRiSIS, Oct 2005, Bourges, France, pp.119-132. inria-00000606

HAL Id: inria-00000606

https://inria.hal.science/inria-00000606v1

Submitted on 7 Nov 2005

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Modélisation des propriétés de sécurité de protocoles de groupe

Najah Chridi et Laurent Vigneron*

LORIA – UHP-UN2 (UMR 7503)
BP 239, 54506 Vandœuvre-lès-Nancy Cedex, France
{chridi,vigneron}@loria.fr

Résumé. Nous proposons un modèle pour les protocoles de groupe et plus généralement pour les protocoles dits contribuants. Ce modèle permet de décrire un protocole contribuant, d'étudier ses caractéristiques et ses propriétés de sécurité. Cette étude permet de détecter différents types d'attaques. Nous avons appliqué le modèle sur plusieurs protocoles, tels que A-GDH.2, SA-GDH.2, Asokan-Ginzboorg et Bresson-Chevassaut-Essiari-Pointcheval, ce qui nous a permis de mettre en évidence différents types d'attaques possibles sur chacun.

1 Introduction

Avec le succès réalisé dans la vérification de protocoles relativement classiques [2,1], de nombreux travaux récents s'orientent vers de nouveaux types de protocoles, beaucoup plus complexes. L'une de ces classes de protocoles concerne les protocoles de groupe [21,11,22,20]. Un protocole de groupe est un protocole ou une suite de sous-protocoles pour l'établissement d'une clé secrète entre un nombre non borné d'agents. Ceci peut prendre la forme d'un établissement simple de clés, exécuté entre les agents, ou d'une série de requêtes pour joindre ou quitter le groupe avec les mises-à-jour associées à la clé. Pour assurer la communication au sein d'un groupe, les membres ont généralement recours à une clé secrète pour sécuriser leurs communications. L'opération la plus cruciale et la plus délicate dans ce type de protocoles est alors la gestion de clés. Des protocoles dédiés à cette opération ont été et continuent d'être le sujet de nombreuses études de recherche [4,3,10,5].

La vérification de protocoles de groupe se confronte à plusieurs problèmes. En effet, la sécurité des communications au sein de groupes n'est pas nécessairement une extension d'une communication sécurisée entre deux parties [21]. Elle est beaucoup plus compliquée. Une fois la communication commencée, le groupe peut changer de structure en ajoutant ou en supprimant un ou plusieurs membres. Les services de sécurité sont alors plus élaborés. En outre, les protocoles de groupe mettent en cause un nombre non borné de participants. Cependant, la plupart des approches automatisées de vérification de protocoles nécessitent un modèle concret. La taille du groupe doit alors être fixée à l'avance. Or, cette contrainte restreint considérablement le nombre d'attaques détectables. Ensuite, vu que les besoins en sécurité sont généralement liés aux protocoles, il est très difficile de décrire les propriétés que le protocole doit satisfaire. Cette phase de spécification des propriétés est critique : toute erreur (de compréhension ou de formalisation de ces propriétés) peut engendrer de fausses failles de sécurité. Des efforts ont été effectués pour une spécification des propriétés indépendamment des applications visées, à l'aide d'abstractions par exemple [7]. Des langages formels ont également servi à définir des propriétés comme l'authentification [12] : un message est accepté par un agent si et seulement si, il l'a effectivement demandé à un second agent, et ce dernier le lui a bien envoyé.

* Ce travail est soutenu par l'ACI Sécurité SATIN, et le projet RNTL 03V360 Prouvé.

La définition et la spécification des propriétés de sécurité est donc un passage obligé très délicat, avant même d'entamer la vérification. La difficulté est d'autant plus grande lorsqu'il faut considérer des propriétés non standards, comme celles liées à la dynamicité des protocoles de groupe.

Dans cet article, nous proposons un modèle pour les protocoles de groupe et plus généralement pour les protocoles contribuants. Ce modèle permet de décrire un protocole contribuant, d'étudier ses caractéristiques et ses propriétés de sécurité, et donc d'identifier les différents types d'attaques possibles. Chaque notion introduite est suivie d'un exemple basé sur le protocole A-GDH.2 [16]. Nous mentionnons également les résultats de l'application de notre modèle sur d'autres protocoles, puis nous citons les travaux existants.

2 Un modèle pour les protocoles de groupe

Comme tout protocole de communication, un protocole de groupe peut être vu comme un échange de messages entre plusieurs participants. Cet échange est habituellement décrit par la liste des actions réalisées par chaque participant lors d'une exécution normale du protocole (exécution sans intervention d'un intrus). Par exemple, le protocole A-GDH.2 pour trois participants peut être décrit par la séquence de messages suivante, où r_i est un nombre aléatoire engendré par le participant a_i et α désigne un générateur commun à tous les membres du groupe :

$$\begin{array}{l} a_1 \longrightarrow a_2 : \alpha, \alpha^{r_1} \\ a_2 \longrightarrow a_3 : \alpha^{r_2}, \alpha^{r_1}, \alpha^{r_1 r_2} \\ a_3 \longrightarrow a_1 : \alpha^{r_2 r_3 k_{13}} \\ a_3 \longrightarrow a_2 : \alpha^{r_1 r_3 k_{23}} \end{array}$$

Ce protocole est constitué de quatre messages entre trois participants. La phase la plus cruciale de ce protocole, comme tout protocole de groupe, est celle du calcul de la clé de groupe. Chacun des participants doit savoir comment agir suite à la réception de toutes les informations nécessaires pour le calcul de la clé de groupe notée $\mathcal{K}_G$. Nous désignons par $\mathcal{A}\mathbf{lg_i}$ l'algorithme effectué par le $i^{ème}$ participant, prenant comme paramètre les informations nécessaires au calcul de la clé et donnant comme résultat la clé du groupe, selon le point de vue du participant i.
Pour l'exemple de A-GDH.2, le participant a_1 calcule la clé du groupe en appliquant l'algorithme $\mathcal{A}lg_1(r_1, k_{13}, X_1) = X_1^{r_1/k_{13}}$, où X_1 est la variable désignant le message $\alpha^{r_2 r_3 k_{13}}$. La notion de variable est introduite vu que a_1 ne peut pas vérifier le contenu du message reçu. De même, a_2 calcule $\mathcal{A}lg_2(r_2, k_{23}, X_2) = X_2^{r_2/k_{23}}$, où X_2 est la variable désignant $\alpha^{r_1 r_3 k_{23}}$.
Enfin, a_3 calcule $\mathcal{A}lg_3(r_3, \emptyset, X_3) = X_3^{r_3}$, où X_3 désigne le message $\alpha^{r_1 r_2}$. Il est à noter que dans cet exemple, tous les membres calculent la même clé du groupe ($\mathcal{K}_G = \alpha^{r_1 r_2 r_3}$). Tout au long de cet article, nous allons travailler sur cet exemple pour illustrer les différentes notions introduites.

2.1 Description du modèle

Le modèle que nous proposons sert au départ à modéliser un protocole de groupe. Il consiste à définir un système global de plusieurs composantes qui interagissent entre elles. Ainsi, un protocole de groupe peut être vu comme un système représenté par un triplet $<\mathcal{A}, \mathcal{K}, \mathcal{S}>$, avec,

$\mathcal{A}$: ensemble des agents membres du groupe,
$\mathcal{K}$: ensemble des connaissances des participants,
$\mathcal{S}$: ensemble des services.

Nous désignons par *service* la *contribution* d'un participant pour engendrer la clé de groupe. La contribution d'un agent a_i à un autre agent a_j est toute information engendrée par a_i, utile pour a_j afin de déduire la clé de groupe. Soit $i \in \mathbb{N}$, notre modèle se base sur les définitions suivantes :

- $\mathbf{a_i} \in \mathcal{A}$ désigne le $i^{ème}$ participant ;
- $\mathbf{S_i} \subseteq \mathcal{S}$ est l'ensemble *minimal* des services utiles à a_i pour construire la clé de groupe ;
- $\mathcal{K}_\mathbf{i} \subseteq \mathcal{K}$ est l'ensemble des connaissances *privées* de a_i. Il s'agit de l'ensemble minimal des connaissances utiles pour construire les services et la clé finale. Il inclut les connaissances privées au départ de la session ainsi que les informations engendrées au cours du déroulement de la session ;
- $\mathcal{K}_\mathbf{ij} \subseteq \mathcal{K}$ est l'ensemble des connaissances *partagées* entre les agents a_i et a_j. C'est l'ensemble minimal des connaissances partagées et utiles pour la génération de la clé de groupe. Ces informations sont données par la spécification du protocole. Il est à noter que $\mathcal{K}_{ij} = \mathcal{K}_{ji}$.

En plus des services utiles pour engendrer la clé de groupe, on distingue d'autres sous-ensembles de services utilisés pour regrouper les services offerts par chaque agent. Ainsi, nous notons S_{a_i} le sous-ensemble de services auxquels a_i a *contribué* (directement ou indirectement) par l'utilisation d'informations privées.

$$\mathbf{S_{a_i}} = \{s \in \mathcal{S} \mid \exists t \text{ sous-terme de } s, \text{ tel que } t \in \mathcal{K}_i\}$$

Exemple 1. Si nous décrivons le protocole A-GDH.2 dans notre modèle, nous obtenons :

$$\begin{aligned}
&\mathcal{A} = \{a_1, a_2, a_3\},\\
&\mathcal{K} = \{r_1, r_2, r_3, k_{13}, k_{23}\},\\
&\mathcal{S} = \{\alpha, \alpha^{r_1}, \alpha^{r_2}, \alpha^{r_1 r_2}, \alpha^{r_1 r_3 k_{23}}, \alpha^{r_2 r_3 k_{13}}\}
\end{aligned}$$

Les informations relatives aux agents sont illustrées dans le tableau suivant :

$\mathbf{a_i}$	$\mathbf{S_i}$	$\mathbf{S_{a_i}}$	$\mathcal{K}_\mathbf{i}$	$\cup_\mathbf{j}\mathcal{K}_\mathbf{ij}$
$\mathbf{a_1}$	$\alpha^{r_2 r_3 k_{13}}$	$\alpha^{r_1}, \alpha^{r_1 r_2}, \alpha^{r_1 r_3 k_{13}}$	r_1	k_{13}
$\mathbf{a_2}$	$\alpha^{r_1 r_3 k_{23}}$	$\alpha^{r_2}, \alpha^{r_1 r_2}, \alpha^{r_2 r_3 k_{13}}$	r_2	k_{23}
$\mathbf{a_3}$	$\alpha^{r_1 r_2}$	$\alpha^{r_2 r_3 k_{13}}, \alpha^{r_1 r_3 k_{23}}$	r_3	k_{13}, k_{23}

En résumé, notre modèle est donc basé sur un triplet $<\mathcal{A}, \mathcal{K}, \mathcal{S}>$, décrivant que chaque agent a_i *possède* des informations confidentielles ($\mathcal{K}_i$), *partage* des informations ($\mathcal{K}_{ij}$) avec d'autres agents, *a besoin* de connaissances (S_i) pour engendrer la clé de groupe et *contribue* (S_{a_i}) à la génération de la clé pour les autres agents du groupe.

2.2 Caractéristiques du modèle

Notre système est donc composé de trois ensembles de base $\mathcal{A}$, $\mathcal{K}$, et $\mathcal{S}$, à partir desquels sont construits de nombreux autres ensembles ($\mathcal{K}_i$, $\mathcal{K}_{ij}$, S_i, S_{a_i}). La construction de ces ensembles pour un protocole donné est assez simple, mais ils doivent cependant satisfaire quelques propriétés indispensables. Nous décrivons ci-dessous ces propriétés, qui concernent essentiellement les interactions entre ces différents ensembles. Ces caractéristiques seront ensuite illustrées par l'exemple de A-GDH.2.

Caractéristique 1 (Unicité des identificateurs des agents) *Les identités des agents doivent être différentes.*

$$\forall a_i \in \mathcal{A}, \forall a_j \in \mathcal{A}, i \neq j \Longrightarrow a_i \neq a_j$$

Caractéristique 2 (Visibilité des connaissances privées) *Les connaissances dites « privées » des agents doivent être réellement confidentielles. Elles ne peuvent pas être partagées :*

$$\forall a_i \in \mathcal{A}, \forall a_j \in \mathcal{A}, i \neq j \Longrightarrow \mathcal{K}_i \cap \mathcal{K}_j = \emptyset$$

$$\forall a_i \in \mathcal{A}, \forall t \in \mathcal{K}_i, \forall a_j, a_k \in \mathcal{A}, t \notin \mathcal{K}_{jk}$$

De plus, elles ne peuvent pas être diffusées en clair :

$$\forall a_i \in \mathcal{A}, \forall t \in \mathcal{K}_i, t \notin \mathcal{S}$$

Caractéristique 3 (Visibilité des connaissances partagées) *Les connaissances partagées entre plusieurs agents sont en fait des connaissances confidentielles vis-à-vis des autres agents. Chaque agent sait avec qui il partage des connaissances :*

$$\forall a_i, a_j \in \mathcal{A}, \forall t \in \mathcal{K}_{ij}, (\exists a_k, a_l \in \mathcal{A}, k \neq i, j \wedge t \in \mathcal{K}_{kl} \Longrightarrow t \in \mathcal{K}_{ik} \cap \mathcal{K}_{kj})$$

Et comme pour les connaissances privées, les connaissances partagées ne doivent pas être transmises en clair :

$$\forall a_i, a_j \in \mathcal{A}, \forall t \in \mathcal{K}_{ij}, t \notin \mathcal{S}$$

Caractéristique 4 (Distinction des services utiles) *Les ensembles des services nécessaires à la construction de la clé de groupe pour deux agents doivent être différenciés par au moins un élément.*

$$\forall a_i, a_j \in \mathcal{A}, i \neq j \Longrightarrow S_i \neq S_j$$

Caractéristique 5 (Indépendance des services utiles) *Les ensembles de services nécessaires à la construction de la clé de groupe pour deux agents ne doivent pas être liés par une relation d'inclusion.*

$$\forall a_i, a_j \in \mathcal{A}, i \neq j \Longrightarrow S_i \nsubseteq S_j$$

Caractéristique 6 (Correspondance des services) *Tout élément de S_i doit correspondre à un élément d'un S_{a_k}. Cela signifie que les services nécessaires à un agent a_i pour construire la clé doivent provenir de contributions d'autres agents.*

$$\forall a_i \in \mathcal{A}, \forall s \in S_i, \exists a_k \in \mathcal{A}, s \in S_{a_k}$$

Les caractéristiques définies précédemment concernent la construction des ensembles de connaissances et de services attachés aux agents. Ces informations sont utilisées par les agents pour construire la clé de groupe $\mathcal{K}_G$. Cette phase de déduction (notée $\models$) de $\mathcal{K}_G$, intervenant à la fin de la session du protocole, doit également être contrôlée. Nous désignons par *déduction* l'application de certaines règles de composition et de décomposition de messages. Mise à part l'hypothèse standard de chiffrement parfait, les règles de déduction comprennent des opérateurs algébriques tels que l'exponentiation, l'inverse... Ces règles doivent donc tenir compte des propriétés de ces opérateurs, comme par exemple l'associativité et la commutativité des exposants dans une exponentiation. Elles s'accompagnent également de simplifications, liées aux propriétés de ces opérateurs.

Caractéristique 7 (Déduction de la même clé de groupe) *Pour un agent a_i, la clé de groupe est engendrée en appliquant l'algorithme $\mathcal{A}lg_i$ aux connaissances privées et partagées, ainsi qu'aux services nécessaires de cet agent.*

$$\mathcal{A}lg_i(\mathcal{K}_i, \cup_{a_k \in \mathcal{A}} \mathcal{K}_{ik}, S_i) \models \mathcal{K}_G$$

Tous les membres du groupe doivent déduire la même clé. L'application de l'algorithme doit donc donner le même résultat pour tous les membres.

$$\forall a_i \in \mathcal{A}, \mathcal{A}lg_i(\mathcal{K}_i, \cup_{a_k \in \mathcal{A}} \mathcal{K}_{ik}, S_i) \models \mathcal{K}_G$$

Caractéristique 8 (Services minimaux) *Un groupe de services est dit minimal si l'accomplissement de l'objectif auquel il est destiné nécessite la participation de tous les éléments du groupe.*
Dans le cadre de notre système, toutes les contributions doivent être utilisées pour la génération de la clé de groupe. Un agent a_i ne peut pas se limiter à un sous-ensemble S'_i de S_i pour déduire la clé.

$$\forall a_i \in \mathcal{A}, \nexists S'_i \subset S_i, S'_i \cup \mathcal{K}_i \cup_j \mathcal{K}_{ij} \models \mathcal{K}_G$$

Un agent a_i doit donc avoir recours à tout l'ensemble S_i en tenant compte de ses connaissances privées et partagées. Une conséquence est la propriété suivante : aucun élément de S_i ne peut être déduit des autres éléments de cet ensemble.

Exemple 2. Nous revenons à notre exemple de A-GDH.2 pour lequel, nous étudions une à une les caractéristiques définies précédemment.

1. L'unicité des identificateurs des agents est vérifiée.
2. La caractéristique de visibilité des connaissances privées est vérifiée. En effet, les ensembles des connaissances privées sont disjoints :

   $$\{r_1\} \cap \{r_2\} \cap \{r_3\} = \emptyset$$

   Et d'autre part,

   $$r_1 \notin \mathcal{S}, r_2 \notin \mathcal{S}, \text{ et } r_3 \notin \mathcal{S}$$

3. La caractéristique de visibilité des connaissances partagées est vérifiée puisque $k_{13} \notin \cup_j \mathcal{K}_{2j}$ et $k_{23} \notin \cup_j \mathcal{K}_{1j}$. En plus, $k_{13} \notin \mathcal{S} \wedge k_{23} \notin \mathcal{S}$.
4. Les propriétés de distinction de services utiles et d'indépendance des services utiles sont vérifiées car :

   $$S_1 \cap S_2 = S_1 \cap S_3 = S_2 \cap S_3 = \emptyset$$

5. La caractéristique de correspondance de services est vérifiée pour les trois membres a_1, a_2 et a_3. En effet,
   – pour a_1, $S_1 \subset S_{a_2}$ et $S_1 \subset S_{a_3}$;
   – pour a_2, $S_2 \in S_{a_3}$ et $S_2 \in S_{a_1}$;
   – pour a_3, $S_3 \subset S_{a_1}$ et $S_3 \subset S_{a_2}$.
6. La caractéristique de déduction de la même clé de groupe est vérifiée. Tout d'abord, la clé du groupe est calculée par chacun des membres comme suit :

   $$\begin{aligned} \mathcal{A}lg_1(r_1, k_{13}, \alpha^{r_2 r_3 k_{13}}) &= \alpha^{r_2 r_3 k_{13} * r_1 / k_{13}} \models \alpha^{r_1 r_2 r_3} \\ \mathcal{A}lg_2(r_2, k_{23}, \alpha^{r_1 r_3 k_{23}}) &= \alpha^{r_1 r_3 k_{23} * r_2 / k_{23}} \models \alpha^{r_1 r_2 r_3} \\ \mathcal{A}lg_3(r_3, \emptyset, \alpha^{r_1 r_2}) &= \alpha^{r_1 r_2 * r_3} \models \alpha^{r_1 r_2 r_3} \end{aligned}$$

   Tous les membres du groupe déduisent donc la même clé ($\alpha^{r_1 r_2 r_3}$).
7. La caractéristique de services minimaux est vérifiée. En effet, pour les trois agents, l'ensemble S_i est l'ensemble minimal à partir duquel on arrive à engendrer la clé du groupe.

2.3 Évolution du système dans le temps

Le modèle introduit dans la section 2.1 décrit un *état stable* du système. Il s'agit d'une instance d'une génération de clés d'un protocole de groupe. Le processus de cette génération est traduit par les relations entre les différents ensembles de l'état du système. Néanmoins, un groupe de membres n'est en général pas stable au cours du temps. Il peut évoluer suite aux différentes opérations telles que l'ajout, la suppression d'un membre ou d'un ensemble de membres. Le système n'a donc pas intérêt à se limiter à un seul état. L'évolution du système se traduit par la modification d'au moins une des trois composantes. Par exemple, l'ajout ou la suppression d'un membre modifie l'ensemble $\mathcal{A}$. Afin de prendre en compte cette dynamique du système, nous allons étendre le modèle proposé dans la section 2.1 en introduisant la notion de temps. Soit T l'ensemble (infini) des entiers naturels représentant le temps. Le modèle proposé auparavant ne représente que l'état observé à un instant τ, noté :

$$GP^{\tau} = <\mathcal{A}, \mathcal{K}, \mathcal{S}>^{\tau} = <\mathcal{A}^{\tau}, \mathcal{K}^{\tau}, \mathcal{S}^{\tau}>$$

Le système global est donc donné par :

$$GP^{T} = \{GP^{\tau}, \tau \in T\}$$

Ordre temporel. L'ensemble T est muni d'un ordre noté "$<$". Soient τ et τ' deux instants de T. L'influence du temps dans le modèle du système, défini dans la section 2.1, peut être alors vue comme suit :
Si $\tau < \tau'$ alors GP^{τ} est un état qui *précède* temporellement $GP^{\tau'}$.
Si $\tau > \tau'$ alors GP^{τ} est un état qui *suit* temporellement $GP^{\tau'}$.
Si $\tau = \tau'$ alors il s'agit du *même* état GP^{τ}.

Définition d'un événement. Le passage d'un état stable à un autre état se fait suite à des *événements*. On désigne par événement toute opération capable de modifier l'un des constituants de l'état du système. Dans le cadre de protocoles de groupe, un événement peut être vu comme une opération d'ajout ou de suppression d'un membre du groupe, ou l'expiration d'un délai (par exemple pour un rafraîchissement de clé). Dans le cadre de notre modèle, un événement peut affecter l'ensemble des agents $\mathcal{A}$ (ajout ou élimination de participants), ou le sous-ensemble des connaissances privées d'au moins un agent. Cette propriété peut s'exprimer comme suit :

$$<\mathcal{A}, \mathcal{K}, \mathcal{S}>^{\tau} \longrightarrow_{event} <\mathcal{A}, \mathcal{K}, \mathcal{S}>^{\tau'}$$

si on a :

$$(\mathcal{A}^{\tau} \neq \mathcal{A}^{\tau'}) \vee (\mathcal{A}^{\tau} = \mathcal{A}^{\tau'} \wedge \exists a_i \in \mathcal{A}, \mathcal{K}_i^{\tau} \neq \mathcal{K}_i^{\tau'})$$

Le rafraîchissement de clés est un événement qui modifie les connaissances privées (comme par exemple le changement de nombre aléatoire r_i).

3 Formalisation des propriétés de sécurité

Nous distinguons dans cette section deux types de propriétés de sécurité. Cette distinction est basée sur l'indépendance ou la dépendance par rapport au temps. Pour pouvoir définir les attaques (non-vérification des propriétés), nous avons besoin de modéliser l'intrus. Un agent malhonnête ou *intrus* est soit un participant non officiel du protocole, soit un participant officiel

(ancien ou actuel membre du groupe) utilisant sa position avantageuse pour perpétrer des actions malhonnêtes.

Dans notre modèle, un intrus peut engendrer des actions laissant le système dans son état stable telles que l'accès aux différents services offerts ($\bigcup_{a_i \in \mathcal{A}} S_{a_i}$) ou l'interception ou la modification de messages échangés. Il peut aussi engendrer des actions provoquant un changement d'état comme l'envoi d'un message d'initialisation de session de protocole.

3.1 Propriétés indépendantes du temps

Il s'agit des propriétés de sécurité liées à un état stable du système. Dans cette section, nous modélisons les propriétés d'authentification implicite de clé, de secret de la clé, de confirmation de clé et d'intégrité.

Un protocole vérifie la propriété d'*authentification implicite* de la clé [18] si, à la fin de la session, chaque participant est assuré qu'aucun élément externe ne peut acquérir sa vue de la clé de groupe. La vue d'un participant a_i est constituée de l'application de l'algorithme $\mathcal{A}lg_i$ en prenant comme paramètres les informations nécessaires pour engendrer la clé du groupe ($\mathcal{K}_i \cup S_i \cup_j \mathcal{K}_{ij}$). Un intrus (noté a_I) peut avoir une telle vue s'il peut la déduire grâce aux différentes informations qu'il possède ($\mathcal{K}_I \cup_j \mathcal{K}_{Ij} \cup_{a_k \in \mathcal{A}} S_{a_k}$).

Propriété 1 (Authentification implicite de la clé) *La propriété d'authentification implicite de la clé pour un agent a_i, différent de l'intrus a_I, est exprimée comme suit :*

$$\mathcal{K}_I \cup_j \mathcal{K}_{Ij} \cup_{a_k \in \mathcal{A}} S_{a_k} \not\models \mathcal{A}lg_i(\mathcal{K}_i, \cup_j \mathcal{K}_{ij}, S_i)$$

Dans un état du système, seuls les membres du groupe peuvent engendrer la clé partagée. Un agent externe a_I peut violer cette propriété en déduisant la clé du groupe à partir des connaissances acquises ($\mathcal{K}_I \cup_j \mathcal{K}_{Ij} \cup_{a_k \in \mathcal{A}} S_{a_k}$).

Propriété 2 (Secret de la clé) *La violation de la propriété de secret est définie par :*

$$\mathcal{K}_I \cup_j \mathcal{K}_{Ij} \cup_{a_k \in \mathcal{A}} S_{a_k} \models \mathcal{K}_G$$

Étant donné que la clé du groupe peut être déduite à partir des informations d'un agent, la définition du secret de la clé est :

$$\forall a_i \in \mathcal{A}, \mathcal{K}_I \cup_j \mathcal{K}_{Ij} \cup_{a_k \in \mathcal{A}} S_{a_k} \not\models \mathcal{A}lg_i(\mathcal{K}_i, \cup_j \mathcal{K}_{ij}, S_i)$$

Cette définition est semblable à la propriété 1 sauf qu'ici, on raisonne sur tout le groupe ($\forall a_i \in \mathcal{A}$) au lieu de raisonner par rapport à un seul élément (a_i).

Dans le contexte de protocoles de groupe, il existe une propriété servant à convaincre une partie du groupe que les autres membres arrivent à engendrer la clé partagée : la *confirmation* de la clé. Comme la seule possibilité pour un agent de pouvoir engendrer la clé est d'avoir les services nécessaires, la confirmation de clé revient alors à la confirmation de ces services. Pour qu'un agent confirme avoir engendré la clé, il doit communiquer à chaque émetteur d'un service utile à la génération de la clé qu'il a reçu sa contribution.

Propriété 3 (Confirmation de la clé) *En considérant les membres deux à deux, un agent a_i confirme avoir reçu le service correspondant à un agent a_j afin de l'utiliser pour la génération de sa vue de la clé de groupe si :*

$$\exists s \in S_i, s \in S_{a_j}$$

La propriété d'*intégrité*, quant à elle, permet de vérifier que tous les agents contribuent à la clé de groupe et que toute personne extérieure au groupe ne doit pas participer à la génération de la clé partagée. La première condition peut être vérifiée si l'ensemble des services S_i de chaque agent a_i résulte des contributions de tous les autres membres du groupe. Il s'agit donc de la confirmation de la clé présentée dans la propriété 3. La deuxième condition peut être vérifiée si aucun élément de S_i ne provient d'une partie extérieure au groupe. Sinon, il y a nécessairement un intrus qui a contribué à la génération de la clé.

Propriété 4 (Intégrité) *L'intégrité de la clé est donc définie, pour tous les membres du groupe, d'une part par la confirmation des services utiles pour engendre la clé :*

$$\forall a_i, a_j \in \mathcal{A}\ (i \neq j), \exists s \in S_i, s \in S_{a_j}$$

et d'autre part, par la correspondance des services :

$$\forall a_i \in \mathcal{A}, \forall s \in S_i, \exists a_j \in \mathcal{A}, s \in S_{a_j}$$

Comme un intrus a_I pourrait ne participer qu'en partie à la conception d'un service nécessaire pour un participant, la propriété traitée exprime qu'aucun sous-terme des services utiles ne peut appartenir aux connaissances de l'intrus :

$$\forall a_i \in \mathcal{A}, \forall s \in S_i, \forall t \text{ sous-terme de } s, t \notin \mathcal{K}_I$$

3.2 Propriétés liées à l'évolution du système dans le temps

Certaines propriétés de sécurité ont un lien fort avec la notion de temps. L'une des plus importantes est l'*indépendance de clés de groupe*. Elle garantit qu'aucun sous-ensemble de clés ne peut être utilisé pour découvrir un autre sous-ensemble de clés qui lui est disjoint. Il s'agit en fait de la combinaison de deux propriétés de sécurité : le secret futur (*forward secrecy*) et le secret passé (*backward secrecy*).

***a*) Secret futur.**
Cette propriété garantit qu'un adversaire passif qui connaît un certain nombre d'anciennes clés de groupe ne peut pas découvrir une clé de groupe plus récente. Soit par exemple [10] la séquence de clés de groupe suivante : $\{K_0, \ldots, K_m\}$. La propriété traitée exprime que du sous-ensemble $\{K_0, K_1, \ldots, K_i\}$ l'intrus passif ne peut pas découvrir une clé de groupe K_j telle que $j > i$. Selon notre modèle, pour pouvoir définir une telle propriété, il est nécessaire de revenir sur la définition de la propriété de *Secret de la clé de groupe*. En effet, le secret d'une clé de groupe est relatif à un instant donné : il est impossible (du point de vue calculatoire) de découvrir la clé à partir des informations de l'instant [10]. Formellement, on a :

$$\forall \tau \in T, \mathcal{K}_I^\tau \cup_{a_j \in \mathcal{A}^\tau} \mathcal{K}_{Ij}^\tau \cup_{a_k \in \mathcal{A}^\tau} \mathcal{S}_{a_k}^\tau \not\models \mathcal{K}_G^\tau$$

Propriété 5 (Secret futur) *Cette propriété peut être donc spécifiée ainsi :*

$$\forall \tau_i, \tau_j \in T, i < j, \{\mathcal{K}_G^{\tau_1}, \mathcal{K}_G^{\tau_2}, \ldots, \mathcal{K}_G^{\tau_i}\} \not\models \mathcal{K}_G^{\tau_j}$$

avec, $\forall k, l, \tau_k < \tau_l$ si $k < l$.

Cette définition de la propriété de secret futur peut être encore raffinée, en décrivant les attaques qui peuvent être détectées :

Propriété 6 (Protection contre une attaque par service connu) *Un protocole est dit vulnérable à ce type d'attaque si la connaissance d'un service d'une session passée permet d'avoir la vue d'un agent pour la clé d'une session future. En supposant que l'état du système est à l'instant τ, il n'y a pas d'attaque par service connu si :*

$$\forall \tau' \in T, \tau' < \tau, \forall a_i \in \mathcal{A}^\tau, S^{\tau'} \cup \mathcal{K}_I^\tau \cup_j \mathcal{K}_{Ij}^\tau \cup_{a_k \in \mathcal{A}} S_{a_k}^\tau \not\models \mathcal{A}lg_i(\mathcal{K}_i^\tau, \cup_j \mathcal{K}_{ij}^\tau, S_i^\tau)$$

Propriété 7 (Protection contre une attaque par clé connue) *Un protocole est dit vulnérable à ce type d'attaque si la connaissance d'une clé à long terme d'un agent d'une session passée permet d'obtenir la vue d'un agent pour la clé d'une session future. En supposant que l'état du système est à l'instant τ, il n'y pas d'attaque par connaissance de clé si :*

$$\forall \tau' \in T, \tau' < \tau, \forall a_i \in \mathcal{A}^\tau, \mathcal{K}^{\tau'} \cup \mathcal{K}_I^\tau \cup_j \mathcal{K}_{Ij}^\tau \cup_{a_k \in \mathcal{A}} S_{a_k}^\tau \not\models \mathcal{A}lg_i(\mathcal{K}_i^\tau, \cup_j \mathcal{K}_{ij}^\tau, S_i^\tau)$$

***b*) Secret passé**
Cette deuxième propriété garantit qu'un intrus passif connaissant un sous-ensemble de clés de groupe ordonnées $\{K_i, K_{i+1}, \ldots, K_j\}$ ne peut pas découvrir une clé de groupe précédente K_l et ceci pour tous l, i, j tels que $l < i < j$.

Propriété 8 (Secret passé) *Cette propriété s'énonce ainsi :*

$$\forall \tau_i, \tau_j, \tau_l \in T, \tau_l < \tau_i < \tau_j, \{\mathcal{K}_G^{\tau_i}, \mathcal{K}_G^{\tau_i+1}, \ldots, \mathcal{K}_G^{\tau_j}\} \not\models \mathcal{K}_G^{\tau_l}$$

4 Détection de types d'attaques

Après avoir défini notre modèle dans la section précédente, nous montrons ci-après les résultats obtenus lors de l'étude de quatre protocoles connus défectueux. Nous avons modélisé ces protocoles, leurs propriétés, puis identifié les types d'attaques auxquelles ils sont vulnérables.

Pour notre étude, nous nous sommes basé sur des scénarii d'attaques. La modélisation d'une exécution de protocole nous a permis de tester une à une les caractéristiques ainsi que les propriétés définies dans notre modèle. Il est à noter que même à partir d'une seule phase d'un scénario d'attaque nous avons pu détecter plusieurs propriétés de sécurité non vérifiées. Par exemple, c'est le cas avec le scénario d'attaque illustré dans la Figure 1, qui concerne A-GDH.2 avec quatre participants dont le troisième est malhonnête, et représenté par l'intrus a_I : il va substituer la deuxième partie du message qu'il émet, $\alpha^{r_1 r_I}$, par $\alpha^{r_1 r_2}$, ce qui va changer la seconde partie du message émis par a_4 et fausser la vue de a_2 sur la clé.

La modélisation de cette phase dans notre modèle nous donne :

$\mathbf{a_i}$	$\mathbf{S_i}$	$\mathbf{S_{a_i}}$	$\mathcal{K}_\mathbf{i}$	$\cup_\mathbf{j}\mathcal{K}_\mathbf{ij}$
$\mathbf{a_1}$	$\alpha^{r_2 r_I r_4 k_{14}}$	$\alpha^{r_1}, \alpha^{r_1 r_2}, \alpha^{r_1 r_2 r_I}, \alpha^{r_1 r_2 r_4 k_{I4}}, \alpha^{r_1 r_2 r_4 k_{24}}$	r_1	k_{14}
$\mathbf{a_2}$	$\alpha^{r_1 r_2 r_4 k_{24}}$	$\alpha^{r_2}, \alpha^{r_1 r_2}, \alpha^{r_2 r_I}, \alpha^{r_1 r_2 r_I}, \alpha^{r_1 r_2 r_4 k_{I4}}, \alpha^{r_1 r_2 r_4 k_{24}}, \alpha^{r_2 r_I r_4 k_{14}}$	r_2	k_{24}
$\mathbf{a_4}$	$\alpha^{r_1 r_2 r_I}$	$\alpha^{r_1 r_2 r_4 k_{I4}}, \alpha^{r_1 r_2 r_4 k_{24}}, \alpha^{r_2 r_I r_4 k_{14}}$	r_4	k_{14}, k_{24}, k_{I4}

À partir de ces informations, nous déduisons que la caractéristique *Déduction de la même clé de groupe* n'est pas satisfaite. En effet, les vues de clé de a_1 et de a_2 ne convergent pas vers la même clé puisque :

$$\begin{aligned}
\mathcal{A}lg_1(r_1, k_{14}, \alpha^{r_2 r_I r_4 k_{14}}) &= \alpha^{(r_2 r_I r_4 k_{14}) * r_1 / k_{14}} \models \alpha^{r_1 r_2 r_I r_4}, \\
\mathcal{A}lg_2(r_2, k_{24}, \alpha^{r_1 r_2 r_3 k_{24}}) &= \alpha^{(r_1 r_2 r_4 k_{24}) * r_2 / k_{24}} \models \alpha^{r_1 r_2 r_4 r_2}, \\
\mathcal{A}lg_4(r_4, \emptyset, \alpha^{r_1 r_2 r_I}) &= \alpha^{(r_1 r_2 r_I) * r_4} \models \alpha^{r_1 r_2 r_I r_4}
\end{aligned}$$

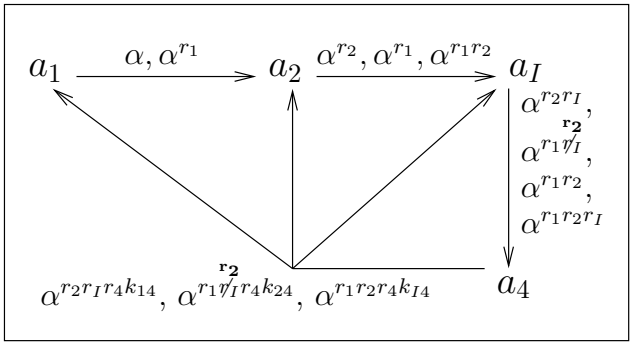

Fig. 1. Attaque dans A-GDH.2

Si nous étudions maintenant la propriété de sécurité *authentification implicite*, l'intrus (a_I) peut avoir la même vue que a_4. Grâce aux informations qu'il possède (r_I et k_{I4}), l'intrus utilise le service offert $\alpha^{r_1 r_2 r_4 k_{I4}}$ pour déduire la vue de a_4 comme suit :

$$\alpha^{(r_1 r_2 r_4 k_{I4}) * r_I / k_{I4}} \models \alpha^{r_1 r_2 r_4 r_I} = \mathcal{A}lg_4(r_4, \emptyset, \alpha^{r_1 r_2 r_I})$$

Dans la Table 1, nous précisons les protocoles que nous avons étudié, leurs références ainsi que les références des scénarii d'attaques utilisés, et les types d'attaques trouvés pour ces scénarii.

5 Travaux existants

Avec une analyse manuelle, Pereira et Quisquater [17] ont pu trouver différentes attaques intéressantes dans la suite de protocoles CLIQUES [4]. La méthode proposée consiste à convertir le problème de possession d'une information par l'intrus en un problème de résolution d'un système d'équations linéaires. Grâce à cette approche, Pereira et Quisquater ont pu trouver des faiblesses dans plusieurs protocoles de CLIQUES. Dans ces attaques, l'intrus peut avoir un comportement très libre ; par exemple, il peut utiliser des informations d'une session dans une session postérieure afin d'obtenir des secrets. D'autres travaux [18] ont eu pour but d'obtenir un résultat générique : il est impossible de désigner un AGKAP (*Authenticated Group Key Agreement Protocol*) basé sur les accords de A-GDH pour un nombre de participants dépassant quatre.

Des méthodes automatiques d'analyse de protocoles de groupe ont été intégrées dans des outils. Taghdiri et Jackson [22] ont obtenu des résultats en modélisant un arrangement de gestion de clé en multicast, proposé par Tanaka et Sato [23]. Ils ont formalisé un modèle pour ce protocole dans leur langage de spécification, Alloy [9]. Ce modèle est ensuite transmis à l'analyseur SAT de Alloy [8] pour la recherche de contre-exemples pour les propriétés de sécurité étudiées. Divers contre-exemples ont ainsi été trouvés ; le plus dangereux a été d'indiquer que les membres courants du groupe acceptent des messages valides diffusés par un membre antérieur de ce groupe. Taghdiri et Jackson ont proposé une version améliorée du protocole. Néanmoins, leur modèle formel de protocoles n'inclut pas d'intrus actif. La nouvelle version améliorée du protocole a été analysé dans [19] par CORAL. Deux attaques encore plus importantes que celles trouvées dans [22] ont été découvertes. CORAL a également été utilisé pour découvrir des attaques concernant le protocole Asokan-Ginzboorg [3] ainsi que le protocole de gestion de clé Iolus [14].

D'autres travaux se sont focalisés sur la complexité de ces problèmes, due à l'exploitation d'un espace de recherche infini. Ce problème, rencontré dans les protocoles de groupe, est dû

Protocole	**Référence protocole**	**Référence attaque**	**Caractérisique(s) non vérifiée(s)**	**Propriété(s) non vérifiée(s)**
A-GDH.2	[16]	[17]	– déduction de la même clé de groupe.	– authentification implicite, – secret de la clé, – confirmation de la clé, – intégrité.
SA-GDH.2	[4]	[17]	– déduction de la même clé de groupe.	– authentification implicite, – secret de la clé, – Secret futur (attaque par service connu).
Asokan-Ginzboorg	[3]	[20]	– déduction de la même clé de groupe.	– authentification implicite.
Bresson-Chevassut-Essiari-Pointcheval	[5]	[15]	– correspondance de services.	– Secret futur (attaque par service connu et attaque par clé connue).

Tableau 1. Synthèse des protocoles étudiés

en fait à ce que dans ce type de protocoles, même une exécution légale nécessite un nombre illimité d'étapes. Meadows [12] a étendu l'analyseur de protocoles NRL afin de traiter la suite de protocoles GDOI (*Group Domain Of Interpretation*). Bien que NPA ait été étendu pour manipuler l'exponentiation de Diffie-Hellman, il n'a pas pu retrouver [11] les attaques de Pereira et Quisquater décrites dans [17]. Une autre extension a été portée sur le langage de spécification de protocoles CAPSL [6], devenu MuCAPSL [13], pour pouvoir décrire les protocoles de groupe. Après avoir traduit le protocole GDH.2 en langage intermédiaire MuCIL, une attaque a été découverte. Néanmoins, cette analyse nécessite de fixer le nombre de participants à l'avance, ce qui peut compromettre les chances de découvrir une attaque.

6 Conclusion

Nous avons présenté une étude qui s'applique aux protocoles de groupe, et plus généralement aux protocoles contribuants. Elle permet de modéliser les protocoles eux-mêmes, mais surtout de décrire formellement leurs caractéristiques et les propriétés de sécurité qu'ils doivent considérer. Cette formalisation permet d'éviter toute ambiguité sur la définition des problèmes de sécurité posés, et donc de garantir que les recherches de failles s'effectuent sur des définitions correctes. Ainsi, les failles trouvées seront effectives.

Nous avons brièvement illustré ces travaux sur quatre protocoles connus, mais non triviaux. Le traitement d'un grand nombre de protocoles passe par l'implantation de méthodes automatiques d'analyse des propriétés définies. Pour cela, nous allons utiliser le logiciel AVISPA [1] : son puissant langage de spécification de protocoles permettra (modulo quelques extensions) de décrire sans grande difficulté les protocoles contribuants ; les propriétés à vérifier pourront également être décrites grâce au langage intégré, basé sur un fragment de la logique temporelle LTL. Les outils

de vérification intégrés dans ce logiciel devront également être étendus pour traiter ces propriétés particulières, comparées aux propriétés standards de secret et d'authentification.

Le travail présenté dans cet article pourra également être étendu pour considérer des propriétés supplémentaires, comme par exemple sur les protocoles de groupe hiérarchisés.

References

1. A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, P.-C. Héam, J. Mantovani, S. Mödersheim, D. von Oheimb, M. Rusinowitch, J. Santos Santiago, M. Turuani, L. Viganò, and L. Vigneron. The AVISPA Tool for the automated validation of internet security protocols and applications. In K. Etessami and S. Rajamani, editors, *17th International Conference on Computer Aided Verification, CAV*, volume 3576 of *Lecture Notes in Computer Science*, Edinburgh, Scotland, 2005. Springer.
2. A. Armando, D. Basin, M. Bouallagui, Y. Chevalier, L. Compagna, S. Mödersheim, M. Rusinowitch, M. Turuani, L. Vigano, and L. Vigneron. The AVISS Security Protocol Analysis Tool. In Ed Brinksma and Kim Guldstrand Larsen, editors, *14th International Conference on Computer Aided Verification, CAV*, volume 2404 of *Lecture Notes in Computer Science*, pages 349–353, Copenhagen, Denmark, 2002. Springer.
3. N. Asokan and P. Ginzboorg. Key agreement in ad hoc networks. *Computer Communications*, 23(17) :1627–1637, 2000.
4. G. Ateniese, M. Steiner, and G. Tsudik. New Multiparty Authentication Services and Key Agreement Protocols. *IEEE Journal on Selected Areas in Communications*, 18(4) :628–639, 2000.
5. E. Bresson, O. Chevassut, A. Essiari, and D. Pointcheval. Mutual Authentication and Group Key Agreement for Low-Power Mobile Devices. *Journal of Computer Communications*, 27(17) :1730–1737, 7 2004. Special Issue on Security and Performance in Wireless and Mobile Networks. Elsevier Science.
6. G. Denker and J. Millen. CAPSL Integrated Protocol Environment. In *DARPA Information and Survivability Conference and Exposition, DISCEX*, pages 207–221, Hilton Head, SC, 2000. IEEE Computer Society.
7. S. Gürgens, P. Ochsenschläger, and C. Rudolph. Authenticity and Provability - A Formal Framework. In *Int. Conference on Infrastructure Security, InfraSec*, volume 2437 of *Lectures Notes in Computer Science*, pages 227–245, Bristol, UK, 2002. Springer.
8. D. Jackson. Automating first-order relational logic. In *Proceedings of the 8th ACM SIGSOFT international symposium on Foundations of software engineering, SIGSOFT/FSE*, pages 130–139, San Diego, CA, 2000. ACM Press.
9. D. Jackson, I. Shlyakhter, and M. Sridharan. A micromodularity mechanism. In *Proceedings of the 8th European Software Engineering Conference, ESEC*, pages 62–73, Vienna, Austria, 2001.
10. Y. Kim, A. Perrig, and G. Tsudik. Tree-based group key agreement. In *ACM Transactions on Information and System Security (TISSEC)*, volume 7, pages 60–96. ACM Press, February 2004.
11. C. Meadows. Extending Formal Cryptographic Protocol Analysis Techniques for Group Protocols and Low-Level Cryptographic Primitives. In P.Degano, editor, *Proceedings of the 1st Workshop on Issues in the Theory of Security, WITS*, pages 87–92, Geneva, Switzerland, 2000.
12. C. Meadows and P. Syverson. Formalizing GDOI group key management requirements in NPATRL. In *Proceedings of the 8th ACM conference on Computer and Communications Security, CCS*, pages 235–244, Philadelphia, PA, 2001. ACM Press.
13. J. Millen and G. Denker. MuCAPSL. In *DARPA Information Survivability Conference and Exposition, DISCEX*, pages 238–249. IEEE Computer Society, 2003.
14. S. Mittra. Iolus : A Framework for Scalable Secure Multicasting. In *Proceedings of ACM SIGCOMM'97*, pages 277–288, Cannes, France, 1997.

15. J. Nam, S. Kim, and D. Won. A Weakness in the Bresson-Chevassut-Essiari-Pointcheval's Group Key Agreement Scheme for Low-Power Mobile Devices. *IEEE Communication Letters*, 9(5) :429–431, 2005.

16. O. Pereira and J.-J. Quisquater. Security Analysis of the Cliques Protocols Suites : First Results. In *Trusted Information : The New Decade Challenge, 16th Annual Working Conference on Information Security, IFIP/Sec*, IFIP Conference Proceedings, pages 151–166, Paris, France, 2001. Kluwer.

17. O. Pereira and J.-J. Quisquater. Some Attacks Upon Authenticated Group Key Agreement Protocols. *Journal of Computer Security*, 11(4) :555–580, 2003.

18. O. Pereira and J.-J. Quisquater. Generic Insecurity of Cliques-Type Authenticated Group Key Agreement Protocols. In *17th IEEE Computer Security Foundation Workshop, CSFW*, pages 16–19, Pacific Grove, CA, 2004. IEEE Computer Society.

19. G. Steel and A. Bundy. Attacking Group Multicast Key Management Protocols using CORAL. In A. Armando and L. Viganó, editors, *Proceedings of the Workshop on Automated Reasoning for Security Protocol Analysis, ARSPA*, volume 125 :1 of *Electronic Notes in Theoretical Computer Science*, pages 125–144, 2004.

20. G. Steel, A. Bundy, and M. Maidl. Attacking a Protocol for Group Key Agreement by Refuting Incorrect Inductive Conjectures. In D. Basin and M. Rusinowitch, editors, *Proceedings of 2nd Int. Joint Conference on Automated Reasoning, IJCAR*, volume 3097 of *Lecture Notes in Artificial Intelligence*, pages 137–151, Cork, Ireland, 2004. Springer.

21. M. Steiner, G. Tsudik, and M. Waidner. CLIQUES : A new approach to group key agreement. In *Proceedings of the 18th International Conference on Distributed Computing Systems, ICDCS*, pages 380–387, Amsterdam, The Netherlands, 1998. IEEE Computer Society.

22. M. Taghdiri and D. Jackson. A Lightweight Formal Analysis of a Multicast Key Management Scheme. In *Formal Techniques for Networked and Distributed Systems, FORTE*, volume 2767 of *Lecture Notes in Computer Science*, pages 240–256, Berlin, Germany, 2003. Springer.

23. S. Tanaka and F. Sato. A Key Distribution and Rekeying Framework with Totally Ordered Multicast Protocols. In *15th Int. Conference on Information Networking, ICOIN*, pages 831–838, Beppu City, Japan, 2001. IEEE Computer Society.