Modèle de contrôle d'accès pour XML :"Application à la protection des données personnelles"

Saïda Medjdoub

Université de Versailles Saint-Quentin-en-Yvelines

THÈSE

Pour obtenir le grade de

Docteur

Discipline : **Informatique**

Présentée et soutenue publiquement

Par

Saïda MEDJDOUB

Le 8 décembre 2005

Sur le sujet

Modèle de contrôle d'accès pour XML : "Application à la protection des données personnelles"

JURY

• Véronique Benzaken	Professeur à l'université de Paris sud 11	Rapporteur
• Lionel Brunie	Professeur à l'INSA de Lyon	Rapporteur
• Béatrice Finance	Maître de conférence à l'université de Versailles	Co-encadrant de thèse
• Philippe Kesmarszky	Médecin, Responsable à ALDS	Examinateur
• Philippe Pucheral	Professeur à l'université de Versailles	Directeur de thèse

Remerciements

Je tiens tout d'abord à exprimer ma gratitude à M^me^ Beatrice Finance avec qui j'ai eu l'honneur de travailler depuis mon stage de DEA. Nous avons passé ensemble des années riches en travail, elle m'a consacré de son temps pour discuter des questions scientifiques diverses. Ces discussions et ces conseils m'ont permis d'avoir un esprit de recherche et de critique objectif de mon travail.

Je remercie aussi vivement M^r^ Philippe Pucheral, mon directeur de thèse, pour sa disponibilité et sa générosité dans le travail et les conseils qu'il me prodiguait à chaque fois. Nous avons eu ensemble, pendant des réunions de travail, des discussions très enrichissantes qui m'ont orienté dans mes travaux de recherche.

Je remercie M^r^ Luc Bouganim d'avoir participé à mes réunions de travail et l'ambiance détendue qu'il apportait.

Je remercie aussi M^me^ Veronique Benzaken et M^r^ Lionel Brunie d'avoir accepté d'être rapporteurs de cette thèse ainsi que pour tout le temps qu'ils ont consacré à la lecture de ce document malgré leur emploi du temps chargé.

Je remercie chaleureusement M^r^ Philippe Kasmarszky, responsable à ALDS, pour sa participation au jury. J'en suis très honorée.

J'ai beaucoup apprécié le climat dans lequel se sont déroulées ces années de thèse que ce soit au laboratoire PRiSM ou à INRIA au sein du Projet SMIS. Je ne peux finir sans avoir une pensée à mes collègues et amis qui ont contribués à faire de ces années une expérience humaine chaleureuse. Je pense en particulier à Souheila ma meilleure copine avec laquelle j'ai partagé des bons moments et elle m'a soutenu dans des situations difficiles, Aziz, Nadjim, Khaled, Saadi, Azdine, Farouk, Midou, Yassine, Mamadou, Jalil, Daniel, Assia, Xue, sonia, Mehdi, François, Nicolas Dieu, Nicolas Anciaux, Christophe Bobineau, Thierry Delot, François Boisson, Christophe, Christophe, Christian, Aurélian, Madiagne, …

Je tiens également à remecier toutes les assistantes et les secrétaitres que ce soit au laboratoire PRiSM, Chantal Ducoin, Annick Buffer, Catherine ou à INRIA, Elisabeth Baque.

Merci enfin à ma famille de m'avoir soutenu et encouragé durant toutes ces années d'études.

Dédicaces

Je dédie ce travail à

Mon très cher père, à qui quoi que je dirai ne suffira pas pour exprimer ma gratitude et ma reconnaissance pour son soutien sans lequel je n'aurai jamais pu mener à bien ce travail et ses encouragements continus pour aller sans cesse en avant, et que je rassure que je serai à la hauteur de ses espérances ;

Ma très chère mère qui m'a toujours épaulé et motivé et que ses conseils me suivront toujours où je serai ;

Mon petit frère Riadh à qui je souhaite un brillant long parcours dans ses études, ainsi qu'à mes deux sœurs Yasmine et Basma et mes frères Abdel Ghani et Abdel Karim ;

Je n'oublirai pas de dédicacer ce travail à la mémoire de mes très chèrs grands-parents

et à tous mes amis pour leurs encouragements en particulier Souheila avec qui j'ai eu le plaisir de passer d'agréables années universitaires.

A mon papa et ma maman

Votre fille Saïda

Table des Matières

Liste des Figures

Liste des Tables

Chapitre I – Introduction

La préservation de la confidentialité est devenue un enjeu majeur pour la majorité des applications, qu'il s'agisse de gestion de données personnelles, de commerce électronique, de systèmes d'information en ligne, d'intelligence ambiante ou plus classiquement de préservation de secrets industriels ou scientifiques. La défiance envers la gestion actuelle des données confidentielles est vue, selon une étude IBM-Harris, comme le principal frein au développement de nouvelles applications sur l'Internet. La préservation de la confidentialité est une tâche gigantesque et touche de nombreuses thématiques : chiffrement des données et des communications, détection d'intrusions, contrôle de droits d'accès, anonymisation des données et des actions, fouille de données préservant la privacité, etc. Mon travail de recherche consiste à étudier les caractéristiques des modèles de contrôles d'accès, particulièrement ceux pour XML, à proposer des améliorations des modèles existants et à valider ces travaux dans un contexte applicatif très sensible, celui du dossier médical personnel.

Lorsque les données sont très sensibles, comme c'est le cas des données à caractère personnel et particulièrement des données médicales, les règles de partage et d'usage des données doivent être clairement définies et respectées. Dans ce contexte, la mise en place des politiques de contrôle d'accès est souvent régie par des textes législatifs. On peut citer par exemple la Directive 95/46/EC du parlement européen sur la protection des individus au regard des traitements effectués sur les données personnelles [44] et la directive américaine HIPAA (federal Health Insurance Portability and Accountability Act) qui porte plus particulièrement sur la protection des données de santé [67]. Dans cette thèse, les limitations des modèles de contrôle d'accès existants seront étudiés à la lumière de ces différents textes et les améliorations proposées auront pour objectif de mieux traduire les principes législatifs en règles de contrôle.

1 Problématique

XML est aujourd'hui le standard de-facto pour décrire, échanger et disséminer tout type d'informations entre différents acteurs et pour des objectifs très variés. Il a d'ailleurs été choisi comme standard d'échange d'information médicale (HL7) [69]. Garantir l'intimité, la confidentialité et la propriété intellectuelle des données XML est devenu un enjeu majeur.

C'est pourquoi de nombreux chercheurs se sont intéressés aux différentes facettes du problème. Des modèles de contrôle d'accès discrétionnaires (DAC) [8, 9, 10, 13, 37, 38, 54, 75, 78], à base de rôle (RBAC) [65, 68, 92, 118, 119], à base de mandats (MAC) [35, 108] ont été proposés dans un contexte XML. Une attention particulière a été portée sur : (1) le niveau de granularité avec lequel on définit le contrôle d'accès (de la DTD aux attributs des instances du document) [9,13, 37, 38, 54, 75, 78], (2) les algorithmes permettant d'implanter ce contrôle [35,23, 85, 96, 97, 117], (3) les canaux de communication et de distribution de l'information (dissémination sélective en mode Push ou Pull) [11, 12, 17, 90, 114, 87] et (4) la non-corruption du modèle de contrôle d'accès par des techniques de chiffrement ou d'environnements d'exécution sécurisés [19].

Tous ces travaux ont en commun de focaliser le contrôle d'accès sur les nœuds d'un document XML (attributs et éléments). Les relations de parenté et de fraternité entre les nœuds ne sont pas considérées comme des éléments de première classe dans ces modèles. Cela pose deux problèmes majeurs :

- **Divulgation de la classification** : la structure d'un document XML révèle certaines classifications (e.g. les différents services d'un hôpital dans lesquels les patients sont traités, les types d'activités ou départements de vente d'une société, une répartition socio-économique). C'est pourquoi l'appartenance d'un nœud à un sous-arbre véhicule par défaut sa classification. Bien que l'on puisse cacher l'information portée par la classe, elle n'en demeure pas moins sensible aux attaques statistiques. En effet, souvent la cardinalité d'une classe peut en révéler sa nature. De plus, cacher l'appartenance d'un élément à une classe entraîne de le faire pour chaque élément de cette classe.
- **Filiation uniforme** : la vue autorisée d'un chemin reliant un ancêtre à ses descendants est la même pour tous ces descendants. En effet, il n'est pas possible dans les modèles existants de donner deux visions différentes du même ancêtre pour deux descendants différents (e.g. un patient désire cacher le nom du service dans lequel il est traité, alors qu'un autre patient peut l'autoriser).

Ces deux problèmes vont à l'encontre de deux principes de base édictés par les directives et lois protégeant l'utilisation des données à caractère personnel. Il s'agit des principes de finalité ou "need-to-know" et de consentement. Le principe de finalité garantit que seule l'information strictement utile à l'accomplissement d'une tâche doit pouvoir être accédée. Clairement, la divulgation systématique de la classification va à l'encontre du principe de finalité, puisque cette information n'est pas utile à l'ensemble des tâches. Le principe de consentement empêche la divulgation d'information sans le consentement explicite du propriétaire de la donnée. Autrement dit, certaines règles de contrôle d'accès doivent pouvoir être personnalisées. Par exemple, dans un contexte médical, le patient garde certaines prérogatives concernant son dossier personnel. Il peut décider de la façon dont son dossier est rendu accessible dans un document XML regroupant un ensemble de dossiers et cette façon peut différer d'un patient à un autre. Cette liberté de choix contredit la filiation

uniforme. Plus que jamais, il est nécessaire de définir des modèles de contrôles d'accès qui permettent de traduire plus fidèlement les principes législatifs. Afin de faire un pas dans cette direction, nous proposons dans cette thèse un nouveau modèle de contrôle d'accès.

Parallèlement à ce travail, nous nous sommes intéressé à la protection des données par des techniques de chiffrement. Plus précisément, nous avons étudié comment traduire une politique de contrôle d'accès en règles de chiffrement d'un document XML et de distribution des clés. L'objectif est de pouvoir garantir une protection effective des données lorsque celles-ci sont gérées dans un environnement qui n'est pas totalement de confiance. Dans ce contexte, le contrôle d'accès réalisé par le serveur peut être contourné de différentes façons : attaques sur l'empreinte disque des données, attaques menées par un administrateur de données outrepassant ses privilèges, usage illicite des données par un hébergeur de données peu scrupuleux. A titre d'exemple, de tels risques existent dans la gestion des données médicales. En effet, la mise en place du dossier médical personnel (DMP) repose sur un principe d'hébergement de données. Par ailleurs, les informations téléchargées dans le cabinet d'un médecin sortent du domaine de protection du DMP et sont susceptibles d'être attaquées. Très récemment, l'attaque de l'empreinte des données de la carte vitale santé en septembre 2004 a fait la une de la presse et a remis au premier plan la nécessité de chiffrer les données médicales personnelles des patients.

Traditionnellement la sécurité des bases de données est assurée par : (1) l'identification/ authentification des utilisateurs, mise en œuvre par des techniques allant du simple login/mot-de-passe à l'utilisation de méthodes de sécurité physique (carte à puce, biométrie); (2) le chiffrement des communications garantissant la confidentialité et l'intégrité des échanges client/serveur et (3) la gestion de droits d'accès sophistiqués régis par le serveur. Cependant ces mécanismes sont inopérants contre les attaques mentionnées précédemment (attaques de l'empreinte disque, attaques de l'administrateur, usage illicite de l'hébergeur). Pour résister à ce type d'attaques menaçant la confidentialité des données, il s'avère indispensable de chiffrer tout ou partie de la base de données. Plusieurs solutions ont été proposées dans ce sens. La majorité d'entre elles stockent les données chiffrées dans la base et les déchiffrent en mémoire au moment de l'évaluation des requêtes. Le déchiffrement s'effectuant sur le serveur, cette protection reste inefficace contre un administrateur malveillant ou lorsque le serveur n'est pas de confiance (par exemple, un hébergeur de données sur l'Internet). D'autres solutions relèguent le déchiffrement au niveau des postes clients. Ces solutions sont généralement mises en œuvre dans des applications de diffusion sélective d'information (ex : échange d'information en pair-à-pair, traitements collaboratifs, gestion de droits d'accès digitaux). Dans cette thèse, nous nous sommes intéressés à ces solutions dans le cadre d'un modèle de contrôle d'accès XML.

2 Contributions

Dans cette section nous résumons les contributions de cette thèse :

- **Caractérisation des carences des modèles existants** : dans une première étape nous avons réalisé un état de l'art en deux parties. Tout d'abord, nous montrons qu'il n'existe pas une sémantique unique et non ambiguë des modèles de contrôle d'accès XML. En effet, pour une même politique de contrôle d'accès, plusieurs interprétations sont parfois possibles, certaines ne respectant pas les règles d'autorisation définies. D'autre part, les modèles existants ne permettent pas, dans un nombre significatif de situations, de traduire fidèlement les deux principes fondateurs (finalité et consentement) édictés dans les textes de loi pour la protection des données personnelles. A ce titre, notre première contribution est une étude confrontant les textes de loi avec les modèles de contrôle d'accès existant et faisant ressortir des carences bien identifiées de ces modèles. C'est au cours de cette étude que nous avons défini un exemple de référence issu du domaine médical qui a motivé nos travaux de recherche sur la mise en œuvre d'un nouveau modèle de contrôle d'accès XML.

- **Modèles de contrôle d'accès XML** : notre seconde contribution est la proposition d'un nouveau modèle de contrôle d'accès pour documents XML qui répond de façon crédible aux problèmes mentionnés ci-dessus. Ce modèle intègre les concepts de finalité et de consentement. Ce modèle repose sur des règles permettant de protéger un nœud vis-à-vis de ses ancêtres et de certains de ses frères. Le modèle proposé a un fort pouvoir d'expression tout en gardant un fort degré de concision. Les politiques de contrôle d'accès exprimables dans ce modèle sont dites sûres dans le sens où il existe un algorithme déterministe et de sémantique claire qui permet de calculer la vue autorisée à partir du document initial. Un des intérêts de l'approche est de rester compatible avec les modèles existants se focalisant sur la protection des nœuds. En effet, nous avons étendu les modèles existants afin d'intégrer la gestion des droits sur les associations entre les nœuds. Cette approche ascendante a été validée par un prototype écrit en Java. Il a été construit à partir d'un prototype existant dans le domaine public [40] qui a été étendu par la gestion des règles d'autorisation sur les associations. Nous avons réalisé des tests de performance qui montrent que le coût d'intégration des règles d'autorisation sur les associations est tout à fait comparable au coût d'intégration de règles d'autorisation sur les nœuds.

- **Partage de documents XML chiffrés** : notre troisième contribution porte sur l'étude des techniques de chiffrement d'un document permettant, à partir d'un ensemble de règles d'autorisation sur les nœuds, de définir un partage sécurisé d'un document chiffré. L'utilisateur peut déchiffrer le document s'il a en sa possession les clés associées à ses droits d'accès. L'étude des méthodes existantes de

chiffrement de documents XML a permis d'exhiber quelques contre-exemples montrant que ces méthodes peuvent être prises en défaut et générer un résultat incorrect. Nous avons proposé une autre technique de chiffrement permettant de résoudre ces problèmes. D'autre part, les méthodes de chiffrement actuelles se comportent mal en cas de mise à jour des politiques de contrôle d'accès. Nous avons identifié une piste de réflexion permettant de mieux supporter la dynamicité des politiques de contrôle d'accès en changeant radicalement la base sur laquelle repose le chiffrement du document.

3 Plan du manuscrit

Cette thèse est organisée de la façon suivante. Le deuxième chapitre présente, dans un premier temps, un état de l'art des modèles de contrôle d'accès existants. Dans un second temps, il se focalise sur les modèles de contrôle d'accès pour XML. Cette étude permet d'identifier les carences des modèles de contrôle d'accès pour XML et facilite la compréhension de nos contributions par la suite. Le troisième chapitre est consacré à la description des différentes législations relatives à la protection des données à caractère personnel. Des spécificités liées à la protection des données médicales seront également présentées. Ce chapitre introduit enfin un exemple de référence relatif à la gestion de dossiers médicaux personnels. Sur la base de cet exemple nous montrons pourquoi les modèles de contrôle d'accès XML existants ne permettent pas de traduire les principes de consentement et de finalité érigées par la loi. Le quatrième chapitre présente notre propre modèle de contrôle d'accès qui défend, plus particulièrement, l'intégration des associations entre les nœuds comme des éléments de première classe dans la définition des politiques de contrôle d'accès. Le cinquième chapitre décrit le prototype implémenté ainsi que les tests de performance réalisés. Le sixième chapitre se focalise sur l'étude des modèles de contrôle d'accès à base de chiffrement. Enfin, une conclusion générale ainsi que les perspectives des travaux de recherche terminent cette thèse.

Chapitre II – Modèles de contrôle d'accès

L'une des exigences majeures du partage des données entre plusieurs utilisateurs est la protection de ces données contre des atteintes à la confidentialité (divulgations d'information non autorisées), contre des atteintes à l'intégrité (modifications non autorisées) et contre des atteintes à la disponibilité (déni de service). Afin d'assurer cette protection, chaque accès aux données doit être contrôlé et bien évidemment tous les accès non autorisés doivent être impérativement bloqués. Cela est appelé le ***contrôle d'accès***. Le développement d'un modèle de contrôle d'accès repose sur la définition de politiques de contrôle d'accès qui déterminent ***qui a le droit d'effectuer quelle action sur quelle donnée***. Le modèle veille à ce que les données ne soient accessibles que par des utilisateurs ayant le droit d'y accéder.

Dans la littérature, un très grand nombre de modèles de contrôle d'accès ont été proposés afin de garantir la confidentialité des données. Chaque modèle proposé tente de résoudre certains des inconvénients des modèles qui le précèdent. Il existe deux grandes classes de modèles de contrôle d'accès : i) les modèles de contrôle d'accès discrétionnaires DAC (Discretionary Access Control), le créateur d'un objet se voit affecter tous les droits sur cet objet et peut transmettre tout ou partie de ses droits à d'autres utilisateurs, ii) le modèle de contrôle d'accès obligatoires MAC (Mandatory Access Control), le plus connu, fixe des niveaux hiérarchiques de sécurité aux données (public, confidentiel, secret…) et des niveaux d'habilitation (public, confidentiel, secret…) aux utilisateurs. Afin de mieux s'adapter à des organisations particulières, d'autres modèles ont été définis, en particulier, le modèle de contrôle d'accès basé sur la notion de rôle RBAC (Role Based Access Control) qui affecte des droits à des rôles; un utilisateur peut être autorisé à jouer des rôles différents au cours de sessions différentes ou dans une même session. Le modèle basé sur la notion d'équipe TMAC (Team Access control), regroupe un ensemble de rôles différents pour former une équipe de travail afin d'atteindre le même objectif. Le modèle de contrôle d'accès basé sur la notion d'organisation (Or-Bac) permet d'introduire un niveau d'abstraction permettant d'exprimer la politique de contrôle d'accès indépendamment de son implémentation. La plupart de ces modèles ont été adaptés dans le contexte des bases de données relationnelles, bases de données orientées objets et dans le contexte des documents XML. Dans cet état de l'art nous nous sommes plus particulièrement intéressés aux modèles de contrôle d'accès pour XML.

En effet, l'apparition de XML comme nouveau standard d'échange d'informations à travers le Web pose plusieurs questions sur la gestion du contrôle d'accès des documents

XML. La structure arborescente des documents XML et la nature semi-structurée des données imposent d'autres exigences au niveau du contrôle d'accès par rapport aux bases de données relationnelles. Récemment, différents modèles de contrôle d'accès pour XML ont été proposés. Certains ont adopté le modèle DAC et d'autres le modèle RBAC. Très peu de travaux se sont basés sur le modèle MAC. Parmi tous ces travaux, certains se sont intéressés à la définition de la sémantique du contrôle d'accès, d'autres se sont plutôt focalisés sur l'évaluation de performance. Des techniques de chiffrement ont été également proposées pour la gestion des droits sur des documents XML. Malgré cette diversité, tous ces modèles partagent les mêmes éléments fondateurs pour définir un modèle de contrôle d'accès pour XML. Indépendamment des modèles (DAC, MAC, RBAC) considérés, la définition d'une sémantique claire et non ambiguë reste à faire comme nous l'illustrerons dans ce chapitre.

Le premier objectif de ce chapitre est de décrire les différents modèles de contrôle d'accès cités précédemment (DAC, MAC, RBAC, TMAC, Or-BAC) et de présenter les avantages et les inconvénients de chacun d'eux. Le deuxième objectif est de présenter, dans un premier temps, les éléments fondateurs des modèles de contrôle d'accès pour XML. Dans un deuxième temps, ces éléments fondateurs sont caractérisés, nous précisons les différents choix et hypothèses de travail des travaux de recherche proposés dans la littérature. Cette synthèse ainsi rédigée nous permet de mieux comprendre les nuances parfois subtiles entre les modèles et nous permet d'appréhender au mieux les différentes approches. Pour finir, nous présentons en conclusion les carences des modèles vis-à-vis de la sémantique du contrôle d'accès et les problèmes ouverts que nous cherchons à résoudre dans cette thèse.

Avant d'entamer la description des différents modèles de contrôle d'accès, nous présentons brièvement les propriétés caractérisant la sécurité des données.

1 Sécurité des données : Définition

ITSEC (*Information Technology Security Evaluation Criteria*) [28] définit la sécurité des données comme étant la combinaison de trois propriétés :

- la *confidentialité* des données,
- l'*intégrité* des données,
- la *disponibilité* du système.

On entend par :

- **Confidentialité** : "*empêcher une divulgation non autorisée de l'information"*. En d'autres termes, cela consiste à protéger les informations sensibles contre les accès des utilisateurs non autorisés. Par exemple, dans le domaine médical "*cacher le diagnostic des patients pour les secrétaires médicales*".
- **Intégrité** : " *empêcher une modification non autorisée"*, c'est-à-dire empêcher

toute modification (suppression, ajout, mise à jour) d'une donnée par un utilisateur non légitime. Par exemple, dans le domaine médical, toute modification, malintentionnée ou pas, d'un diagnostic d'un patient peut mettre sa santé, voire sa vie, en danger.

- **Disponibilité** : "*empêcher un déni non autorisé d'accès à l'information ou à des ressources"*. En d'autres termes, garantir de rendre une donnée accessible lorsqu'un utilisateur autorisé en a besoin.

Dans nos travaux de recherche nous nous sommes focalisés plus précisément sur la propriété de ***confidentialité***.

2 Modèles de contrôle d'accès

Avant de présenter ces différents modèles, nous allons définir, dans cette section, ce que nous appelons une politique de contrôle d'accès.

2.1 Définition d'une politique de contrôle d'accès

Les politiques de contrôle d'accès sont définies comme étant des directives (règles) de haut niveau [104, 105] qui spécifient ***qui*** a la permission d'exercer ***quoi*** sur ***quelle*** donnée. A partir de cette définition nous dégageons trois concepts fondamentaux d'une politique de contrôle d'accès qui sont :

- *Sujet* : entité active qui accède aux données du système. Le sujet peut être un utilisateur, une application, une adresse IP …
- *Objet* : entité passive qui représente les données à protéger. L'objet peut être, par exemple, un fichier, une table relationnelle, une classe …
- *Action* : représente l'action à traiter par le sujet sur l'objet. L'action peut être lire, écrire, exécuter …

2.2 Modèle de contrôle d'accès discrétionnaire (DAC)

TCSEC (Trusted Computer System Evaluation Criteria) définit le contrôle d'accès discrétionnaire comme : "*un moyen de restriction d'accès aux objets basé sur l'identité des sujets et/ou groupes auquel ils appartiennent. Les contrôles sont discrétionnaires dans le sens où le sujet est capable de transférer les permissions d'accès à d'autres sujets*" [88].

Le modèle de contrôle d'accès discrétionnaire représente les politiques de contrôle d'accès sous forme d'un triplet <*utilisateur, objet, action*> qui exprime que l'*utilisateur* peut effectuer une certaine opération identifiée par l'*action* (e.g, lire) sur l'*objet* spécifié. Le triplet est appelé une ***règle d'autorisation***.

Dans certains systèmes, l'accès se fait uniquement en spécifiant d'une façon explicite un ensemble de règles d'autorisation. En d'autres termes, si aucune règle d'autorisation n'est définie pour un utilisateur, l'accès lui sera interdit. Ce type de politique est appelé une politique *fermée* [105]. Inversement, dans les systèmes adoptant une politique *ouverte* l'accès aux objets sera interdit, uniquement, en présence des règles d'autorisation (négative), c'est-à-dire, l'utilisateur a le droit d'accéder à tous les objets du système sauf si une règle d'autorisation a été définie explicitement lui interdisant l'accès aux objets.

Les travaux récents combinent les règles d'autorisation positives et négatives afin de supporter des exceptions. Par exemple, nous souhaitons accorder une autorisation à tout membre d'un groupe composé de mille utilisateurs, à l'exception d'un seul membre spécifique Bob. Pour répondre à cette politique de contrôle d'accès, dans le cas d'une politique fermée, nous sommes obligés de définir une règle d'autorisation positive pour chaque membre du groupe à l'exception de Bob. Alors qu'en combinant les règles positives et les règles négatives, nous pouvons répondre à cette politique uniquement en définissant deux règles d'autorisation. Une règle d'autorisation positive pour tout le groupe et une autre règle d'autorisation négative pour Bob. La combinaison des règles négatives et des règles positives rend le contrôle d'accès plus flexible. Mais cela pose quelques problèmes de gestion de conflits que nous étudierons par la suite dans le contexte bien précis de XML.

La description des règles d'autorisation est basée sur le modèle de *matrice d'accès*. Cette notion de matrice d'accès a été introduite initialement dès 1971 par Lampson [77]. Ensuite, Harison, Ruzzo et Ullman (HRU) ont développé une version plus générale du modèle en 1976 [63]. Dans ce dernier, l'état du système est défini par un triplet (*S*, *O*, *M*) où *S* représente l'ensemble des sujets (e.g, utilisateurs, processus…) pouvant exercer un ensemble d'actions. *O* représente l'ensemble des objets (e.g, fichier, table, classe, programme…) sur lesquels les actions peuvent être effectuées. Enfin, *M* représente la matrice d'accès, où les lignes correspondent aux sujets et les colonnes correspondent aux objets. Une entrée *M*[S, O] caractérise les actions *a* de *s* sur *o* (lecture, écriture,…). Un exemple de matrice est présenté dans la Figure 1.

	$Fichier_1$	$Fichier_2$	$Fichier_3$	programme
Ann	propriétaire	lire écrire		exécuter
Bob	lire		lire écrire	
Carl	écrire			exécuter lire

Figure 1: Exemple de matrice d'accès

Bien que la matrice offre une bonne représentation conceptuelle des autorisations, elle est inappropriée pour l'implémentation. En effet, cette matrice peut être immense et son

stockage direct comme un tableau à deux dimensions peut consommer beaucoup d'espace mémoire [41, 104, 105]. Pour cette raison, il existe en pratique trois approches pour implémenter la matrice :

1. ***Table d'autorisation*** : les entrées vides de la matrice ne sont pas reportées dans la table. La table est composée de trois colonnes qui correspondent aux sujets, aux actions et aux objets. Chaque n-uplet de la table correspond à une autorisation.
2. ***ACL* (*Access Control List*)** : la matrice est stockée par colonne. Chaque objet est associé à une liste indiquant pour chaque utilisateur les actions pouvant être exercées par ce dernier sur cet objet.
3. ***Capacité* (*capability*)** : la matrice est stockée par ligne. Chaque utilisateur a une liste, appelée une liste de capacité**,** indiquant pour chaque objet les actions que l'utilisateur est autorisé à effectuer sur cet objet.

2.2.1 Les limites du modèle de contrôle d'accès discrétionnaire

Le modèle de contrôle d'accès discrétionnaire limite l'accès aux objets uniquement en se basant sur l'identité de l'utilisateur. Pour cela, le modele DAC est appelé également, IBAC (Identity Based Access Control) [33]. Ce principe de base rend le contrôle d'accès vulnérable aux chevaux de Troie [88, 104, 105]. Afin de comprendre comment le cheval de Troie peut amener à une fuite d'information vers des utilisateurs non autorisés, nous prenons un exemple pour illustrer le problème (Figure 2).

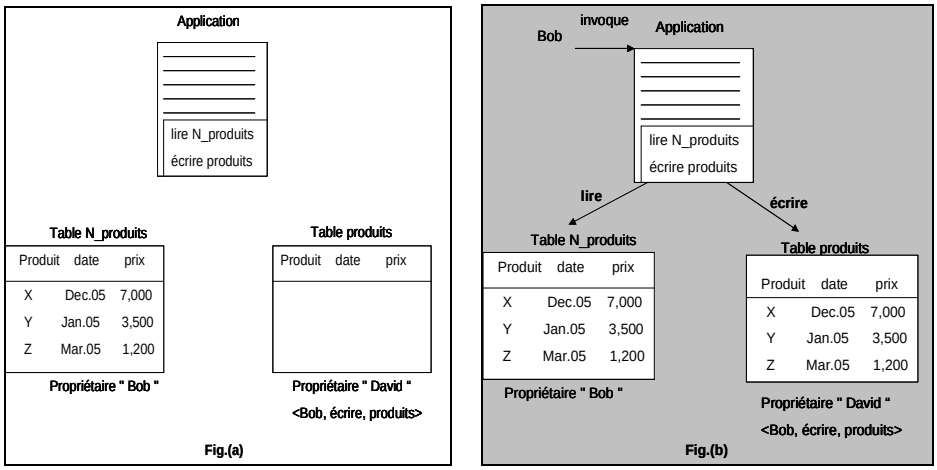


Figure 2 : Exemple de chevaux de Troie

Supposons dans une organisation, Bob, directeur, crée un fichier *nouveaux_produit* (*N-produits*) contenant des informations très sensibles sur les nouveaux produits. Ces informations sensibles, d'après la politique de l'organisation, ne devraient être accessibles que par Bob. Supposons maintenant qu'un utilisateur malveillant *David*, un adjoint de Bob, veuille récupérer cette information sensible pour la vendre à une organisation concurrente. Pour cela, David crée un fichier *Produits* et donne l'autorisation à Bob d'écrire dans ce

fichier. David, ensuite, introduit deux opérations cachées dans l'application utilisée par Bob. Ces opérations sont *lire* dans le fichier *N-produits* et *écrire* dans le fichier *produits*. Une fois que Bob exécute l'application, les opérations lire et écrire vont être permises. Puisque, l'utilisateur malveillant David est le propriétaire du fichier *produit* il pourra accéder à ce fichier et récupérer les informations désirées.

Nous constatons que malgré le fait que nous fassions confiance aux utilisateurs pour qu'ils obéissent aux politiques de l'organisation nous ne pouvons pas faire confiance aux processus qui s'exécutent pour leur compte d'où la nécessité de distinguer entre les utilisateurs[1] et les processus qui s'exécutent pour leurs comptes (sujets).

Dans la section suivante, nous montrons comment les modèles MAC (plus précisément le modèle multi-niveaux) font la distinction entre sujets et objets pour résoudre les problèmes des chevaux de Troie et de fuite d'information.

2.3 Modèle de contrôle d'accès obligatoire (MAC)

Afin de remédier au problème de fuites d'information des modèles de contrôle d'accès discrétionnaires, les modèles obligatoires (Mandatory Access Control) fixent des règles incontournables destinées à forcer le respect des exigences de contrôle d'accès. Ainsi, le modèle multi-niveaux affecte aux sujets et aux objets des niveaux de sécurité non modifiables par les utilisateurs et, par conséquent, limite leurs pouvoirs dans la gestion des accès à leurs données. Le premier modèle, appelé modèle de Bell et LaPudula [14], a été développé pour le département de la défense américaine et vise, plus particulièrement, à assurer la confidentialité. Le deuxième modèle, appelé modèle de Biba [16], s'est intéressé à l'intégrité. D'autres modèles obligatoires, moins formalisés, ont été développés pour les systèmes commerciaux (Modèle de Clark et Wilson) [34] et pour les institutions financières britanniques (Modèle de muraille de Chine)[42].

Dans cette section, nous nous intéressons plus particulièrement au modèle multi-niveaux qui vise à assurer la confidentialité. Ce dernier est basé sur la classification des sujets et des objets. Le principe consiste à attribuer une classe d'accès à chaque sujet et à chaque objet. Généralement, une classe d'accès est constituée de deux composants : un *niveau de sécurité* et un ensemble de *catégories*. Le niveau de sécurité est un élément d'un ensemble hiérarchique ordonné, tel que TS (Top Secret) > S (Secret) > C (Confidential)> U (Unclassified)>. L'ensemble des catégories est un sous-ensemble d'un ensemble non ordonné, dont les éléments représentent soit une compétence, une région, un département. Par exemple, pour les systèmes militaires nous pouvons avons les catégories nucléaire, défense et pour les systèmes commerciaux nous pouvons avoir les catégories administration et recherche. Le niveau de sécurité associé à un objet reflète son degré de sensibilité. Le

[1] Entité passive qui peut se connecter au système et pour laquelle les règles d'autorisation peuvent être spécifiées.

niveau de sécurité associé à un sujet (appelé en anglais "*clearance*") reflète son niveau de confiance.

Le modèle de Bell-LaPadula est basé sur la notion de treillis où chaque classe d'accès est caractérisée par deux attributs *n* = (cl, C) :

- cl : représente une de ces classifications : Top Secret (TS), Secret (S), Confidential, Unclassified.
- C : représente les catégories. Par exemple, {nucléaire, défense}.

Les classes d'accès constituent un treillis partialement ordonnée par une relation de dominance notée "≤" qui est définie par :

Soit n = (cl, C) et n' = (cl', C) , n ≤ n'(n' domine n) si et seulement si cl ≤ cl' et C ⊆ C'.

La Figure 3 présente un exemple de treillis avec deux niveaux de sécurité (TS ≥ S) et l'ensemble des catégories {nucléaire, défense}.

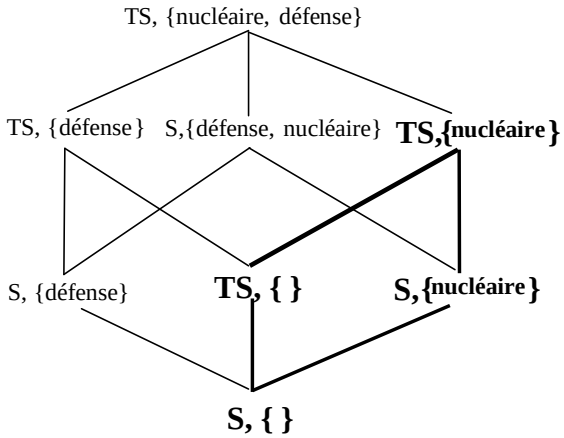


Figure 3 : Exemple de treillis de sécurité

Comme nous l'avons dit précédemment, le modèle de contrôle d'accès discrétionnaire ne distingue pas les utilisateurs des sujets. Cette non distinction amène à des problèmes de fuite d'information. Le modèle de contrôle d'accès obligatoire remédie à ce problème en faisant une différence entre les utilisateurs et les sujets. Les utilisateurs sont des entités passives qui peuvent se connecter au système alors que les sujets sont des processus qui s'exécutent pour le compte des utilisateurs. Un utilisateur se connectant au système avec une classe d'accès donnée génère un sujet de cette classe d'accès. En fonction du niveau de sécurité (confiance) attribué à l'utilisateur, différents sujets peuvent être générés par cet utilisateur. Par exemple, si le niveau de sécurité d'un utilisateur est (Secret, ∅) alors il pourra se connecter au système comme un sujet (Secret, ∅), (Confidential, ∅) ou (Unclassified, ∅).

Dans ce modèle, afin de respecter la confidentialité des informations et d'éviter les fuites d'information qui peuvent survenir, la décision d'accès aux objets doit obligatoirement respecter les deux principes fondamentaux :

- ***No read up*** : un sujet est autorisé à lire un objet donné uniquement si sa classe d'accès domine la classe d'accès de l'objet.
- ***No write down*** : un sujet est autorisé à écrire dans un objet donné uniquement si la classe d'accès de l'objet domine sa classe d'accès.

Maintenant, nous montrons comment le respect de ces deux principes évite la fuite d'information lié à l'exemple des chevaux de Troie précédemment cité. D'après la politique de contrôle d'accès, le fichier *N_produits* est un fichier contenant des informations sensibles et il ne peut être accessible que par l'utilisateur Bob. Une classification possible pour répondre à cette politique est : *Secret* pour *Bob* et le fichier *N_produits*, *Unclassified* pour David et le fichier *produits*. Si *Bob* se connecte au système comme un sujet Secret, l'application s'exécute avec une classe d'accès Secret, l'opération d'écriture dans le fichier *Produits* avec le niveau de sécurité Unclassified sera bloquée (principe No write down). Si Bob invoque l'application avec un niveau de sécurité Unclassified, l'opération de lecture de fichier *N_produits* sera bloquée (principe No read up). Par conséquent, en respectant les deux principes, le cheval de Troie ne peut se terminer avec succès.

Bien que le modèle MAC résolve le problème de fuite d'information des modèles DAC, il est quand même un modèle très rigide. Il ne permet pas de gérer les exceptions entre les différents niveaux de sécurité. Par exemple, un utilisateur de niveau de sécurité secret ne peut accéder, pour des raisons exceptionnelles, à la donnée de niveau de sécurité top secret.

2.4 Modèle de contrôle d'accès à base de rôle (RBAC)

La motivation principale autour du contrôle d'accès à base de rôle RBAC (Role Based Access Control) est de faciliter l'administration de la politique de contrôle d'accès et de proposer un modèle de contrôle d'accès qui reflète la structure organisationnelle de l'entreprise [48, 49, 106]. Le cœur de RBAC est le *rôle*. Ce dernier représente d'une façon abstraite une fonction particulière dans une organisation (par exemple, médecin, infirmière, statisticien…). Le *rôle* est une entité intermédiaire entre les permissions d'accès, appelées aussi *privilège* ou *droit d'accès*, et les utilisateurs. Il regroupe un ensemble de privilèges qui va être ensuite attribué aux utilisateurs en fonction de leurs positions organisationnelles. Par conséquent, contrairement au contrôle d'accès discrétionnaire, l'utilisateur ne reçoit pas directement ses permissions d'accès, mais les reçoit via des rôles. La Figure 4 montre l'attribution des opérations aux utilisateurs à travers les rôles.

Figure 4 : Attribution des permissions en RBAC

Un rôle peut avoir plusieurs permissions et une permission peut être associée à plusieurs rôles. Un utilisateur peut jouer plusieurs rôles et un rôle peut être attribué à plusieurs utilisateurs. A titre d'exemple, dans le domaine médical, le médecin Durant est à la fois chirurgien et directeur de l'hôpital. Dans ce cas, il pourra accéder aux dossiers médicaux des patients en jouant le rôle chirurgien et aux dossiers administratifs de l'hôpital en jouant le rôle directeur de l'hôpital. Un utilisateur établit une session durant laquelle il active un sous-ensemble de ses rôles. Dans une session, plusieurs rôles peuvent être activés (à la discrétion de l'utilisateur) et chaque session est associée à un seul utilisateur.

Des variantes de RBAC ont été proposées, appelées dans la littérature famille de RBAC[106]. En réalité, il existe quatre types de modèles RBAC qui sont représentés dans la Figure 5. $RBAC_0$ représente le modèle de base qui contient les éléments minimaux d'un modèle de contrôle d'accès à base de rôle (utilisateurs, rôles, permissions, sessions). Des extensions à $RBAC_0$ ont été faites par l'ajout du concept de hiérarchie de rôles (un rôle peut hériter d'un autre rôle). A titre d'exemple, un médecin peut hériter de toutes les permissions attribuées à un infirmier. Cette extension a donné naissance à un type de RBAC appelé $RBAC_1$. $RBAC_2$ ajoute un ensemble de contraintes. Ces contraintes incluent des règles de cardinalité et d'exclusion mutuelle qui peuvent être appliquées, par exemple, sur des rôles. A titre d'exemple, un utilisateur ayant deux rôles ne peut pas les activer en même temps. D'autres contraintes peuvent être des contraintes de temps et de lieu. Par exemple, l'accès à une base de données le soir n'est pas autorisé. Le dernier modèle $RBAC_3$ est la combinaison des deux modèles précédents ($RBAC_1$ et $RBAC_2$).

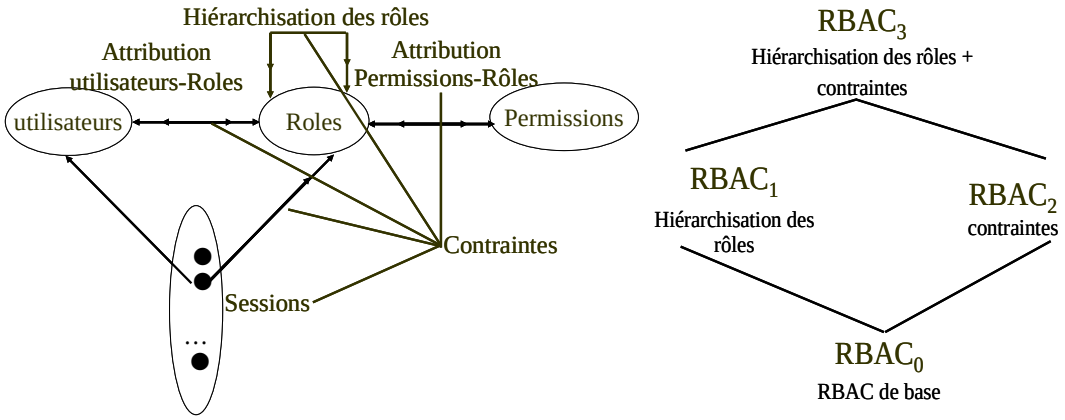


Figure 5 : Présentation de $RBAC_3$ (figure gauche) et famille de RBAC (figure droite)

Comme nous l'avons mentionné plus haut, l'objectif principal du modèle RBAC est de faciliter l'administration des politiques de contrôle d'accès. La gestion d'un nouvel utilisateur est immédiate. Il suffit d'affecter le(s) rôle(s) que l'utilisateur assurera au sein de l'organisation. Sa ré-affectation à d'autres fonctions est très simple, il suffit de désactiver son ancien rôle et de l'affecter au(x) nouveau(x) rôle(s). De la même façon, nous pouvons également faire évoluer les tâches des rôles en mettant à jour les privilèges. L'organisation des rôles en hiérarchie facilite encore plus l'administration des contrôles d'accès. Un autre avantage de RBAC est que, dans les systèmes distribués, la tâche de l'administrateur peut être divisée entre le domaine de protection central et le domaine de protection local. Les politiques de contrôle d'accès central peuvent être définies au niveau de l'entreprise alors que l'activation de ces politiques peut être faite au niveau des unités organisationnelles. Par exemple, dans un système de soins de santé distribué, les opérations associées aux fournisseurs de soins peuvent être spécifiées d'une façon centralisée et concernent tous les hôpitaux et cliniques. Par contre, l'affectation/révocation des utilisateurs aux rôles peut être spécifiée par des administrateurs au niveau des sites locaux.

L'inconvénient de RBAC réside dans la difficulté de gérer des règles dépendant du contexte, par exemple, de types « seuls les médecins traitants peuvent accéder aux informations médicales du dossier d'un patient » ou « les étudiants ont le droit d'accéder uniquement à leurs données personnelles »[118]. Une des solutions envisagées est de créer, par exemple, pour chaque étudiant un rôle privé. Théoriquement c'est une solution mais en pratique cela n'est pas faisable car il existe un très grand nombre d'étudiants et cela fait perdre à RBAC sa simplicité d'administration.

2.5 Modèle de contrôle d'accès à base d'équipe

Le modèle de contrôle d'accès à base d'équipe (TMAC : *Team-based Access Control*) a été initialement proposé par K.Thomas [110]. L'objectif est de fournir un contrôle d'accès dans le cas d'un travail collaboratif tout en exploitant la flexibilité du modèle de contrôle d'accès à base de rôle (RBAC). L'entité de base de TMAC, *équipe* ou "*team*", est une abstraction qui encapsule un ensemble d'utilisateurs, qui ont des rôles différents et qui collaborent dans le but d'accomplir une tâche commune ou d'atteindre un objectif commun. Les utilisateurs affectés à l'équipe devront bénéficier d'un accès à toutes les ressources de l'équipe. Cependant, les permissions exactes de chaque utilisateur sont déterminées par le rôle qu'il joue et l'activité courante de l'équipe. Le travail de K.Thomas [110] est considéré comme un point de départ pour la définition d'un contrôle d'accès à base d'équipe. Une extension à TMAC a été proposée par Georgiadis [59] en intégrant la notion de contexte qui a donné naissance à *C-TMAC* (Context-based Team Access Control). Le contexte inclut des informations concernant les objets demandés pour une tâche spécifique (e.g. l'identifiant d'un patient traité), ainsi que les informations contextuelles tel que l'intervalle de temps et le lieu. Dans ce qui suit, nous allons présenter les concepts du modèle C-TMAC. La Figure 6 regroupe cet ensemble de concepts.

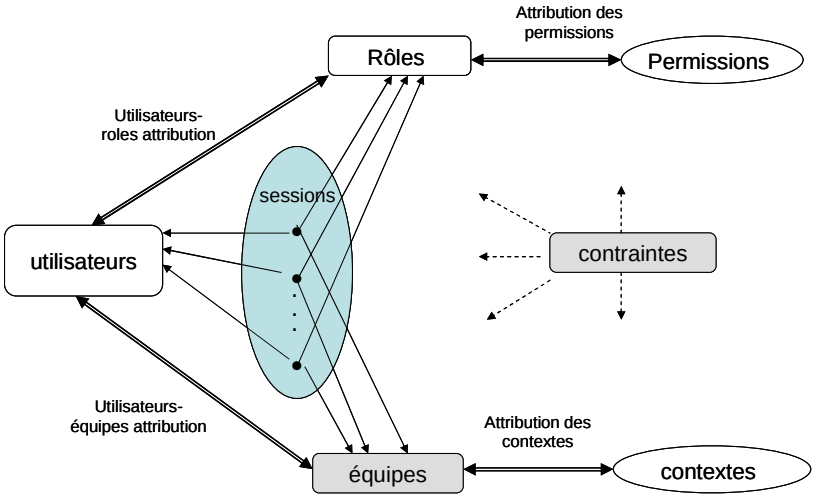


Figure 6 : Approche C-TMAC

L'élément *contexte* inclut des informations concernant les objets demandés pour une activité donnée (e.g., identifiant d'un patient donné), ainsi que les informations de lieu et de temps. L'entité *équipe* est utilisée pour représenter un groupe d'utilisateurs ayant des rôles spécifiques pour accomplir une tâche dans un contexte particulier. Cependant, le concept d'équipe est représenté comme une entité intermédiaire entre l'utilisateur et le contexte (d'une façon similaire au rôle qui est utilisé comme entité intermédiaire entre l'utilisateur et les permissions). Au cours d'une session, un utilisateur peut participer à plusieurs équipes. L'activation des équipes dans une session se fait à la discrétion des utilisateurs. L'ensemble des permissions disponibles à l'équipe est obtenu par la combinaison de toutes les permissions des rôles (activés) participant dans cette équipe.

Enfin, une équipe peut avoir plusieurs contextes et le même contexte peut être attribué à plusieurs équipes. De la même manière, un utilisateur peut être membre de plusieurs équipes et une équipe peut avoir plusieurs utilisateurs. Des contraintes existent lors de l'attribution des utilisateurs aux équipes. Par exemple, un utilisateur ayant deux rôles différents ne peut les activer dans une même équipe. E.g., un utilisateur qui a été affecté aux rôles médecin et directeur ne peut pas participer dans une équipe de soins comme directeur.

Bien que les modèles TMAC et C-TMAC offrent un moyen de contrôle d'accès dans un cadre de travail collaboratif tout en bénéficiant de la flexibilité de RBAC, ces modèles souffrent d'un inconvénient majeur concernant la gestion des droits d'accès. Cela est lié, plus précisément, à l'ensemble des permissions offertes à l'équipe qui peut violer le *principe du moindre privilège*. D'après les modèles, un utilisateur rejoignant une équipe renforce les permissions de cette équipe en ajoutant les siens. Néanmoins, dans le secteur médical, bien que les professionnels de santé appartiennent à la même équipe dans le même hôpital, ils n'ont pas forcement les mêmes droits sur les parties du dossier médical du patient. Logiquement, il est évident que les permissions finales du médecin doivent être différentes de celle de l'infirmière même s'ils appartiennent à la même équipe.

2.6 Modèle de contrôle d'accès Or-BAC

Le modèle de contrôle d'accès Or-BAC (Organization Based Access Control) [1] vise à résoudre certains problèmes rencontrés par les modèles de contrôle d'accès existants et à établir une politique de contrôle d'accès plus abstraite. Il s'intéresse, non seulement aux permissions, mais aussi aux interdictions, obligations et recommandations. Or-BAC introduit, en plus du concept de rôle pour structurer les sujets, des concepts (notions) pour structurer les objets et les actions. Comme son nom l'indique, l'entité centrale du modèle est l'*Organisation*. Une organisation peut être un groupe structuré des sujets jouant certains rôles, où des entités comme hôpital, clinique médicale, service d'urgence… Le fait d'introduire ce concept *organisation* comme un élément de base dans le modèle de contrôle d'accès résout les problèmes de TMAC et RBAC. Ces derniers définissent des relations binaires entre l'utilisateur et rôle, et entre utilisateur et équipe. Ce qui veut dire que l'utilisateur ayant plusieurs rôles peut activer soit tous les rôles soit un sous-ensemble de ses rôles, dans n'importe quelle équipe à laquelle il participe. Dans la pratique, même si un utilisateur possède plusieurs rôles, il n'a pas forcément le droit de les jouer dans toutes les équipes auxquelles il appartient [1, 2, 4, 120, 29].

La Figure 7 montre les différents éléments du modèle Or-BAC [29]. Les rectangles représentent les entités et les ovales représentent les relations. La figure montre les deux niveaux de la politique (le niveau abstrait et le niveau concret), ainsi que les différentes relations existantes entre les entités de ces deux niveaux.

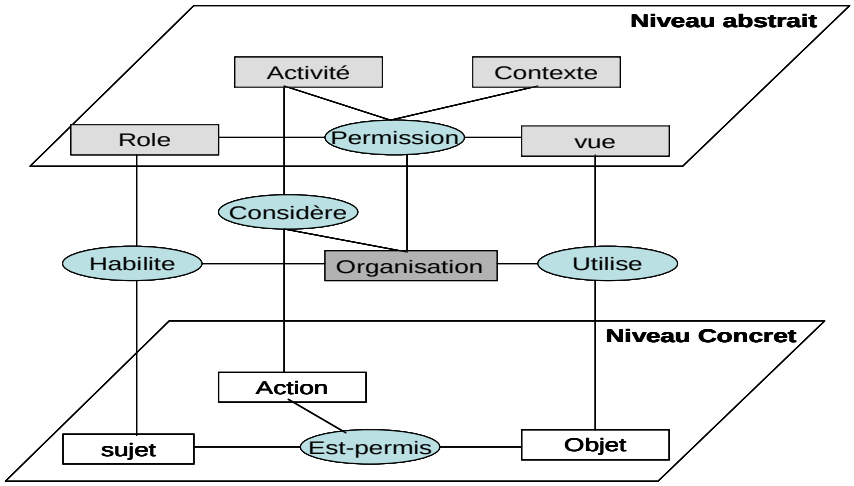


Figure 7 : Modèle Or-BAC

Dans ce qui suit nous décrivons brièvement les relations existantes entre les éléments du niveau concret et les éléments de niveau abstrait d'Or-BAC.

- *Les sujets et les rôles* : le sujet représente soit une entité active, c'est-à-dire un utilisateur (Marie, Pierre, …), soit une organisation (service d'urgence de l'hôpital

de Purpan). Dans Or-BAC, l'entité rôle structure le lien entre les sujets et les organisations. Les rôles *médecin* ou *infirmière* sont joués par des utilisateurs alors que les rôles "s*ervice des urgences*" ou "*unité des soins intensifs*" sont joués par des organisations. Comme des sujets jouent des rôles dans des organisations, une relation entre ces entités, appelée *Habilite*, est définie comme suit :

- Si *org* est une organisation, *s* est un sujet et *r* est un rôle alors *Habilite(org, s, r)* signifie que *org* habilite le sujet *s* à jouer le rôle *r*.
- *Exemple* :
  - *Habilite(Purpan, jean, cardiologue)* : "l'hôpital Purpan habilite Jean dans le rôle cardiologue".
  - *Habilite(Rangueil, ICU31, unité_des_soins_intensifs) :* "l'hôpital Rangueil habilite l'unité ICU31 dans le rôle d'unité des soins intensifs".

– *Les objets et les vues* : les objets représentent des entités passives comme par exemple les fichiers, les dossiers administratifs, les dossiers médicaux… L'entité *vue* permet de structurer les objets (notion similaire aux rôles par rapport aux sujets) et elle caractérise la façon dont les objets sont utilisés dans l'organisation. Par conséquent, une relation, appelée *Utilise*, a été définie pour relier les organisations, les objets et les vues :

- Si *org* est une organisation, *o* est un objet, et *v* est une vue alors *Utilise(org, o, v)* signifie que *org* utilise l'objet *o* dans la vue *v*.
- *Exemple* : les organisations peuvent donner des définitions différentes à la même vue. La vue dossier médical peut être définie dans l'hôpital *Purpan* comme un ensemble de documents Word. Alors que dans l'hôpital Rangueil comme un ensemble de documents Latex.
  - *Utilise(Purpan, F31.doc, dossier_médical)* "l'hôpital *Purpan* utilise *F31.doc* comme un *dossier médical* ".
  - *Utilise*(*Rangueil*, *F32.tex*, *dossier_médical*) " l'hôpital *Rangueil* utilise *F32.tex* comme un *dossier médical* ".

– *Les actions et les activité*s : les actions représentent les opérations qui peuvent être effectuées par les sujets sur les objets. Dans ce modèle, l'entité *Action* représente les actions informatiques "lire", "écrire"…etc. Les *Activités* correspondent aux actions qui ont le même objectif. Par exemple, *consulter*, *modifier*…etc. Là encore, l'objectif est de permettre à des organisations de structurer différemment les mêmes activités. D'après la Figure 7 la relation *Considère* est utilisée pour associer les trois

entités *Organisation*, *Action* et *Activité* :

- Si *org* est une organisation, α est une action et *a* représente une activité, alors *Considère(org, α, a)* signifie que l'organisation *org* considère l'action α comme faisant partie de l'activité *a*.
- *Exemple* : l'activité "*consultation*" peut correspondre dans l'organisation hôpital *Purpan* à l'action "*lire*" un fichier mais elle peut correspondre à l'action *select* sur la base de données dans l'hôpital Rangueil. Par conséquent, l'objectif est de pouvoir caractériser
  - *Considère(Purpan, lire, consultation)* : l'hôpital *Purpan* considère *lire* comme une consultation.
  - *Considère(Rangueil, select, consultation)* : l'hôpital Rangueil considère *select* comme une consultation.

– *Les contextes :* cette entité joue un rôle important dans la définition des politiques de contrôle d'accès spécifiques. Elle permet d'exprimer des politiques dépendantes du contexte (circonstances concrètes) dans lesquelles les organisations accordent les permissions de réaliser des activités sur des vues. Par exemple, dans le domaine médical, le médecin n'a le droit d'accéder qu'aux dossiers médicaux des patients qu'il traite. Par contre, dans des situations *d'urgence* le médecin peut accéder au dossier du patient en question. Or-BAC définit différents types de contextes [30] contexte temporel, spatial, contexte provisionnel… Une nouvelle relation "*définit*" a été introduite pour relier les entités *Organisation*, *Sujet*, *Objet*, *Action*, et *Contexte* telle que :

- Si *org* est une organisation, *s* est un sujet, *o* est un objet, α est une action et *c* est un contexte alors *Définit(org, s, α , o, c)* signifie qu'au sein de l'organisation *org*, le contexte *c* est vrai entre le sujet *s*, l'objet *o* et l'action α.
- *Exemple* :
  - *Définit(Purpan, Jean, lire, F31.doc, urgence)* : dans le contexte d'urgence, Jean peut consulter le dossier du patient F31.doc.
  - *Définit(Rangueil, Marie, lire, F32.tex, médecin_traitant)* : Marie peut consulter le fichier du patient F32.tex si seulement si elle est le médecin traitant du patient dont le dossier est *F32.tex*.

Nous remarquons que, les entités rôle, vue et activité sont respectivement des abstractions de sujet, objet et action. Cela permet au modèle Or-BAC de définir des politiques de contrôle d'accès abstraites (rôle, activité, vue) indépendantes des choix

d'implémentation (sujet, action, objet). L'entité permission (voir la Figure 7) est une entité abstraite entre un rôle, une activité, une vue et un contexte. L'organisation dans laquelle une permission est valide est aussi présente dans la relation : *Permission (Organisation, Rôle, Activité, Vue, Contexte)*[2]. Cette relation signifie que l'organisation donne la permission à un rôle de réaliser une activité sur une vue dans un certain contexte. Par contre, le contrôle d'accès au bas niveau doit permettre de décrire les actions concrètes que réalisent les sujets sur les objets. D'où l'entité *Est_permis* est introduite dans le modèle. Les permissions concrètes sont dérivées des permissions abstraites par la règle suivante :

Si *permission (organisation, rôle, activité, vue, contexte)* et
Habilite(organisation, sujet, rôle) et
Considère(organisation, action, activité) et
Utilise(organisation, objet, vue) et
Définit(organisation, sujet, action , objet, contexte)
Alors *Est_permis (sujet, action, objet)*

Comme nous l'avons évoqué précédemment, le modèle définit également des interdictions, des obligations et des recommandations. La combinaison entre ces possibilités amène à des situations conflictuelles. La résolution de conflits est basée sur l'attribution de niveaux de priorité entre les règles de contrôle d'accès [29, 31]. En plus, le modèle Or-BAC offre la possibilité de hiérarchiser des rôles, des activités, des vues, et des organisations [32]. La formalisation de Or-BAC est fondée sur la logique de premier ordre [1]. Un autre formalisme a été proposé qui est fondé sur la logique déontologie [4].

Dans cette section, nous avons présenté les modèles de contrôle d'accès existants dans la littérature. Ces modèles ont été conçus de façon complètement indépendante des modèles de données. Ils fixent la sémantique du contrôle au niveau abstrait, étudient les facilités d'administration d'un ensemble de politiques de contrôle d'accès. Dans la littérature, ces modèles de contrôle d'accès ont été mis en œuvre dans des contextes bien différents que cela soit celui des bases de données relationnelles, des bases de données orientées objets [41, 47, 62] et de la protection des documents XML. Quel que soit le modèle considéré, ce qui les distingue véritablement c'est la mise en œuvre des mécanismes de base indispensables. Ces mécanismes sont parfois bien différents d'un modèle de données à un autre. Par exemple, on s'appuie dans les bases de données relationnelles sur le principe de vue SQL [61], par contre dans les bases de données semi-structurées (XML) la presque totalité des modèles de contrôle d'accès reposent sur un ensemble de règles d'autorisation que nous allons décrire dans la section suivante.

3 Modèles de contrôle d'accès pour XML

[2] Le même raisonnement est fait pour l'obligation et l'interdiction.

L'apparition de XML comme nouveau standard d'échange d'information à travers le Web a relancé les travaux de recherche sur les modèles de contrôle d'accès. En effet, la structure arborescente des documents XML et la nature semi-structurée des données imposent d'autres exigences au niveau du contrôle d'accès. Les travaux proposés dans la littérature s'appuient principalement sur les modèles DAC [8, 9, 10, 13, 37, 38, 54, 78], RBAC [65, 68, 92, 118, 119] et MAC [35, 108]. Des extensions au modèle DAC ont été proposées dans [75], l'idée consiste à intégrer des autorisations provisionnelles dans le modèle afin que le système puisse contrôler si l'utilisateur est autorisé à accéder à certaines informations seulement s'il (et/ou le système) a effectué certaines actions de sécurité.

Bien plus que le modèle de contrôle d'accès lui-même (DAC, MAC, RBAC), l'attention des auteurs s'est plutôt portée : (1) sur la caractérisation du modèle de contrôle d'accès c'est-à-dire la granularité de protection (éléments, attribut, DTD, XMLSchema…), la modélisation des sujets (utilisateur, groupe, rôle…), les modes d'accès (lecture, mise à jour) [9,13, 37, 38, 54, 75, 78], (2) sur les algorithmes permettant d'implanter ce contrôle[35,23, 85, 96, 97, 117], (3) sur les canaux de communication et de distribution de l'information (dissémination sélective en mode Push ou Pull) [11, 12, 90, 114, 87] et (4) sur la non-corruption du modèle de contrôle d'accès par des techniques de chiffrement ou d'environnements d'exécution sécurisés [19].

Parmi tous ces travaux, ce qui nous a particulièrement intéressé dans cette section c'est la caractérisation du modèle de contrôle d'accès lui-même, c'est-à-dire les principes de base adoptés par les différents auteurs pour définir la politique de contrôle d'accès et cela quel que soit le contexte mis en œuvre, avec ou sans optimisation. Une étude plus approfondie des problèmes liés à la non corruption du modèle de contrôle d'accès par des techniques de chiffrement fait l'objet du chapitre VI.

Etant donné le très grand nombre de papiers présentés dans la littérature et certains aspects parfois très redondants d'un modèle à l'autre, nous n'avons pas choisi de présenter chaque approche de façon indépendante et exhaustive. Au contraire nous avons choisi d'en présenter une synthèse. C'est pourquoi dans une première partie, nous avons caractérisé les éléments fondateurs et communs à tous les modèles de contrôles d'accès. Cela fixe le cadre général. Ensuite nous avons repris chacun de ces éléments et affiner sa définition. Pour chacun des aspects, nous donnons les définitions spécifiques adoptées dans la littérature. Ces points seront utilisés et synthétisés à nouveau dans la conclusion du chapitre.

3.1 Les éléments fondateurs d'un modèle de contrôle d'accès pour XML

Avant de présenter les éléments fondateurs de modèles de contrôle d'accès, nous rappelons, brièvement, les concepts de base de XML, ainsi que le langage d'interrogation des documents XML le plus utilisé dans les modèles de contrôle d'accès. Ensuite, nous présentons les éléments fondateurs de modèle qui vont être détaillés dans la section suivante.

Ces éléments fondateurs représentent notre base de comparaison des modèles de contrôle d'accès.

- ***Les concepts de XML***

XML [115] est un langage de balisage pour décrire les informations semi-structurées. Un document XML est composé d'un ensemble *d'éléments* qui peuvent, à leur tour, contenir d'autres éléments (sous-éléments) formant ainsi une structure hiérarchique. La Figure 8 présente un exemple de document XML et sa représentation graphique (structure arborescente). Un élément contenant une portion du document est délimité par un *tag de début* de forme <*nom de tag* > (e.g, <Nom>, <Diagnostics>, …) et un tag de fin de la forme </*nom de tag* > (e.g, </Nom>, </Diagnostics>, …). Un élément vide est représenté par <*nom de tag/*>.

Une liste d'attributs de différents types peut être spécifiée pour chaque élément. Un attribut peut être un identifiant d'un élément souvent appelé id (e.g, id = "mrobert"), un lien pour référencer d'autres éléments dans un document via les attributs IDREF(S) ou des informations supplémentaires sur l'élément.

```
<Hôpital>

  <Dossier id="mrobert">
    <Nom>Martin Robert</Nom>
    <Adresse> 45 av des Etats Unis</Adresse>
    <Diagnostics>
      <Item>Cancer</Item>
    </Diagnostics>
  </Dossier>

</ Hôpital >
```

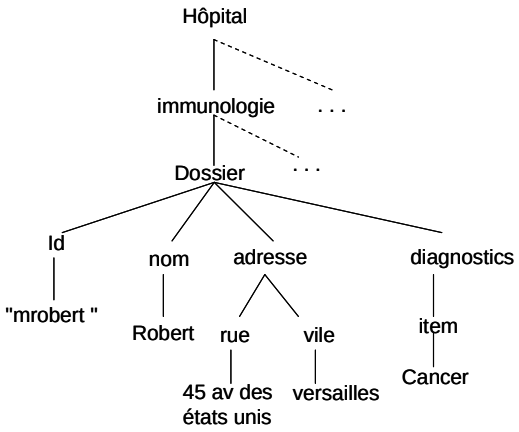


Figure 8 : Exemple du document XML et sa représentation graphique

Un document XML peut se classifier en deux catégories : un document XML est bien formé s'il obéit à la syntaxe de XML (e,g., chaque tag début non vide doit correspondre à un tag de fin). Un document XML est valide s'il est conforme à une DTD (Data Type Document). Une DTD est un fichier contenant une définition formelle d'un type particulier de document XML. En d'autres termes, elle définit la structure du document XML. Cette DTD est composée de deux parties : les déclarations d'éléments et les déclarations de liste d'attributs. La partie de déclarations d'éléments spécifie la structure des éléments contenus dans le document. En particulier, elle spécifie pour un élément donné ses sous-éléments et leurs cardinalités ('*' (0-n) ou '+' (1-n)), s'ils sont optionnels ('?')…. Le type de contenu de

chaque sous-éléments est, également, spécifié, il peut être EMPTY (l'élément n'aucun contenu), ANY (l'élément peut avoir n'importe quel contenu) ou #PCDATA (l'élément peut contenir du contenu). La partie déclaration de liste d'attributs spécifie, pour chaque élément, la liste des attributs en terme de nom, de type et de clause optionnelle (# IMPLIED indique un attribut optionnel, #REQUIRED indique un attribut obligatoire qui peut avoir également une valeur par défaut).

- ***Le langage d'interrogation***

La localisation des portions du document XML se fait via le langage d'expression XPath [116]. XPath utilise des axes pour représenter les associations structurelles entre les éléments et, entre les éléments et les attributs. Pour des raisons de simplicité, nous considérons trois axes qui sont fils (/), descendance (//) et attribut (@). Par exemple, /*hôpital//nom* sélectionne tous les noms des patients qui sont des descendants de l'hôpital.

Des conditions, appelées des prédicats peuvent être définies au niveau de l'expression XPath. Par exemple, /hôpital/dossier[@id= "mrobert"] : sélectionne le dossier du patient ayant comme attribut id ="mrobert". Pour sélectionner n'importe quel élément, XPath permet d'utiliser le joker (*).

- ***Le modèle de contrôle d'accès***

La plupart des modèles de contrôle d'accès pour XML sont basés sur un ensemble de règles d'autorisation. Une ***règle d'autorisation*** spécifie si un sujet donné a la permission (règle positive) ou l'interdiction (règle négative) d'accéder à un objet donné. Les ***sujets*** sont décrits dans un document XML et sont sélectionnés par des requêtes XPath. Un exemple simple de document concernant les sujets est présenté dans la Figure 10. L'***objet*** représente l'entité à laquelle le sujet a demandé l'accès. Il représente soit le document XML ou une partie du document. L'ensemble de règles d'autorisation est défini dans un document XML.

La Figure 9 présente un exemple de base de règles d'autorisation défini pour le document XML présenté dans la Figure 8. Nous supposons que ce document est partagé par trois groupes : les médecins, les infirmières et les secrétaires. Les médecins ont le droit de tout voir. Les infirmières ont le droit de tout voir à l'exception de l'élément diagnostic. Les secrétaires n'ont pas le droit de voir le diagnostic du patient ni les services où les patients sont traités (e,g., immunologie).

```
<Sujet>
 <utilisateurs>
  <member id="dupont">
   <name>Pierre Dupont</name>
  </member>
        .
        .
 </ utilisateurs >

<groupes>
  <personnel-médical>
      <Medecin>
         <member idref="dupont"/>
      </Medecin>
          .
          .
  </ personnel-médical >
</groupes>
 </Sujet>
```

Figure 10 : Exemple de base de sujets

```
<Base de règles d'autorisation>
</Sujets, /hôpital, +, cascade, +>
<//infirmière, /hôpital/immunologie//diagnostics, - , cascade >
<//Secrétaire, /hôpital/immunologie//diagnostics, -, cascade>
<//Secrétaire, /hôpital/immunologie, -, no propagation>
</Base de règles d'autorisation>
```

Figure 9 : Exemple de base de règles d'autorisation

Afin de minimiser le nombre de règles d'autorisation à définir, la structure hiérarchique du document XML peut être exploitée pour définir ce que nous appelons des ***politiques de propagation***. Ces politiques permettent à une règle d'autorisation définie explicitement sur un élément de se propager récursivement, par exemple, à ses descendants.

Par conséquent, chaque élément et attribut reçoit une règle d'autorisation soit d'une façon explicite, soit d'une façon implicite. Dans le cas où aucune règle d'autorisation n'a été définie pour un élément (ou attribut) donné que ce soit explicitement ou implicitement, une ***politique par défaut*** lui sera appliquée. Cette dernière peut être fermée, l'accès est interdit et, par conséquent, la règle d'autorisation par défaut est négative. Inversement, dans une politique ouverte l'accès est autorisé et la règle d'autorisation par défaut est positive.

Cependant, la possibilité de combiner des règles positives et des règles négatives et l'exploitation de la structure hiérarchique du document amènent à des situations conflictuelles. Pour résoudre ces conflits, un ensemble de ***politiques de gestion des conflits*** ont été proposées dans la littérature.

Concevoir un modèle de contrôle d'accès pour XML consiste, dans un premier temps, à identifier les sujets et les objets. Ensuite, à définir les règles d'autorisation qui s'appliquent sur eux. L'administrateur doit choisir, également, les politiques de propagation et les politiques de résolution de conflit mises en place. Le choix d'une politique par défaut est également nécessaire.

3.2 Comparaison et synthèse des approches existantes

Dans la section précédente, nous avons présenté, très brièvement, les éléments fondateurs des modèles de contrôle d'accès. L'objectif de cette section est de reprendre ces éléments et

de les définir d'une façon générale, ensuite de montrer d'une façon spécifique comment les approches les plus connues, Bertino [8, 10, 13], Gabillon [54] et Damiani [37, 38], les caractérisent. Des références à d'autres travaux de l'état de l'art seront également faites au fur et à mesure.

3.2.1 L'expression des règles d'autorisation

La plupart des modèles de contrôle d'accès représentent une règle d'autorisation sous forme d'un 5-tuplet <***Sujet, Objet, Action, Signe, Propagation***>. Cette règle représente la forme de base de modèle de contrôle d'accès discrétionnaire. Le choix de représentation de la règle d'autorisation sous cette forme, dans cet état de l'art, est lié, plus particulièrement, au choix de modèle de contrôle d'accès utilisé pour notre étude décrite dans le chapitre IV.

Les approches existantes diffèrent plus ou moins dans la définition des concepts de cette règle d'autorisation. Dans ce qui suit nous décrivons les quatre premiers concepts et nous les comparons dans les différentes approches. Le concept de *propagation* sera décrit par la suite.

- **Sujet** : entité qui demande l'accès aux objets. Dans XML, les sujets sont décrits dans un fichier XML sélectionnés par des requêtes XPath. Un sujet peut être un utilisateur, un groupe d'utilisateurs, rôle, profil, adresses IP… Les modèles existants diffèrent dans la caractérisation de ces sujets.

  Dans le modèle de Bertino [8, 10, 13], le sujet peut être un *identifiant* ou un *profil* d'utilisateur (credential[3]). Par exemple, *//secrétaire*[*département=cardiologie*] désigne toutes les secrétaires qui travaillent dans le département cardiologie.

  Dans le modèle de Damiani [37, 38], le sujet est caractérisé par un triplet <*Id_d'utilisateur/groupe, adresse IP/pattern, domaine/pattern*>. Par exemple, <*infirmières, *.149.100, *cardiologie.hôpital.com*>. Le modèle de Gabillon [54] définit le sujet comme un *id* de l'utilisateur ou un groupe d'utilisateurs. Par exemple, la requête *//secrétaire* sélectionne toutes les secrétaires.

- **Objet** : représente la granularité de protection. L'objet peut être une DTD, un document XML, un élément, un attribut, un lien (Idref). Les objets sont sélectionnés par des requêtes XPath. Les modèles de contrôle d'accès existants varient légèrement dans la définition de leur granularité de protection.

  Dans les modèles de Damiani [37, 38] et de Bertino [8, 10, 13] l'objet peut être une DTD, un document XML, un élément/attribut. Le modèle de Bertino protège également les liens *Idref* entre les éléments dans un document XML. Le modèle de Gabillon [54] définit une granularité encore plus fine, l'objet peut être un document XML, un élément, un attribut/texte/commentaire.

- **Action** : représente toutes les opérations qui peuvent être effectuées sur un objet.

[3] Un profil peut être caractérisé par l'age, la nationalité, la position organisationnelle …

Ces opérations sont la lecture, l'écriture, la suppression et la modification.

Le modèle de Bertino définit deux types d'opération (privilège) *browsing* et *authoring* [109, 13, 8]. Le privilège *browsing* permet de lire ou de naviguer à travers les liens (e.g Idref). Le privilège *authoring* permet d'ajouter ou de modifier des éléments (ou leur contenu).

Le modèle de Damiani [38] s'est focalisé sur l'opération de lecture dans sa première description du modèle de contrôle d'accès. Néanmoins, dans leur version étendue du modèle [37], les auteurs décrivent brièvement les privilèges d'écriture (correspondant à l'ajout d'un élément/attribut), de suppression et de mise à jour (correspondant à changer la valeur d'un attribut ou à changer le texte d'un élément).

Dans le modèle de Gabillon [54], la seule opération traitée est l'opération de lecture. Les nouvelles versions du modèle de Gabillon [55, 56][4] traitent les opérations d'écriture (insertion, suppression, mise à jour). Même si certains modèles font référence aux opérations d'écriture, peu de travaux se sont intéressés à la spécification et à la sémantique des différentes opérations d'écriture [55, 56, 78]. C.Lim [78] a repris, exactement, le modèle de Damiani [38] pour intégrer d'autres types d'opérations d'écriture (InsertBefore|After, Replace, Rename). Il ajoute une autre sémantique aux opérations dans la règle d'autorisation. Cette sémantique consiste à autoriser certaines opérations par rapport à la structure du document. Par exemple, Alice a le droit d'insérer un nouvel élément tout en changeant la structure prédéfinie par la DTD. Alors que Bob a le droit d'insérer un nouvel élément si et seulement si cela ne change pas la structure du document. Nous remarquons que la même opération a deux sémantiques différentes en fonction de la structure de la DTD.

- **Signe** : représente le mode d'accès aux objets. En général, dans tous les modèles de contrôle d'accès, le signe prend deux valeurs : i) " +" l'accès à l'objet sélectionné par la règle d'autorisation est autorisé, ii) " -" l'accès à l'objet sélectionné par la règle d'autorisation est interdit. Le modèle de Gabillon [55, 56] a introduit un autre mode d'accès appelé "*position*". Ce mode d'accès permet de conserver l'existence d'un nœud tout en cachant sa valeur (renommage du tag).

3.2.2 Politiques de propagation

Pour minimiser le nombre de règles d'autorisation à définir dans une base de règles d'autorisation, les modèles de contrôle d'accès exploitent la structure hiérarchique du document pour définir ce que nous appelons des *politiques de propagation*. La notion de propagation introduit le principe "*appliquer par défaut*" pour les règles d'autorisation. Nous définissons dans ce qui suit les différents types de propagation.

[4] Les sujets dans ces versions sont considérés comme un utilisateur ou un rôle.

- **Politiques de propagation au niveau du document (instance)**

Grâce aux associations existantes entre les éléments et les sous-éléments (attributs et liens), une règle d'autorisation spécifiée sur un élément donné, se propage récursivement à tous ses sous-éléments/attributs. Le contrôle du niveau de propagation dans la hiérarchie (du document XML) peut être spécifié. Les différentes options de propagation les plus connues dans la littérature sont :

- No propagation : aucune propagation n'est appliquée. La règle d'autorisation s'applique uniquement à l'élément et à ses propres attributs spécifiés par la requête XPath.
- 1-niveau : la règle d'autorisation se propage à tous les fils directs de l'élément sélectionné par la requête.
- Propagation descendante : la règle d'autorisation se propage à tous les sous-éléments/attributs de l'élément donné.
- Propagation ascendante : la règle d'autorisation se propage à tous les ancêtres de l'élément en question [75].

- **Politiques de propagation au niveau de la DTD/XMLSchema**

L'association entre la DTD/XMLSchema et ses instances valides (ou partialement conformes à la DTD/XMLSchema) peut être exploitée, également, pour définir des politiques de propagation. Les règles d'autorisation définies au niveau de la DTD/XMLSchema peuvent se propager à tous les documents (parties des documents) reconnus comme instances valides. Comme le document XML n'est pas toujours valide par rapport à une DTD/XMLSchema, différentes possibilités (voir la Figure 11) peuvent se présenter dans la gestion des politiques de propagation. Nous faisons la référence, dans cette figure, à la DTD mais cela reste vrai, également, pour XMLSchema.

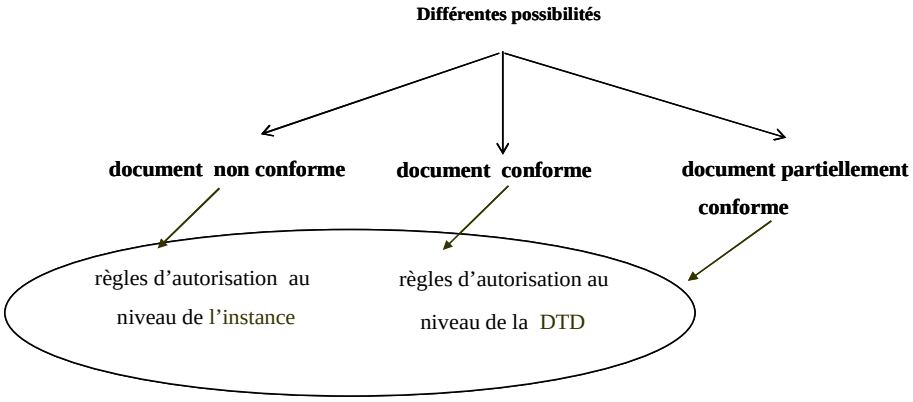


Figure 11 : Politiques de propagation et la présence de la DTD

1. *Les documents non conformes à la DTD* : les règles d'autorisation doivent être définies au

niveau des instances du document elles-mêmes.

2. *Les documents conformes à la DTD* : les règles d'autorisation peuvent être définies au nivau de la DTD et toutes les instances valides bénéficieront de ces règles d'autorisation par le biais de la propagation.

3. *Les documents partiellement conformes à la DTD* : il existe uniquement quelques portions du document XML qui sont conformes à la DTD. Dans ce cas, les portions conformes à la DTD bénéficieront des règles d'autorisation définies au niveau de la DTD. Par contre, d'autres politiques de contrôle d'accès seront envisagées pour traiter les portions non conformes du document XML qui sont :

   - *Politique fondé sur la propagation :* un élément du document non conforme peut bénéficier de la règle d'autorisation de son ancêtre (conforme).

   - *Politique fondé sur l'affinité :* un élément du document peut bénéficier de la règle d'autorisation d'un autre élément sémantiquement reliés en utilisant l'ontologie (e.g. dossier médical et enregistrement médical sont synonymes).

   - *Politique fondé sur le document :* cette politique est adoptée lorsqu'aucune des deux politiques précédemment définies ne peut être appliquée (par propagation ou à base d'affinité). Par conséquent, la règle d'autorisation est définie explicitement sur l'élément en question.

Les modèles de contrôle d'accès diffèrent plus ou moins dans leurs choix des politiques de propagation. Les modèles de Damiani [37, 38, 39] adoptent les options de propagation cascade (appelée dans leur modèle *récursive*) et no propagation (appelée locale). Leurs modèles adoptent les politiques de propagation au niveau de la DTD qui s'appliquent uniquement les instances valides. Les modèles de Bertino [10, 9, 13, 8] adoptent les options de propagation 1-niveau (qui devient *n-niveau* dans un autre modèle Bertino[11]), no propagation, et cascade. La particularité du modèle de Bertino [8] est sa capacité à définir des politiques de propagation qui prennent en compte les documents valides (conformes à une DTD), les documents bien formés et les documents partiellement conformes à une DTD. Dans le modèle de Gabillon [54], l'option de propagation n'a pas été spécifiée d'une façon explicite. Implicitement, il adopte l'option de propagation cascade. Contrairement à la plupart des modèles de contrôle d'accès pour XML qui définissent les droits d'accès sur la DTD, [118, 119] définissent plutôt des droits d'accès sur XMLSchema.

3.2.3 Politique de résolution des conflits

Tous les modèles de contrôle d'accès pour XML définissent des règles d'autorisation positives et des règles d'autorisation négatives. La définition des règles négatives permet de gérer des exceptions [72] et de limiter la portée de propagation des règles positives et vice-versa. Cela est intéressant lorsque, par exemple, l'accès est autorisé à tout le document sauf à

quelques éléments/attributs de ce document. Il suffit dans ce cas de définir une règle positive au niveau du document avec option de propagation cascade, et une (des) règle(s) négative(s) pour la portion du document interdite.

L'exploitation de la structure du document pour définir les politiques de propagation minimise le nombre de règles à définir. L'utilisation des règles d'autorisation positives et des règles d'autorisation négatives permet de gérer des exceptions et rend le modèle de contrôle d'accès plus flexible. Mais, la combinaison de ces deux options (propagation et règles positives/négatives) amène à des situations conflictuelles. Ces situations apparaissent lorsqu'un élément/attribut possède deux règles d'autorisation, une règle d'autorisation interdit son accès et l'autre règle d'autorisation autorise son accès en même temps. Ce conflit peut survenir lorsque l'élément/attribut bénéficie d'une règle d'autorisation par le biais de la propagation, et d'une autre règle d'autorisation qui lui a été définie d'une façon explicite (ou les deux règles d'autorisation lui ont été définies d'une façon explicite). Pour remédier à ce problème différentes politiques de résolution de conflit sont adoptées par les modèles de contrôle d'accès pour XML :

- *Définition de niveaux de priorité* : à chaque règle d'autorisation est attribué un niveau de priorité (numéro) et la règle d'autorisation ayant un niveau de priorité plus élevé est prioritaire.
- Les règles d'autorisation définies au niveau du document sont prioritaires par rapport à celles définies au niveau de la DTD ou vice versa.
- L'objet/sujet le plus spécifique est prioritaire : la règle d'autorisation définie à un niveau donné de la hiérarchie XML est plus prioritaire par rapport à celle définie à un niveau plus haut dans la hiérarchie XML.
- La règle d'autorisation négative est prioritaire par rapport à la règle d'autorisation positive.

Généralement, les modèles de contrôle d'accès combinent plusieurs politiques de résolution de conflit et l'administrateur de sécurité fait un choix sur cette combinaison.

Les modèles de contrôle d'accès pour XML diffèrent dans la combinaison de ces politiques de résolution de conflit. Les modèles de Bertino adoptent la politique *" l'objet le plus spécifique est prioritaire*". Dans le cas où le conflit persiste, le modèle adopte la politique "*la règle d'autorisation négative est prioritaire que la règle d'autorisation positive*".

Bien que la plupart des modèles de contrôle d'accès pour XML adoptent le principe le plus spécifique est prioritaire, les modèles de Damiani font une exception. Ils associent aux options de propagation les types hard et soft. Le type hard s'applique au niveau de la DTD et impose que toutes les instances de cette DTD respectent la règle d'autorisation définie au niveau de la DTD. Le type soft s'applique au niveau de l'instance et signifie que la règle

d'autorisation définie sur le document s'applique uniquement s'il n'existe pas de règle d'autorisation définie au niveau de la DTD. La combinaison de ces deux types (hard et soft) avec les options de propagation conduit à huit types de règle d'autorisation. Leurs sémantiques donnent un ordre de priorité entre les règles d'autorisation qui est : LDH (local hard authorization) > RDH (recursive hard authorization) > L (local authorization) > R (recursive authorization) > LD (local authorization over the DTD) > RD (recursive authorization over the DTD) > LS (local soft authorization) > RS (recursive soft authorization). En plus de cet ordre entre les règles d'autorisation, les modèles de Damiani utilisent la politique de résolution de conflit "*le sujet le plus spécifique est prioritaire*". Dans le cas où le conflit persiste, "*la règle d'autorisation négative est prioritaire que la règle d'autorisation positive*". Ces deux dernières politiques de résolution de conflit sont utilisées dans le cas où deux règles d'autorisation du même type (les deux sont locales ou les deux sont récursives, ….) sont définies avec des signes différents.

Le modèle de Gabillon utilise la politique de résolution de conflit basée sur les niveaux de priorité attribués à chaque règle d'autorisation. Pour cette raison, la forme de la règle d'autorisation englobe l'option priorité (<sujet, objet, accès, priorité>) qui prend un nombre entier. Plus précisément, sa politique de résolution de conflit dit : i) s'il existe un conflit entre un ensemble de règles alors les règles ayant la plus haute priorité sont sélectionnées. ii) S'il y a plus qu'une règle d'autorisation sélectionnée alors seule la dernière règle dans l'ordre de lecture de la feuille XAS (XML Authorization Sheet) est conservée.

3.2.4 Vue autorisée –Requête

Lorsqu'un utilisateur demande l'accès à un document, il reçoit une ***vue*** (c'est-à-dire un/une portion du document XML) de ce document compatible avec ses droits d'accès. Deux approches de vérification des droits peuvent être distinguées. La première approche est basée sur matérialisation de la vue autorisée qui consiste à construire une vue globale (autorisée) pour un utilisateur donné (demandeur d'accès). Ensuite, la requête d'accès posée par l'utilisateur est évaluée sur cette vue globale. La deuxième approche est basée sur la réécriture de requêtes qui consiste à transformer la requête d'accès posée par l'utilisateur en une autre requête qui sera, ensuite, évaluée sur le document source. Cette deuxième approche est utilisée, plus particulièrement, pour améliorer la performance du modèle de contrôle d'accès. Dans notre travail de recherche, nous ne nous sommes pas intéressés particulièrement aux problèmes de performance. Néanmoins, nous citons quelques travaux qui se sont intéressés à ce problème dans cette section.

3.2.4.1 Matérialisation de la vue autorisée

Dans cette approche, l'exécution de la requête de l'utilisateur se fait en deux étapes. La première étape consiste à faire un prétraitement sur tout le document source qui supprime tous les éléments/attributs interdits pour l'utilisateur. En d'autres termes, la vue autorisée

globale du document source (indépendamment de la requête d'accès) est construite. Le calcul d'une telle vue est basé sur un ***processus de marquage*** [9, 11, 37, 38, 85]. Ce dernier s'effectue en plusieurs étapes :

- ***Marquage des éléments/attributs*** : dans un premier temps, les règles d'autorisation concernant l'utilisateur (demandeur d'accès) sont identifiées. Ensuite, chaque élément/attribut sélectionné par une règle d'autorisation est marqué par cette dernière. Si la règle d'autorisation correspondante est du type cascade, alors elle s'applique d'une façon récursive à tous les descendants de l'élément.
- ***Résolution des conflits*** : consiste à résoudre les conflits entre les règles d'autorisation conflictuelles du même élément/attribut en appliquant les politiques de résolution de conflits choisies par l'administrateur de sécurité. La Figure 12 présente le marquage final du document XML pour le groupe de secrétaires après l'application de la base de règle définie précédemment dans la Figure 9.

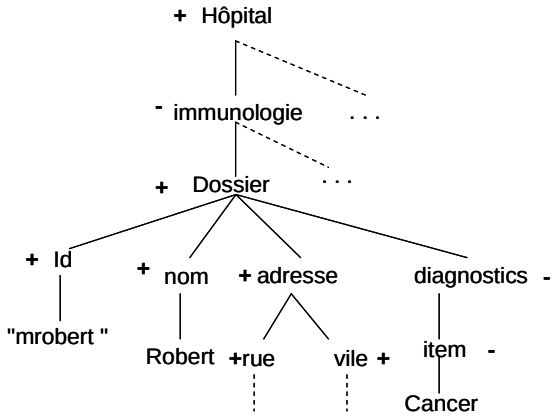


Figure 12 : Le document marqué

- ***La vue autorisée du document*** : chaque élément/attribut marqué négativement est normalement supprimé. Par contre, nous verrons par la suite, les différentes sémantiques adoptées pour un tel marquage par les différents modèles de contrôle d'accès existants. Si l'utilisateur, bien évidemment, a le droit de tout voir, la vue autorisée de ce dernier coïncide avec le document source lui-même

Une fois la vue globale autorisée calculée, la requête d'accès peut être évaluée.

3.2.4.2 Réécriture de requête

La réécriture de requête consiste à transformer la requête originale posée par l'utilisateur en une autre requête qui sera évaluée sur le document source. Cette approche est utilisée pour remédier au problème de performance due à la matérialisation de la vue autorisée. Fan [46] propose un modèle de contrôle d'accès à base de vues sécurisées

(restrictives) contenant uniquement les parties que l'utilisateur a le droit de voir. L'utilisateur, ensuite, ne pourra poser de requêtes que sur la vue qui lui y est autorisée.

La vue sécurisée *V* est définie comme une paire V = (D_v, σ), où D_v : représente la vue restrictive de la DTD originale obtenue à partir d'une politique de contrôle d'accès. σ : définit les *annotations des requêtes XPath* utilisées pour extraire les données accessibles du document XML original. Ces annotations sont définies au niveau de la DTD et sont invisibles pour les utilisateurs. Afin d'éviter la matérialisation des vues, Fan [46] propose un algorithme de réécriture de requêtes qui transforme automatiquement une requête XPath posée sur la ***vue sécurisée*** en une autre requête équivalente sur la DTD originale (source) en exploitant les *annotations des requêtes XPath* pour extraire les parties autorisées.

Le modèle de S.Cho[35] basé sur le modèle multi-niveaux réécrit la requête originale en ajoutant un prédicat à chaque élément de cette requête qui compare le niveau de sécurité de cet élément avec le niveau de sécurité de l'utilisateur. Afin d'optimiser l'évaluation de la requête, la réécriture est basée sur des informations spécifiées au niveau de la DTD.

Une autre approche a été adoptée par Gabillon [54] en utilisant XSLT pour calculer la vue. Il traduit une bonne fois pour toutes la base de règles d'autorisation en une feuille de style XSLT et cette base de règles ne sera plus utilisée par la suite (à moins qu'elle soit modifiée, dans ce cas elle est automatiquement retraduite). Pour chaque utilisateur une feuille de style est définie. Lorsqu'un utilisateur accède à un document, le système lui calcule automatiquement une vue du document à l'aide de la feuille de style.

4 Conclusion et problèmes ouverts

Dans ce chapitre nous avons décrit les différentes variantes des modèles de contrôle d'accès existants dans la littérature (DAC, MAC, RBAC, TMAC, Or-BAC) rappelant ainsi les caractéristiques de ces modèles, qui sont comme nous l'avons montré indépendants des modèles de données. L'évolution des modèles montre le besoin croissant d'une plus grande expressivité et d'une plus grande flexibilité dans la définition et la gestion des droits d'accès. Ces modèles ont été mis en œuvre que cela soit dans le cadre relationnel, objet, mais également XML comme nous l'avons montré dans la section précédente. Mon travail de recherche ne porte pas sur la mise en œuvre de ces politiques dans le cadre de XML, mais plutôt sur le modèle de contrôle d'accès XML lui-même car malgré tous les efforts consacrés, la définition d'une sémantique claire et non ambiguë reste à faire. Ce problème de sémantique n'est pas lié au type de modèle adopté (DAC, MAC et RBAC) mais plutôt lié à la conception du noyau du modèle lui-même. Normalement, pour une seule politique de contrôle d'accès donnée, nous devons avoir ***une et une seule sémantique*** qui amène toujours au même résultat (vue autorisée). Les modèles de contrôle d'accès XML existants diffèrent d'une façon remarquable dans leur sémantique. Par exemple, supposons le document XML décrit dans la Figure 12 et dont les nœuds sont marqués par le signe positif si le nœud est

autorisé et négatif sinon. Pour un même marquage, c'est-à-dire pour une même politique de contrôle d'accès, différentes vues autorisées sont générées selon les sémantiques des modèles existants. Cela est décrit dans la Figure 13.

La vue autorisée présentée dans la fig.16 (a) est générée par les modèles de contrôle d'accès Bertino [8], Gabillon [54], Murata [85], Wang [118] et Zhang [119] qui interdissent l'accès à l'élément autorisé dans un sous-arbre interdit. Ces modèles imposent une sémantique qui dit : si l'accès à un élément est ***non-autorisé*** alors l'accès à tous ses descendants est également ***non-autorisé***. Par conséquent, ces modèles de contrôle d'accès souffrent de la disponibilité des informations et contredisent la base de règles d'autorisation définie en interdisant l'accès aux éléments qui étaient autorisés au départ.

D'autres modèles de contrôle d'accès comme Damiani [38, 37], Kudo [75], Fan [46] et Gabillon [55] permettent l'accès à l'élément autorisé dans un sous-arbre non autorisé (c'est-à-dire le sous-arbre est enraciné par un élément non-autorisé). Ces modèles peuvent être décomposés en deux sous-catégories. La première sous-catégorie est composée des modèles de Damiani [37, 38] et Kudo [75] qui révèlent tous les ancêtres (uniquement leurs tags) non-autorisés d'un élément accessible (autorisé). Ces modèles contredisent la base de règles d'autorisation prédéfinie car ils rendent visible ce qui a été interdit au départ par la base de règles. Par conséquent, l'information à protéger est divulguée. La vue autorisée du document XML pour cette catégorie de modèles est présentée dans la fig.16 (b).

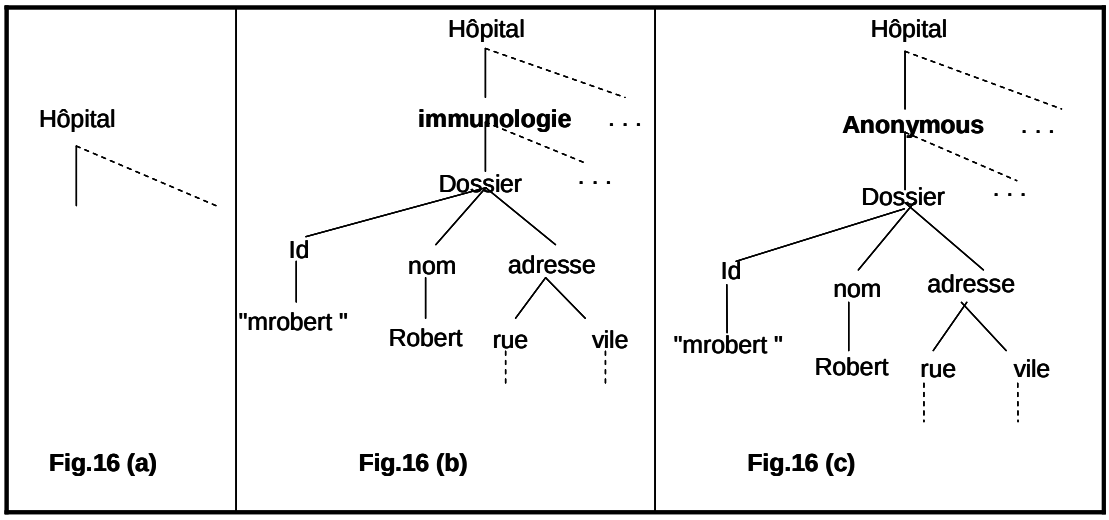


Figure 13 : Les différentes vues retournées selon la sémantique des modèles existants

Afin de résoudre le problème de disponibilité posé par les modèles précédents et de minimiser la divulgation d'information non autorisée, la deuxième sous-catégorie des modèles de contrôle d'accès de Fan [46] et Gabillon [55] proposent de renommer l'élément inaccessible ayant des descendants accessibles (autorisés). Cela est exprimé dans les modèles de Gabillon [55] par un privilège spécifique (*position*) qui garde l'existence de l'élément sans connaître sa valeur (par le biais du renommage). La vue autorisée du document XML

est présentée dans la fig.16 (c). Concernant l’élément inaccessible ayant un descendant accessible le modèle de Fan [46] donne une autre alternative qui consiste à supprimer cet l’élément si cela ne viole pas la structure de la DTD initiale. La sémantique de l’élément inaccessible, dans ce modèle, dépend de la DTD.

Nous constatons que les modèles existants focalisent le contrôle d’accès sur les ***éléments*** et les ***attributs*** et supposent que ***la vue autorisée est un sous-ensemble du document original***, que la sémantique de contrôle d’accès sur les éléments diffère d’un modèle à un autre particulièrement lorsque qu'il existe un élément intermédiaire non autorisé entre deux éléments autorisés d'un même chemin dans le document XML. Cette différence entre les modèles est fortement liée à la difficulté de ***définir la vue exacte d’un chemin*** qui amène à un élément autorisé dans les modèles de contrôle d'accès existants. Ces trois aspects (ceux en gras dans le texte) posent, nous le verrons dans le chapitre suivant, de nombreux problèmes car ils restreignent les possibilités quant à l'expression de certaines politiques de contrôle d'accès notamment dans le contexte de la protection des données personnelles, tel que le dossier médical.

Chapitre III – Protection des données personnelles : de la loi à la pratique

1 Introduction

Dès les années 70 les juristes s'inquiètent de l'idée que les machines informatiques, en permettant la collecte, le stockage, le traitement, la diffusion des informations relatives aux personnes physiques, puissent servir d'instruments portants atteinte aux libertés, spécialement à la vie privée des individus. En effet, l'avènement de la technologie informatique actuelle facilitant la collecte des données personnelles augmente considérablement la possibilité d'une utilisation abusive de ces données. Afin de faire face à cette utilisation abusive, la plupart des pays ont défini des mesures ***législatives*** spécifiques à l'informatique pour protéger les données personnelles et, en l'occurrence, l'intimité et le droit à la vie privée reconnus dans l'article 8 de la convention européenne de la sauvegarde des droits de l'homme et des libertés fondamentales [26].

L'objectif de ce chapitre n'est pas de faire une étude exhaustive de toutes les législations existantes ou de critiquer les aspects juridiques de ces dernières. Mais plutôt, d'étudier le problème de traduction des principes législatifs en terme de politiques de contrôle d'accès et d'analyser le comportement des modèles de contrôle d'accès existants vis-à-vis de ces principes. Etudier le problème dans l'absolu n'a pas de sens et constituerait une tâche trop vaste.

Dans une première partie de ce chapitre, nous résumons les fondements des législations de portée internationale, européenne et nationale en ce qui concerne la protection des données personnelles. Puis, nous décrivons les spécificités liées à la protection des données à caractère médicale puisque cela concerne plus spécifiquement notre domaine d'application. L'étude des textes de lois permet de mettre en évidence deux principes de base importants pour la protection des informations personnelles, et particulièrement celles du domaine de la santé. Le premier principe est le **principe de finalité ou "need-to-know"**, il limite l'accès à l'information, seule l'information strictement utile à une tâche doit pouvoir être accédée. Le deuxième est le **principe de consentement**, il empêche la divulgation d'information sans le consentement explicite du propriétaire de la donnée. Autrement dit, certaines règles de contrôle d'accès doivent pouvoir être personnalisées. En effet, le patient garde certaines

prérogatives concernant son dossier médical. Il peut décider de la façon dont son dossier est accessible et cette façon peut différée d'un patient à un autre.

Dans une deuxième partie de ce chapitre, nous nous intéressons plus particulièrement au dossier médical personnel qui constitue notre exemple de référence. Sur la base de cet exemple nous montrons comment les modèles de contrôle d'accès XML existants ne permettent pas de traduire les principes de consentement et de finalité érigées par les lois.

2 Les législations

Dans cette section, nous détaillons tout d'abord les législations relatives à la protection des données personnelles, puis nous étudions les spécificités liées à la protection des données dans le domaine de la santé.

2.1 Législations relatives à la protection des données personnelles

Le droit à la vie privée, sans lien avec le développement de l'informatique, est consacré par la déclaration universelle des droits de l'homme de l'ONU 1948 (Art.12) et par la Convention européenne de sauvegarde des droits de l'Homme et des libertés fondamentales de 1950 (Article 8) [26]. Mais il est remarquable que dès les années 1970 de nombreux pays se dotent de lois nationales spécifiques à l'informatique. Par exemple, le Land de Hesse en Allemagne 1970 [36] et la loi du 6 janvier 1978 en France, lois relatives à l'informatique aux fichiers et aux libertés [79]. A coté des lois nationales, les données à caractère personnel sont encadrées par des textes internationaux qui serviront de références et source d'inspiration pour les états. Nous citons, par exemple, l'action de l'Organisation de Coopération et de Développement Economique (OCDE) qui fixe la recommandation du 23 septembre 1980 [91]. L'ONU a adopté des lignes directrices relatives aux fichiers automatisés le 14 décembre 1990 par le biais de l'assemblée générale [64].

Dans ce chapitre, nous présentons, dans un premier temps, les lignes directrices imposées par l'ONU. L'objectif est d'introduire les principes fondateurs de la protection des données à caractère personnel. Par la suite, nous décrivons une analogie de ces principes avec la législation européenne[5]. Cette analogie sera l'occasion d'introduire un ensemble de définitions. La précision apportée par cette législation, par rapport aux lignes directrices de l'ONU, permet sa transposition dans chaque état membre de l'union européenne. Pour éviter les redondances des textes des lois (plus précisément les principes législatifs), nous présentons dans la section 2.3, au niveau national, uniquement, le *privacy Act* de 1974 [93] des Etats-Unis qui représente une approche différente de la protection des données à caractère personnel.

[5] La loi de 6 janvier 1978 en France présente des similarités fortes avec la législation européenne puisque des transpositions ont été exigées par la directive européenne.

2.1.1 Texte International

La mission de l'Organisation des Nations Unies (ONU) concernant la protection des données à caractère personnel informatisées était de proposer des principes de base et des garanties minimales devant être adoptés par les législations nationales. En 1990, l'ONU adoptait des « *Principes directeurs pour la réglementation des fichiers de données personnelles informatisés* » qui s'appliquent aux fichiers publics et privés [64]. L'ONU laisse l'initiative à chaque etat les modalités d'application des règlements concernant ces fichiers informatisés. Les principes qui devraient être prévus dans les législations nationales sont :

1. *Principe de licéité et de loyauté :* la collection et le traitement des données à caractère personnel devraient se faire d'une façon licite et loyale. Ces données ne devraient être utilisées à des fins contraires aux buts et aux principes de la Charte des Nations Unies.

2. *Principe d'exactitude* : la vérification de l'exactitude et la pertinence des données enregistrées[6] devraient être faites par les personnes responsables de l'établissement d'un fichier ou celles responsables de leur mise en œuvre.

3. *Principe de finalité* : la finalité de création et d'utilisation d'un fichier devraient être spécifiées, justifiées et, lors de la mise en œuvre, la personne concernée doit être informée, afin qu'il soit ultérieurement possible de vérifier si : i) toutes les données personnelles collectées et enregistrées restent pertinentes par rapport à la finalité poursuivie ; ii) aucune desdites données personnelles n'est utilisée ou divulguée, sauf accord de la personne concernée, à des fins incompatibles avec celles ainsi spécifiées; iii) la durée de la conservation des données personnelles n'excède pas celle permettant d'atteindre la finalité pour laquelle elles ont été enregistrées.

4. *Principe d'accès par les personnes concernées :* chaque personne justifiant de son identité a le droit de connaître si ses données personnelles font l'objet d'un traitement. La personne a le droit de rectifier ou détruire des données en cas d'enregistrement illicite, injustifié ou inexact. La personne concernée a, également, le droit de savoir tous les tiers ayant accès à ces données personnelles tel que le principe le mentionne "*lorsqu'elles sont communiquées, d'en connaître les destinataires*" et d'en connaître le destinataire lors de la communication. Il est souhaitable que les dispositions de ce principe s'appliquent à toute personne, quelle que soit sa nationalité ou sa résidence.

5. *Principe de non-discrimination* : toute donnée pouvant engendrer une discrimination[7] illégitime ne devrait être collectée. Des dérogations à ce principe sont prévues sous le principe 6.

6. *Faculté de dérogation* : Des dérogations aux principes 1 à 4 ne peuvent être autorisées

[6] Elles doivent être aussi complètes que possible et mises à jour.

que si elles sont nécessaires pour protéger la sécurité nationale, l'ordre public, la santé ou la moralité publique ainsi que, notamment, les droits et libertés d'autrui. Ces dérogations devraient être prévues par la loi qui fixe les limites et exige des garanties appropriées. Les dérogations au principe 5 concernant la discrimination ne pourraient être autorisées que dans les limites prévues par la charte internationale des droits de l'homme.

7. *Principe de sécurité* **:** toute mesure de sécurité devrait être prise non seulement contre les risques humains tel que la gestion des droits d'accès, l'utilisation détournée des données et contamination par des virus informatiques, mais aussi contre les accidents naturels (destruction par sinistre).

8. *Contrôle et sanctions* : le respect des principes précités devrait être contrôlé, en conformité avec le système juridique interne, par une autorité désignée par une législation. Cette autorité devrait présenter des garanties de justice, d'indépendance à l'égard des personnes ou organismes responsables des traitements et de leur mise en oeuvre, et de compétence technique. Des sanctions devraient être prévues (ainsi que des recours individuels appropriés) dans le cas de non respect des principes précités.

9. *Flux transfrontières des données* : la circulation des données à caractère personnel d'un pays à l'autre ne peut se faire uniquement si les différents pays présentent des garanties comparables au regard de la protection de la vie privée. En absence des garanties comparables, des limitations à cette circulation ne peuvent être imposées indûment et seulement dans la stricte mesure où la protection de la vie privée l'exige.

10. *Champs d'application* : les présents principes devraient s'appliquer en premier lieu à tous les fichiers informatisés publics et privés et, d'une façon facultative, aux fichiers traités manuellement.

2.1.2 Législation européenne

L'objectif de la directive européenne [44] est de faciliter la circulation des données à caractère personnel entre la communauté européenne tout en respectant la vie privée des personnes. Cela ne peut se réaliser uniquement si le niveau de protection des droits et des libertés des personnes à l'égard des traitements des données à caractère personnel est équivalent dans tous les états membres. Pour cette raison, l'objectif principal de la directive est d'harmoniser la législation sur la protection du traitement des données à caractère personnel pouvant porter atteinte à la vie privée d'une personne. Par rapport au texte du l'ONU, la directive européenne est donc plus précise et plus détaillée. Le ***consentement*** de la personne et le principe de ***need to know*** sont affirmés avec force dans cette directive. Ci-dessous, nous donnons, dans un premier temps, les définitions des termes utilisés :

- ***Données à caractère personnel*** *: "toute information concernant une personne*

[7] Par exemple, les informations sur l'origine raciale ou ethnique, la vie sexuelle …

physique identifiée ou identifiable (personne concernée); est réputée identifiable une personne qui peut être identifiée, directement ou indirectement, notamment par référence à un numéro d'identification ou à un ou plusieurs éléments spécifiques, propres à son identité physique, physiologique, psychique, économique, culturelle ou sociale";

- ***Traitement de données à caractère personnel (traitement)***: "*toute opération ou ensemble d'opérations effectuées ou non à l'aide de procédés automatisés et appliquées à des données à caractère personnel, telles que la collecte, l'enregistrement, l'organisation, la conservation, l'adaptation ou la modification, l'extraction, la consultation, l'utilisation, la communication par transmission, diffusion ou toute autre forme de mise à disposition, le rapprochement ou l'interconnexion, ainsi que le verrouillage, l'effacement ou la destruction*" ;

- ***Fichier de données à caractère personnel (fichier):*** "*tout ensemble structuré de données à caractère personnel accessible selon des critères déterminés, que cet ensemble soit centralisé, décentralisé ou réparti de manière fonctionnelle ou géographique*";

- ***Responsable du traitement:*** "*la personne physique ou morale, l'autorité publique, le service ou tout autre organisme qui, seul ou conjointement avec d'autres, détermine les finalités et les moyens du traitement de données à caractère personnel; lorsque les finalités et les moyens du traitement sont déterminés par des dispositions législatives ou réglementaires nationales ou communautaires, le responsable du traitement ou les critères spécifiques pour le désigner peuvent être fixés par le droit national ou communautaire*";

- ***Sous-traitant*** : "*la personne physique ou morale, l'autorité publique, le service ou tout autre organisme qui traite des données à caractère personnel pour le compte du responsable du traitement*";

- ***Tiers***: "*la personne physique ou morale, l'autorité publique, le service ou tout autre organisme autre que la personne concernée, le responsable du traitement, le sous-traitant et les personnes qui, placées sous l'autorité directe du responsable du traitement ou du sous-traitant, sont habilitées à traiter les données*";

- ***Destinataire:*** "*la personne physique ou morale, l'autorité publique, le service ou tout autre organisme qui reçoit communication de données, qu'il s'agisse ou non d'un tiers; les autorités qui sont susceptibles de recevoir communication de données dans le cadre d'une mission d'enquête particulière ne sont toutefois pas considérées comme des destinataires*";

- ***Consentement de la personne concernée*** : "*toute manifestation de volonté,*

libre, spécifique et informée par laquelle la personne concernée accepte que des données à caractère personnel la concernant fassent l'objet d'un traitement".

Dans ce que suit nous résumons les principes de cette présente directive :

1. *Principe de licéité et de loyauté* : les états membres doivent traiter les données à caractère personnel d'une façon licite est loyale (Article 6). Le traitement est licite lorsqu'il est effectué en vue de protéger un intérêt essentiel à la vie de la personne concernée, la personne concernée a donné son ***consentement*** explicite, cela est nécessaire à la conclusion ou l'exécution d'un contrat liant la personne concernée dans le cadre des obligations légales, ou pour l'intérêt public. Le traitement est loyal lorsque les personnes concernées sont informées des traitements de leurs données à caractère personnel et également des finalités de la collection.

2. *Principe de finalité* : les finalités des traitements doivent être légitimes, explicites et déterminées lors de la collecte des données. Aucun traitement ultérieur n'est autorisé si cela est incompatible avec les finalités spécifiées au départ. Un traitement ultérieur à des fins historiques, statistiques ou scientifiques n'est pas réputé incompatible pour autant que les états membres prévoient des garanties appropriées. La durée de la conservation des données personnelles, sous forme permettant l'identification des personnes concernées, ne doit pas excéder la durée nécessaire à la réalisation des finalités pour lesquelles elles sont collectées. Cette durée peut être prolongée au delà de la période précitée à condition que les Etats membres prévoient des garanties appropriées (article 6).

3. *Principe d'exactitude* : Les données à caractère personnel doivent être adéquates, pertinentes et ***non excessives*** (respect du principe de ***need to know***) au regard des finalités pour lesquelles elles sont collectées. Elles doivent être exactes et mises à jour. Des mesures de rectification et destruction doivent être prises pour les données incomplètes et inexactes (article 6).

4. *Principe de droit d'accès* : la personne concernée a le droit d'obtenir du responsable de traitement la confirmation que ses données font, ou non, l'objet d'un traitement. Si ses données font l'objet d'un traitement, la personne concernée a le droit d'obtenir les informations concernant les circonstances de ce traitement et de connaître les catégories de destinataires auxquels les données sont communiquées (Article 12). La personne a, également, le droit de rectification, d'effacement et le verrouillage de ses données personnelles lorsque le traitement n'est pas conforme aux dispositions de la directive.

5. *Principe d'information et de **consentement** de la personne concernée **:*** les mesures d'information des personnes concernées par un traitement des données à caractère personnel doivent être prises. La personne reçoit toutes les informations concernant la finalité de la collecte, la communication de ses données à des tiers et son droit

d'opposition (Article 10 et 11)… L'un des principes relatifs à la légitimation des traitements de données est basé sur le ***consentement*** de la personne concernée tel que l'article 7 le mentionne "*si la personne concernée a indubitablement donné son consentement*". Le consentement de la personne est requis dans le cas des catégories particulières de traitement comme, par exemple, le traitement des données relatives à la santé : "*lorsque la personne concernée a donné son consentement explicite à un tel traitement*". Par conséquent, la personne concernée a le droit de s'opposer, pour des raisons légitimes, à ce que ses données fassent objet d'un traitement (article 14).

6. *Principe de non discrimination* **:** la directive protége toutes les données pouvant engendrer des discriminations tel l'article 8 le mentionne "*les états membres interdisent tout traitement des données à caractère personnel portant sur des catégories particulières de données qui révèlent l'origine raciale ou éthique, les opinions politiques, les convictions religieuses ou philosophiques, l'appartenance syndicale, vie sexuelle*". En plus des données personnelles pouvant engendrer des discriminations illégitimes, les états membres interdissent également le traitement des données relatives à la santé.

7. *Principe de dérogation* **:** la directive prévoit des dérogations aux principes 5 et 6 lorsque la personne concernée a donné son consentement[8] (article 8), le traitement est nécessaire à la défense vitale de la personne concernée, l'intérêt public important, pour sauvegarder la sécurité public(article 13), recherche scientifique… Une dérogation a été donnée au traitement des données de santé lorsque cela est nécessaire aux fins de la médecine préventive, des diagnostics médicaux, de l'administration de soins ou de la gestion de services de santé. Bien évidemment, quand ce traitement est effectué par un praticien de santé qui est soumis à une obligation de secret professionnelle équivalente.

8. *Principe de sécurité du traitement* : le responsable de traitement doit assurer la sécurité des données contre toute destruction accidentelle et accès non autorisé tel que l'article 17 le mentionne : "*les états membre prévoient que le responsable du traitement doit mettre en œuvre les mesures techniques et d'organisation appropriées pour protéger les données à caractère personnel contre la destruction accidentelle ou illicite, la perte accidentelle, l'altération, la diffusion ou l'accès non autorisés, notamment lorsque le traitement comporte des transmissions de données dans un réseau*…". Si le traitement est effectué en compte du responsable du traitement, ce dernier doit choisir un sous traitant de confiance qui assura la sécurité technique et organisationnelle relatives au traitement.

9. *Principe de notification* : le responsable de traitement (ou son représentant) doit adresser à une autorité de contrôle une notification avant la mise en œuvre d'un

[8] Sauf si l'interdiction ne peut pas être levée par le consentement de la personne selon la loi.

traitement (Article 18). Elle contient un ensemble d'information comme, par exemple, le nom et l'adresse du responsable (ou son représentant), les finalités du traitement…etc (Article 19). Des dérogations à ce principe sont prévues par la directive (Article 18).

10. *Transfert des données à caractère personnel vers des pays tiers* : le transfert de ces données ne peut se faire uniquement si le pays tiers assure une protection adéquate (Article 25). Des dérogations à cette obligation sont prévues par la directive, par exemple, lorsque la personne concernée a donné son consentement à un tel transfert et lorsque le transfert est nécessaire à la sauvegarde de l'intérêt vital de la personne concernée (Article 26)…

11. *Champ d'application* **:** la directive s'applique d'après l'article 3 ainsi rédigé " *au traitement de données à caractère personnel, automatisé en tout ou en partie, ainsi qu'au traitement non automatisé de données à caractère personnel contenues ou appelées à figurer dans un fichier*". La directive ne s'applique pas au traitement des données à caractère personnel lorsque il s'agit de la sûreté de l'état et les activités de l'état relatives à des domaines du droit pénal.

12. *Principe de sanction* **:** la directive prévoit des sanctions pour tout traitement non loyal et illicite.

2.1.3 Législative américaine : Privacy Act 1974

La protection des données à caractère personnel aux États-Unis est assurée par la loi ***Privacy Act 1974*** [94], loi protégeant les citoyens face aux abus informatiques. Elle est équivalente à la loi du 6 janvier 1978 en France. Cependant, le *Privacy Act* en 1974 est destiné à protéger les individus de la communication de leurs données personnelles par des organes de l'état, surtout fédéral. Aucune loi fédérale ne réglemente la collecte, l'utilisation des données personnelles par le secteur privée. Le *Federal Trade Commission* est le seul organisme ayant autorité sur ce type d'activités. Il veille à l'application des lois, mais ne peut pas en créer de nouvelles. Les défenseurs de la vie privée demandent depuis longtemps que le secteur privé soit soumis à des directives concernant le traitement des informations personnelles, sanctionnées par une loi. Une exception a été faite en 1998 avec le *Children Online Privacy Protection Act (COPPA)* [109], rendu applicable le 20 avril 2000, qui concerne la protection des enfants de moins de 13 ans. Toute personne ou service opérant sur le Web, proposant tout service Internet à des enfants, ou collectant des données sur eux, doit rédiger une *notice* sur sa politique de *privacy*. Il doit préciser l'objet de ses propositions, la diffusion des données, comment prendre contact avec un agent responsable, et surtout adresser ces notices aux parents et obtenir leur ***accord*** avant de collecter ou diffuser des données sur les enfants. C'est le *Federal Trade Commission*[9] qui assure l'information et

9 *(www.ftc.gov/opa/2000/07/coppacompli.htm)*

l'éducation du public, en relation avec le département de l'éducation, et qui en surveille l'application.

Le privacy Act 1974 affirme par force le principe du ***consentement*** de la personne en le mentionnant explicitement dans une de ses règles appelée ***"No Disclosure Without Consent" Rule***. Cette règle représente le cœur de toute divulgation aux tiers. Comme autre législation des exceptions à cette règle sont prévues par le Privacy Act. Ce dernier identifie douze règles d'exception et parmi ces exceptions le principe de ***need to know***.

Récemment, le Privacy Act 2005 [95] a été introduit devant le Congrès américain le 24 janvier 2005. Le champs d'application du « Privacy Act » est élargi puisque, en 1974, seules les administrations fédérales entraient dans son champ d'application. Aujourd'hui, le texte a vocation à réguler, non seulement les activités des administrations, mais encore celles des acteurs privés.

2.2 Protection des données à caractère personnel dans le domaine de la santé

"Il ne s'agit pas de faire le bien du patient mais de respecter sa liberté, sa dignité d'être qui décide pour ce qui le concerne" [70]. En 1964, la Déclaration d'Helsinki [43], adoptée sur l'initiative de l'Association médicale mondiale, constitue une déclaration de principes éthiques dont l'objectif est de fournir des recommandations aux médecins et autres participants à la recherche médicale sur des êtres humains. Tout comme les interventions médicales faites sur les êtres humains respectent certains principes, le traitement des données médicales à caractère personnel doit également suivre certains principes et recommandations. Comme les données médicales ne sont pas des données comme les autres, leur contenu informationnel relève de la sphère la plus intime de la vie privée et leur révélation peut être lourde de conséquence pour leur sujet. A titre d'exemple, il est facile d'imaginer l'intérêt que peut porter un employeur à l'état de santé de son personnel lors du recrutement, non pour souci de santé public, mais plutôt pour des considérations plus matérialistes.

Dans ce qui suit, nous faisons le point sur les législations relatives aux données médicales à l'échelle européenne et américaine.

2.2.1 Législation européenne

Le conseil d'Europe, a révisé sa recommandation n°R(81) relative à la recommandation applicable aux banques de données médicales automatisées [103], ce qui a donné naissance à une autre recommandation en 1997 relative à la protection des données médicales [102]. Les données médicales sont définies comme étant "*toutes les données à caractère personnel relatives à la santé d'une personne. Elles se réfèrent également aux données ayant un lien manifeste et étroit avec la santé, ainsi qu'aux données génétiques*" [102]. La directive

européenne sur la protection des données personnelles de 1995 classifie les données médicales dans la catégorie des données sensibles et commence par tout interdire. L'article 8 mentionne "*les états membres interdisent le traitement des données à caractère personnel qui révèlent l'origine raciale ou ethnique, les opinions politiques, les convictions religieuses ou philosophiques, l'appartenance syndicale, ainsi que le traitement des données relatives à la* ***santé*** *et à la vie sexuelle*". Par contre, la Directive prévoit des dérogations concernant le traitement de ces données lorsqu'un motif d'intérêt public important le justifie, particulièrement, pour assurer la qualité et la rentabilité concernant la procédure pour régler les demandes de prestations et les services dans le régime de l'assurance maladie.

Les traitements de ces données doit être faits par une personne (praticien de la santé ou non) soumise au secret professionnel. Si la personne est dans l'incapacité d'agir en son propre nom, le ***consentement*** de la personne légale est requis. Des dérogations existent dans des cas extrêmes, comme par exemple, en cas d'urgence, les données médicales peuvent être collectées et traitées avant d'informer la personne concernée. Concernant la recherche scientifique, les données utilisées doivent être anonymes et les organisations professionnelles et scientifiques doivent promouvoir le développement des techniques d'anonymisation. Si l'anonymisation constitue un obstacle pour la recherche scientifique, ces recherches peuvent être effectuées avec des données à caractère personnel uniquement si elles remplissent certaines conditions, par exemple avoir le ***consentement*** de la personne concernée.

2.2.2 Législation Américaine : HIPAA

La protection des données médicales est assurée aux Etats unis par HIPAA (*Health Insurance Portablity and Accountability*). Cette loi a été adoptée par le congrès en 1996, elle a pour but de définir un standard national pour les échanges électroniques des données médicales [67]. Après différentes modifications, DHHS[10] (the U.S. Department of Health and Human Services) publie les nouvelles règles de HIPAA (HIPAA Privacy rule) [67]. Ces règles sont rentrées en vigueur le 14 Avril 2003. Une des modifications importantes se situe au niveau du droit d'accès au dossier médical. Actuellement, HIPAA donne le droit à tous les patients d'accéder, de copier, et de demander des rectifications de leur dossier médical. Notamment, HIPAA exige les médecins et les hôpitaux à respecter le principe de finalité ***need-to-know*** lors d'une divulgation d'informations aux tiers, ce qui est appelé le ***minimum nécessaire***.

Contrairement, aux autres législations européennes et nationales, HIPAA distingue le consentement et l'autorisation. Une autorisation[11] est un document détaillé qui donne la permission aux entités couvertes[12] par HIPAA d'utiliser les informations médicales personnelles à des fins autres que le traitement, le paiement, et de donner la permission de

[10] Il s'occupe d'écriture des règles sur la confidentialité (privacy)
[11] Contient les éléments à divulguer, la durée de conservation…
[12] Professionnel de soins, assurances maladies, les services de facturation, associées d'affaires…

divulguer ces informations aux tiers. Par contre, le consentement est moins formel. L'entité couverte peut, mais pas obligatoirement, demander le consentement du patient pour utiliser ou divulguer ses informations, par exemple, pour le paiement. Des exceptions aux autorisations existent lorsque la divulgation et le traitement sont prévus par la loi (l'intérêt et santé publics, à des fins de recherches…). Le patient ne peut donc pas s'opposer aux traitements de ses informations médicales. D'un autre coté, le patient peut quand même refuser d'être affiché dans le répertoire de l'hôpital[13], ou que nous utilisons ses informations médicales pour des raisons commerciales[14].

3 Le Dossier Médical Personnel

En France, une étape significative et importante a été franchie par l'adoption, le 13 août 2004, de la loi relative à l'assurance maladie [82]. Cette loi a pour objectif principal la réorganisation de l'assurance maladie et la gestion des dépenses de santé. Parmi les mesures envisagées afin de diminuer les dépenses considérables dans ce secteur et d'améliorer la qualité des soins, est la création du ***dossier médical personnel,*** appelé ***DMP***. D'après le bulletin de l'ordre des médecins [21], le DMP " *est une application informatique qui permet de collecter des données individuelles de santé auprès des professionnels, de conserver ces données dans un lieu* ***sécurisé****, de* ***protéger ces données*** *et de* ***gérer des droits d'accès****, enfin de mettre ces données à la disposition des personnes habilitées, simplement et rapidement*".

La centralisation du DMP permet aux professionnels de santé de travailler davantage en collaboration et en réseau et il évite des examens redondants. Quant aux patients, ils deviennent les acteurs de leur propre santé, en d'autres termes, ils sont les propriétaires de leur dossier médical. Cette propriété a été affirmée, initialement, par la loi n° 2002-303 du 4 mars 2002 relative aux droits des malades et à la qualité du système de santé [81]. Il est à noter que le gouvernement a choisi d'utiliser l'expression « *dossier médical personnel* » plutôt que « *dossier médical partagé* » afin de renforcer la propriété de ce dossier médical. Le gouvernement vise à souligner qu'il s'agit avant tout du dossier du ***Patient*** [45]. Avant cette loi de 4 mars 2002, l'accès d'une personne à son dossier médical ne pourrait se faire que par l'intermédiaire d'un médecin choisi.

L'idée de mettre en place un dossier médical informatisé, en France, pose des questions pertinentes sur la confidentialité des informations hautement sensibles contenues dans ce dossier. Ces données sensibles sont des données à caractère personnel encadrées juridiquement par un ensemble de législations. Comme XML devient de plus en plus un standard d'échange d'information médicale (HL7) [69], il devient nécessaire de vérifier que les modèles de contrôle d'accès présentés dans la littérature permettent d'exprimer en pratique les principes édictés par les législations. Afin de mieux comprendre ce qu'est le

[13] Par exemple, victime d'une violence domestique, célébrités…
[14] Le terme commercial n'est pas encore bien spécifié.

DMP, nous commençons par décrire le contenu XML du DMP, puis nous abordons les problèmes de confidentialité. Dans une troisième partie, nous donnons un exemple de politique de contrôle d'accès mettant en évidence les carences des modèles de contrôles d'accès XML.

3.1 La mise en place du DMP

La directive européenne 95/46/CE précise que le contenu du fichier doit être structuré selon des critères déterminés relatifs aux personnes permettant un accès facile aux données à caractère personnel. En France, la constitution des éléments de base du dossier médical d'un patient est précisée par l'article R. 1112-2 du code de la santé publique.

Le dossier médical est composé (i) d'une identification du patient, (ii) d'un ensemble d'informations formalisées recueillies lors de la consultation externe dispensée dans un établissement, lors de l'accueil au service des urgences ou au moment de l'admission (e.g, ***consentement***), (iii) d'un ensemble d'informations formalisées établies à la fin du séjour (e.g prescription), (iv) d'un ensemble d'informations mentionnant quelles informations ont été recueillies auprès de tiers n'intervenant pas dans la prise en charge thérapeutique, etc…

Afin d'illustrer la problématique, nous avons choisi de ne pas représenter tous les éléments de base du dossier médical, mais plutôt de raisonner sur une abstraction des dossiers médicaux. Actuellement, dans le domaine médical, il n'existe pas un exemple de référence qui représente la réalité. Car la réalité est très diverse. Dans ce contexte, nous présentons un exemple qui est simplement une abstraction qui permet d'illustrer où peuvent apparaître les problèmes. Cet exemple n'a pas vocation de représenter la réalité. Une telle abstraction des dossiers médicaux est présentée dans la Figure 14.

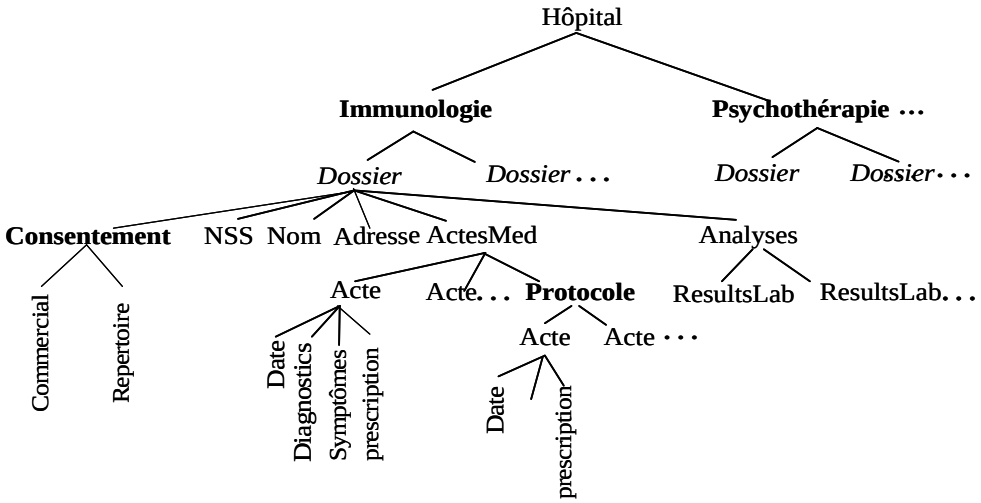


Figure 14: Dossiers Médicaux

Les dossiers médicaux des patients sont présentés au format ***XML***. Ils sont organisés en

service (e.g *Immunologie*, *Psychothérapie*). Chaque dossier est composé d'un ensemble de données administratives (numéro de sécurité social (NSS), nom, adresse) et de données médicales (actes médicaux, analyses). Les actes médicaux (ActesMed) sont décomposés, à leurs tour, en deux types d'actes médicaux : i) les actes médicaux normaux et ii) les actes médicaux sous protocole. D'après l'article R. 1112-2 du code de la santé publique, le consentement du patient est un élément de base du dossier médical. Nous l'intégrons donc comme un *élément* à part entière du dossier XML.

3.2 La confidentialité du DMP

La Commission Nationale Informatique et Libertés (CNIL) rappelle dans son avis consultatif rendu le 10 juin 2004 que "*le DMP doit être tenu dans le respect du secret médical* " et que les accès à ce dossier doivent être encadrés. Le patient doit avoir un accès automatique à son dossier médical. Quant aux professionnels de santé, ils peuvent échanger des données de santé uniquement si la personne a donné son consentement : "*deux ou plusieurs professionnels de santé peuvent toutefois, **sauf opposition de la personne dûment avertie,** échanger des informations relatives à une même personne prise en charge, afin d'assurer la continuité des soins ou de déterminer la meilleure prise en charge sanitaire possible*" (Art. L.1110-4 de la loi du 4 mars 2002). De même, si *"le praticien qui a prescrit l'hospitalisation demande la communication du dossier, cette communication ne peut intervenir qu'après **accord** du patient*". Cependant cette restriction peut être levée en cas d'urgence; l'accès au dossier devient alors automatique afin de permettre le traitement rapide du patient.

Des interdictions d'accès strictes au dossier médical personnel ont été explicitement introduites dans le cadre de la médecine du travail et lors de la signature des contrats type assurance même avec le consentement de la personne concernée (Art. L. 161-36-3 de loi relative à l'assurance maladie).

Lorsque l'accès est autorisé, la loi 4 mars relative aux droits des malades dans son article 6 confirme le respect du principe de finalité ou ***need-to-know*** qui consiste à retourner uniquement l'ensemble d'informations nécessaire pour accomplir une tâche de travail : "*Les praticiens-conseils du service du contrôle médical et les personnes placées sous leur autorité n'ont accès aux données de santé à caractère personnel que si elles sont strictement nécessaires à l'exercice de leur mission, dans le respect du secret médical*".

3.3 Politique de contrôle d'accès et les carences des modèles existants

Dans cette section, nous décrivons notre politique de contrôle d'accès dont les règles ont été motivées par la lecture des textes législatifs notamment de HIPAA. Cette politique comprend trois règles d'autorisation importantes de l'hôpital et est représentative des principes de ***consentement*** et de ***need-to-know***. Pour chacune des règles d'autorisation nous

montrons comment les modèles de contrôle d'accès présentés au chapitre précédent traite la règle.

Les dossiers médicaux sont partagés entre plusieurs utilisateurs (patients, professionnels de santé, laboratoire médical et compagnies d'assurance) ayant des objectifs et des tâches de travail différents.

1. *Règle 1* : *"cacher au groupe d'annuaire*[15] *le nom du service où les patients sont traités pour ceux qui n'ont pas donnés leur consentement"*.

La première règle d'autorisation de l'hôpital dit que le patient (e.g ; victime d'une violence domestique, célébrités…) a le droit de cacher le service dans lequel il est traité tout en appartenant au répertoire du l'hôpital [67]. Cette règle vise, donc, à respecter la vie privée et la liberté individuelle des patients, en occurrence, respecter le principe de consentement imposé par les législations.

L'effet de cette règle d'autorisation sur le document XML présenté dans la Figure 14 devrait être de rattacher l'élément dossier de patient en question à un service dépersonnalisé (e.g élément avec un tag anonyme) tout en gardant les autres dossiers médicaux non affectés. Cette opération est appelée "*dépersonnalisation des ancêtres*". La restructuration devrait se faire de telle sorte à empêcher l'inférence de la classification initiale. Par conséquent, la vue retournée devrait, en principe, être comme suit (Figure 15) :

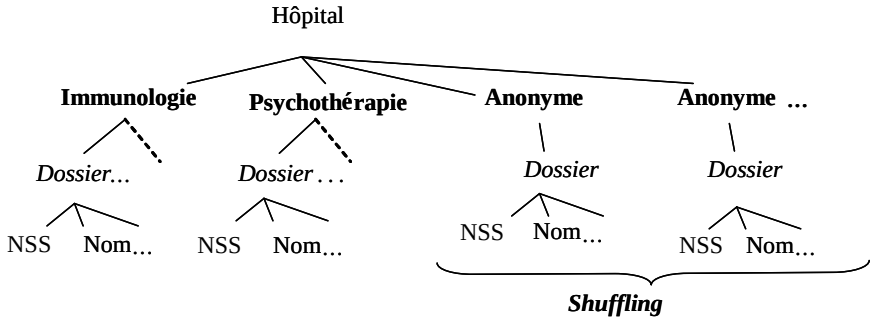


Figure 15 : Dépersonnalisation des ancêtres

Dans ce qui suit nous montrons comment les modèles proposés dans la littérature réagissent à la règle d'autorisation Règle 1 :

1. Les modèles de Bertino [8], Gabillon [54], Murata [85], Wang [118] et Zhang [119] ne permettent pas d'autoriser l'accès à un nœud dans un sous-arbre inaccessible. Ce qui veut dire que, si un nœud donné est inaccessible, tout le sous-arbre porté par ce nœud sera inaccessible. La seule possibilité donnée par ces modèles est de cacher complètement le dossier médical du patient concerné en définissant une règle

[15] Les utilisateurs responsables de la gestion d'annuaire de hôpital.

d'autorisation négative sur l'élément dossier. La vue retournée dans ce cas est schématisée dans la Figure 16.

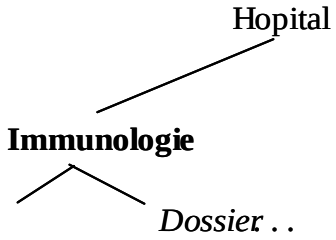


Figure 16 : La vue retournée

Nous remarquons que cela ne répond pas à la règle 1 car cela supprime définitivement le dossier médical du patient. Par conséquent, la présence du patient dans l'hôpital est également cachée. Ce n'est pas l'objectif de la règle d'autorisation *Règle 1*. Par conséquent, ces modèles ne respectent pas le ***consentement*** du patient.

2. Les modèles de Damiani [37, 38] et de Kudo [75] autorisent l'accès à un nœud dans un sous-arbre inaccessible. Ils permettent donc de définir une règle d'autorisation négative sur l'élément service (e.g, *immunologie*) et une autre règle positive sur l'élément dossier. Par contre, afin de garder la structure initiale du document XML, ils révèlent tous les éléments ancêtres[16] du nœud (descendant) autorisé. La vue retournée dans ce cas est schématisée dans la Figure 17.

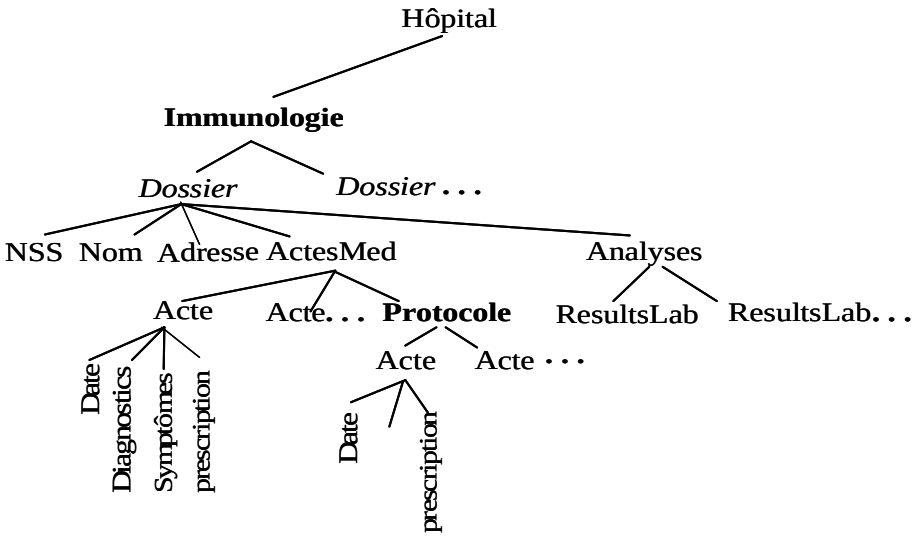


Figure 17 : La vue retournée dans les modèles Damiani et Hada

D'après le schéma ci-dessus, les modèles révèlent l'élément qui devait être caché (le

[16] Ils révèlent uniquement le tag des éléments ancêtres. Les attributs de ces éléments sont supprimés s'ils existent.

nom du service dans lequel le patient est traité) pour permettre de garder la structure du document initial. Ces modèles ne répondent pas, également, au consentement du patient.

3. Les modèles de contrôle d'accès de Wei Fan [46] et Gabillon [55, 56] résolvent le problème de révélation des tags des ancêtres du nœud autorisé en les renommant par un tag *anonyme*. Ces modèles donnent donc la possibilité de définir une règle d'autorisation négative sur l'élément service (*e.g, immunologie*) et une autre règle positive sur l'élément *dossier*. La vue retournée est schématisée dans la Figure 18.

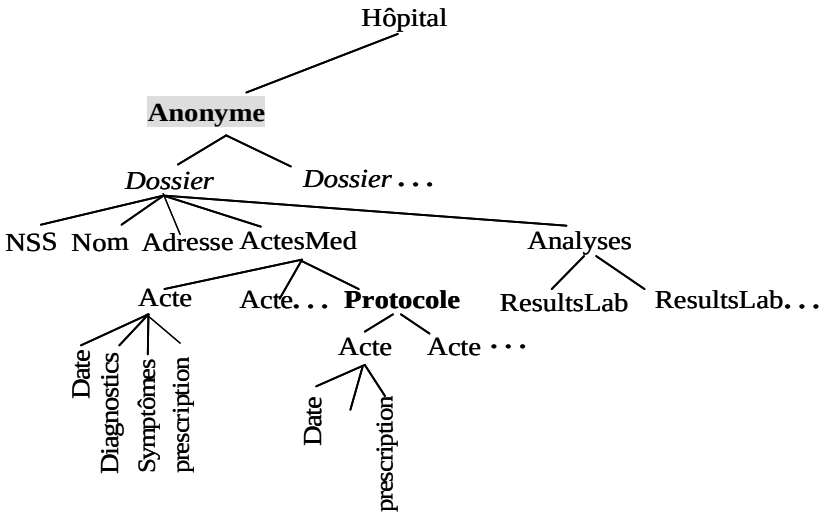


Figure 18 : La vue retournée dans les modèles Wei Fan et Gabillon

Bien que les modèles cachent le nom du service en question (*e.g, immunologie*), ils ne répondent pas à la règle d'autorisation pour les raisons suivantes :

- Le renommage du service s'applique à tous les dossiers médicaux des patients de ce service (sans exception) même pour les patients qui n'ont pas donné leur consentement. Cela viole le principe du respect de la *liberté individuelle* imposé par la convention européenne des droits de l'Homme.

- Le fait que les dossiers des patients restent classés ensemble, des problèmes d'inférence liés à cette classification peuvent survenir. Par exemple, si une personne connaît un des patients de ce service alors elle peut inférer le nom du service de tous les patients appartenant à cette classe.

2. *Règle 2* : "*cacher pour les pharmaciens le fait que certaines prescriptions sont administrées dans le cadre d'un protocole*".

La deuxième règle d'autorisation indique que le pharmacien n'a pas le droit de savoir si certaines prescriptions sont administrées via un protocole. Cela veut dire qu'il ne faut pas révéler plus d'information nécessaire à accomplir sa tâche du pharmacien; en d'autres termes, il faut respecter le principe de **need-to-know** imposé par les législations.

L'effet de cette règle d'autorisation consiste à supprimer le nœud *protocole* du dossier du patient. Ensuite, de rattacher tous les éléments *Acte* au dessous du protocole comme des descendants directs de l'élément *ActesMed* pour qu'ils rejoignent la classe des Actes médicaux ordinaires. Cette opération est appelée la "*réduction de chemin*". La vue retournée devrait être comme suit.

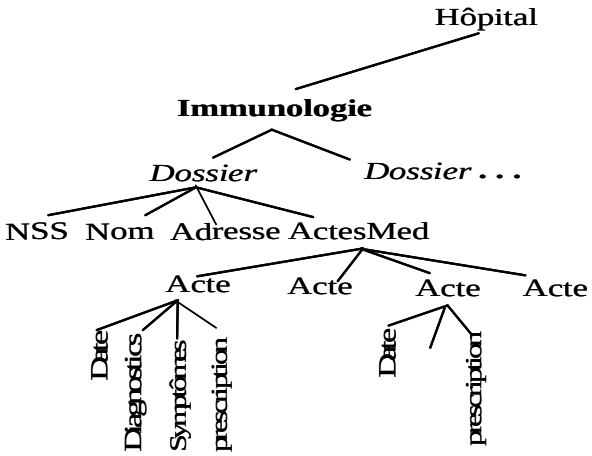


Figure 19 : Réduction de chemin

Tout comme dans le cas de la première règle d'autorisation, les modèles de contrôle d'accès existants échouent à répondre à cette deuxième règle d'autorisation[17]. Ici le renommage de l'élément *protocole* proposé par Wei Fan [46] et Gabillon [55, 56] est inutile car la présence du nœud lui même révèle une information qui amène à faire la différence entre les types des actes médicaux (voir la Figure 20). Cette information, le pharmacien ne doit pas la connaître pour accomplir sa tâche. Cela viole donc le principe de ***need-to-know***.

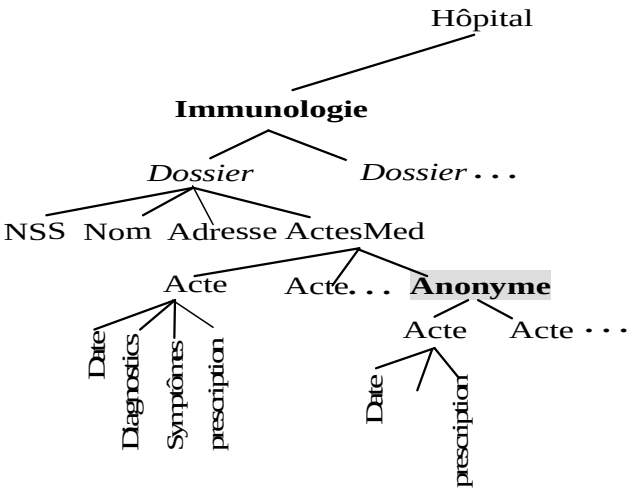


Figure 20 : Vue retournée dans les modèles Wei Fan et Gabillon

[17] Les modèles Wei fan [46]et al et C.Stoica [108] propose une autre alternative sous des contraintes spécifiques sur le schéma du document.

3. *Règle 3* : "*cacher aux laboratoires médicaux la corrélation entre les informations médicales et les informations administratives pour chaque dossier*".

Que le patient le veuille ou pas, il n'est pas la seule personne à décider de la gestion de ses données médicales. En effet, pour des raisons d'intérêt public, les législations autorisent la transmission des données médicales des patients à des fins de recherches scientifiques. Par contre, pour tout traitement avec un objectif commercial, la loi HIPAA exige le consentement du patient. Dans cette situation les données médicales doivent être accessibles pour répondre au principe de ***need-to-know*** concernant les recherches scientifiques et les données administratives (nom, adresse) doivent être accessibles pour répondre au principe de consentement[18]. Le patient souhaite donc donner son consentement pour recevoir des offres commerciales mais il veut cacher, exactement, quelles sont les données médicales qui le concernent. Le résultat de cette règle d'autorisation est schématisé dans la Figure 21.

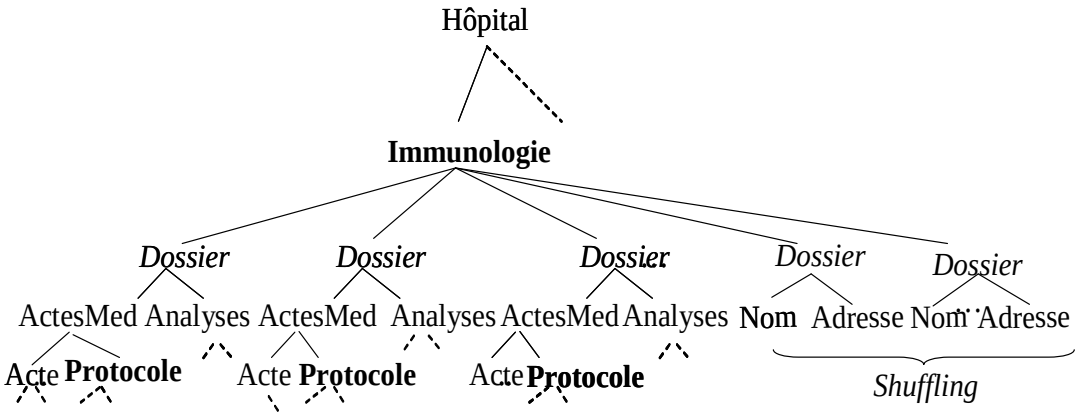


Figure 21 : Décorrelation

Les modèles de contrôle d'accès ne peuvent répondre à cette règle d'autorisation car cela imposerait de définir deux bases de règles d'autorisation pour le même document et pour le même utilisateur (e.g, laboratoire médical). Par conséquent, l'utilisateur obtiendra deux vues séparées du même document XML. Par contre, la conjonction de ces deux vues peut permettre d'inférer la structure initiale du document en comparant l'ordre des éléments dans les deux vues.

En résumé, bien que la sémantique du contrôle d'accès des modèles existants diffère d'un modèle à un autre, tous les modèles proposés imposent une contrainte forte concernant la structure du document XML. La structure de la vue retournée doit être compatible avec la structure du document initial. Cela amène à ce que tous les descendants d'un nœud se comportent de la même façon par rapport à leurs ancêtres.

La politique de contrôle d'accès que nous venons de décrire montre clairement les

[18] Bien évidement il y a ceux qui consentirent pour des buts commerciaux et d'autres ne donnent pas leur consentement.

carences des modèles de contrôle d'accès existants pour répondre aux principes de ***consentement*** et de ***need-to-know*** et montre la nécessité de mieux prendre en compte les associations ancêtre-descendant et les associations de fraternité dans l'expression des règles d'autorisation.

3.4 Infrastructure du DMP

La loi du 4 mars 2002 stipule que le DMP peut être créé auprès d'un hébergeur, appelé *hébergeur de données de santé à caractère personnel* [81], qui ne peut intervenir qu'après le consentement exprès de la personne concernée (*Art. L.1111-8* de loi 4 mars 2002). Les hébergeurs seront désignés au terme d'un appel d'offres pour leur fiabilité, notamment pour les garanties apportées en matière de confidentialité des données. Ils doivent être agréés et les conditions d'agrément sont fixées par décret en Conseil d'état, pris après l'avis de Commission Nationale de l'Informatique et des Libertés (CNIL) et des conseils de l'ordre des professions de santé, ainsi que du conseil des professions paramédicales. Ce décret mentionne les informations à fournir lors de la demande de l'agrément, notamment les dispositions à prévoir pour garantir la sécurité des données traitées en particulier les mécanismes de contrôle et de sécurité dans le domaine de l'informatique. L'hébergement doit être réalisé avec le respect du secret professionnel et la prestation d'un tel hébergement fait l'objet d'un contrat. La loi du 4 mars 2004 prévoit, également, les obligations appliquées sur les hébergeurs en cas de retrait de l'agrément (*Art. L.1111-8*). L'hébergeur doit restituer les données qui lui sont confiées, sans en garder une copie, aux professionnels, établissements de santé ou à la personne concernée ayant contracté avec lui. En résumé, l'hébergeur des données de santé à caractère personnel doit être un *tiers de confiance*.

Dans le cas où nous n'avons pas confiance en l'hébergeur sur la sécurisation du dossier médical, nous ne pouvons ni déposer le dossier en clair, ni permettre à l'hébergeur de gérer les droits d'accès. Pour remédier à ce problème, d'autres mécanismes de sécurité basés sur le chiffrement peuvent être envisagés. L'utilisation des techniques de chiffrement dans la gestion des droits d'accès pour les documents XML fera l'objet de chapitre IV de cette thèse sur la sécurisation des documents XML basée sur le chiffrement.

4 Conclusion

Dans ce chapitre nous avons présenté, dans un premier temps, les textes juridiques encadrant la protection des données à caractère personnel à l'échelle international, européen et américain. Ensuite, nous avons fait le point sur les législations encadrant les données médicales. Ces législations édictent un ensemble de principes qui devront être respectés dans n'importe quel traitement de données personnelles. Nous nous sommes intéressés dans notre étude à deux principes fondateurs qui sont liés directement à la confidentialité des données personnelles dans le domaine de la santé, à savoir le ***consentement*** et le ***need-to-know***. Notre

étude a porté sur un exemple concret et important qu'est le dossier médical personnel (DMP). Cette étude nous a permis d'identifier les carences des modèles de contrôle d'accès existants vis-à-vis des deux principes précédemment cités et nous a motivé à définir un nouveau modèle de contrôle d'accès XML, qui est décrit dans le chapitre suivant.

Chapitre IV – Modèle de contrôle d’accès pour XML intégrant les associations

1 Introduction

Dans le chapitre précédant nous avons montré les limites des modèles de contrôle d’accès pour XML proposés dans la littérature, vis-à–vis de deux principes fondateurs de la protection des données à caractère personnel "*le consentement*" et "*le need to know*". Bien que différents modèles de contrôle d’accès (DAC, MAC et RBAC) ont été proposés pour sécuriser les documents XML, les efforts ont portés, plus particulièrement, sur la granularité de protection de plus en plus fine (DTD, document, élément, attribut), sur les politiques de propagation et de résolution de conflit [9,13, 37, 38, 54, 75, 78], sur la performance des algorithmes de contrôle [35,23, 85, 96, 97, 117], sur les canaux de communication et de distribution de l'information (dissémination sélective en mode Push ou Pull) [11, 12, 90, 114, 87] et sur la non corruption du modèle de contrôle d'accès par des techniques de chiffrement ou d'environnements d'exécution sécurisés [19]. Par contre, tous ces modèles focalisent leurs contrôles d’accès sur les nœuds (élément/attribut). Malheureusement, même la sémantique de contrôle d’accès sur les nœuds n’est pas clairement définie (c.f deuxième chapitre). Cela est lié fortement à la difficulté de définir la vue exacte qui amène au nœud autorisé. Aucun des modèles de contrôle d’accès existants ne s’est intéressé à la protection des associations existantes entre les nœuds. Ces associations dans le document XML ont été exploitées, uniquement, pour définir les politiques de propagation pour les règles d’autorisation. Pourtant, le problème est fondamental du point de vue de la préservation de la confidentialité et de l’intimité des données. Les associations ancêtre-descendant et les associations de fraternité dans un document sont, également, porteuses d’information potentiellement sensible.

Par conséquent, ne pas tenir compte des associations dans la définition d’un modèle du contrôle d’accès amène à deux problèmes fondamentaux :

- *Divulgation de la classification* : la structure d'un document XML révèle certaines classifications[19] (e.g. les différents services d'un hôpital dans lesquels les patients sont traités, les types d'activités ou départements de vente d'une compagnie, une

[19] Synonyme des catégorisations

répartition socio-économique). C'est pourquoi l'appartenance d'un nœud à un sous-arbre véhicule par défaut sa classification. Bien que l'on puisse cacher l'information portée par la classe, elle n'en demeure pas moins sensible aux attaques statistiques. En effet, souvent la cardinalité d'une classe peut en révéler sa nature. De plus, cacher l'appartenance d'un élément à une classe entraîne de le faire pour chaque élément de cette classe.

- *Filiation uniforme* : les règles d'autorisation exprimées sur les nœuds ancêtres donnent une seule vue autorisée de chemin qui amène à tous ses descendants. En d'autres termes, il n'y a aucun moyen de délivrer deux vues autorisées différentes du même ancêtre pour ses deux descendants. Par exemple, un patient souhaite cacher le service médical où il était traité, alors qu'un autre patient donne son consentement pour divulguer cette information.

Ces deux problèmes violent les deux principes fondateurs imposés par les législations relatives à la protection des données à caractère personnel qui sont le ***need-to-know*** et le ***consentement***. Par conséquent, il est indispensable de définir un modèle de contrôle d'accès qui traduise exactement ces principes de législation en pratique.

L'objectif de ce chapitre est de présenter notre modèle de contrôle d'accès qui vise, plus particulièrement, à protéger les associations dans un document XML. Dans la deuxième section, nous caractérisons les différentes associations qui doivent être protégées. Nous identifions les autorisations sur les associations fondamentales pour traiter les problèmes de divulgation de la classification et de filiation uniforme. Nous présentons, également, les mécanismes nécessaires et indispensables qui aident à traduire exactement les autorisations en vue autorisée "***le clonage et le shuffling*** ". Dans un second temps, nous décrivons notre modèle de contrôle d'accès qui prend en compte la protection des nœuds et les associations entre eux. La troisième section présente notre formalisme à base de règles pour exprimer les autorisations sur les associations ainsi qu'une politique de résolution de conflit pour résoudre les conflits qui peuvent exister entre elles. Nous discutons sur l'interaction des autorisations sur les associations avec les modèles de contrôle d'accès existants, plus particulièrement, avec les autorisations sur les nœuds. Plutôt que de proposer un tout nouveau modèle, l'approche proposée vise à étendre les modèles de contrôle d'accès existants pour XML avec de nouvelles règles sur les associations. Une conclusion fera l'objet de la quatrième section.

2 Caractérisation des autorisations sur les associations

Les modèles de contrôle d'accès existants interprètent la politique de contrôle d'accès comme étant un mapping entre le document source (ou `Source`) et la vue autorisée de ce même document (ou `Vue`) et font l'hypothèse que `Vue` $\subseteq$ `Source`. Plus précisément, les règles d'autorisation sélectionnent un sous-ensemble de nœuds `Source` qui vont participer dans la `Vue` résultat. Par effet de bord, tous les arcs ayant des nœuds extrémités supprimés

par une règle d'autorisation sont, à leur tour, supprimés de la Vue. Notre exemple de motivation du dossier médical, du chapitre précèdent, a montré clairement que la protection des associations nécessite de réviser cette hypothèse puisque la Vue peut être obtenue à partir d'une restructuration plus complexe de la Source. De nouveaux chemins et de nouveaux nœuds peuvent apparaître dans la Vue et l'ordre des nœuds peut être différent de celui de la Source afin d'éviter l'inférence [50, 51, 52].

Dans cette section, nous nous focalisons sur la sémantique des autorisations sur les associations et également, sur l'impact que chacune d'elle a sur la Vue résultante. Nous ne faisons aucune hypothèse ni sur la façon de définir cette Vue (XQuery ou système des règles), ni sur sa construction (matérialisation vs. streaming). Ces deux problématiques vont être discutées respectivement dans les sections 3 et 4.

2.1 Les notations

Nous présentons dans cette section, le formalisme que nous considérons pour le modèle des données du document XML, ainsi que les annotations associées qui vont être utilisées tout au long de ce chapitre. Un XML document *d* est défini sous la forme d'un n-tuple ($(Label_d, valeur_d, N_d, r_d, E_d, \phi_{label}, \lambda_{value}, \varphi_{order})$, où :

- $Label_d$: représente l'ensemble des labels d'élément (appelés aussi les tags) et les noms des attributs de type chaîne de caractère (string).
- $valeur_d$: représente l'ensemble des valeurs d'attribut/élément de type chaîne de caractère (string).
- N_d : $N^e_d \cup N^a_d$: l'ensemble des nœuds représentant les éléments et les attributs, respectivement. Chaque $n \in N^e_d$ possède un *label d'élément obligatoire* $\in Label_d$, ainsi qu'une *valeur d'élément optionnelle* $\in valeur_d$. Tandis que, chaque $n \in N^a_d$ possède un *label* d'*attribut obligatoire* $\in Label_d$, ainsi qu'une *valeur* d'*attribut obligatoire* $\in valeur_d$.
- r_d : un nœud particulier qui représente la racine du document.
- E_d : $E_d \subset (N_d \cup r_d) \times N_d$ est un ensemble d'arcs, où chaque $e \in E_d$ représente une association élément-sous-élément ou une association élément-attribut.
- $\phi_{label} : N_d \rightarrow Label_d$ correspond à la fonction d'étiquetage des nœuds.
- $\lambda_{valeur} : N_d \rightarrow valeur_d$ correspond à la fonction qui associe à chaque nœud sa valeur.
- $\varphi_{order} : N_d \rightarrow Integer$ est une fonction d'ordonnancement des nœuds basée sur un parcours préordre de l'arbre.

Les annotations Anc(n), Child(n), Desc(n) et Sibling *(n)* désignent, respectivement, l'ensemble des ancêtres, de fils, de descendants et de frères d'un noeud donné $n \in N_d$. *Parent*(*n*) désigne le noeud père du *n*. *Path*(n_1, n_2) désigne le chemin reliant le noeud n_1 au noeud n_2.

D'après ce modèle, un document XML peut être représenté sous forme d'un graphe étiqueté où les nœuds représentent les éléments/attributs et les arcs représentent les associations entre eux. Si un nœud donné n'a aucun parent, alors il sera relié implicitement à la racine du document. Notre exemple de motivation sur les dossiers médicaux (Figure 22), présenté dans le chapitre précédant, illustre ce modèle.

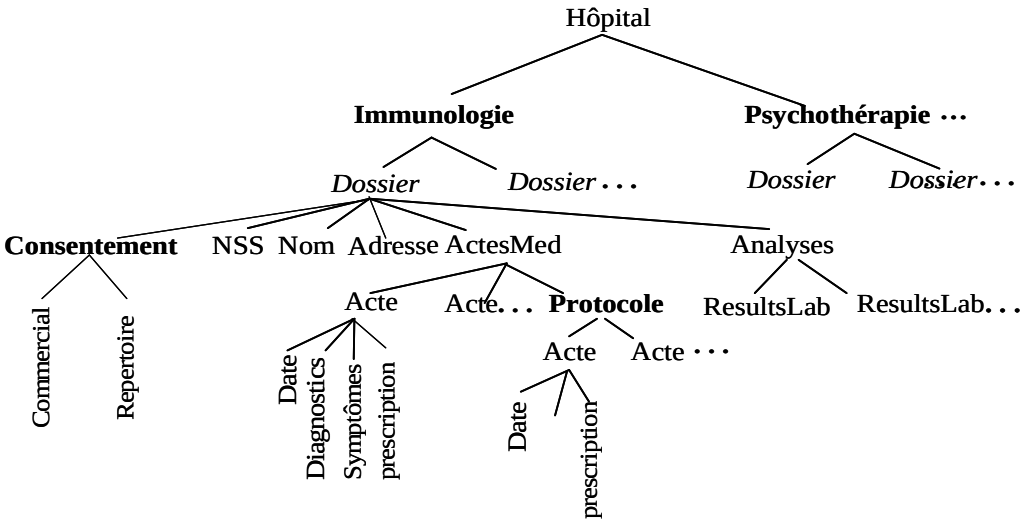


Figure 22 : Exemple des dossiers Médicaux

2.2 Les mécanismes de clonage et de shuffling

Prendre en compte le consentement de l'utilisateur dans les modèles de contrôle d'accès impose de cloner des nœuds et des chemins du document `Source` dans la `Vue` retournée [50, 51, 52].

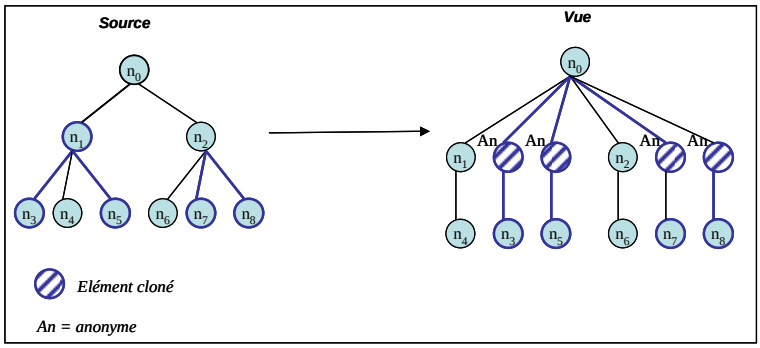


Figure 23 : Mécanisme de clonage

Fondamentalement, le clonage de nœud `Source` n_1 (voir la Figure 23) est obligatoire à chaque fois que deux de ses descendants autorisés n_3 et n_4 ont deux règles d'autorisation conflictuelles qui peut correspond, par exemple, à deux consentements différents. Cela veut dire que les deux descendants (n_3 et n_4) ont deux visions différentes par rapport à leur ancêtre (n_1). Par exemple, le noeud n_1, qui peut faire référence à l'élément service (immunologie ou psychothérapie) de notre exemple des dossiers médicaux (Figure 22), doit se dépersonnaliser pour le noeud n_3 qui peut correspond, par exemple, à un *patient VIP*. Par contre, ce nœud n_1 ne se dépersonnalise pas pour le noeud n_4, qui peut correspond à un patient qui a donné son consentement de révéler son nom du service. Cela représente exactement le comportement de notre première règle d'autorisation R_1 qui dit "*cacher au groupe du répertoire*[20] *le nom du service où les patients sont traités pour ceux qui n'ont pas donnés leur consentement*". Puisque un document XML est représenté sous forme d'arbre, chaque noeud participant dans le sous-chemin commun Path(*n1*,Parent(*n3*))∩Path(*n1*,Parent (*n4*)) va être, à son tour, cloné.

Le *clonage* est un mécanisme qui permet de dupliquer les chemins et les nœuds du document `Source` dans la `Vue`. Il est à noter que les nœuds feuilles (les éléments et les attributs terminaux) d'un document source ne font jamais l'objet d'un clonage. Dans ce qui suit, nous utilisons les termes *original* et *clone*(*s*) pour distinguer, dans la `Vue`, entre l'image originale d'un chemin ou d'un élément dans la `Source` et le(s) chemin(s) ou le(s) élément(s) obtenu(s) après l'opération du clonage. Nous présentons ci-dessous les deux formules concernant le mécanisme de clonage.

- ***Clonage-Elément*** : nous désignons par $\tilde{n}_i \in N^e_{View}$ le ith clone du nœud $n \in N^e_{Source}$. Les indices sont utilisés lorsque les différents clones d'un même nœud doivent être distingués. Sinon nous n'en tenons pas compte. Le label, la valeur et l'ordre d'un clone sont définis comme suit :

  $$\phi_{label}(\tilde{n}) \in \{\phi_{label}(n), \text{"anonyme"}\}^{21},$$

  $$\lambda_{value}(\tilde{n}) = \Delta, \text{ avec } \Delta \text{ dénote une chaîne de caractère vide,}$$

  $$\varphi_{order}(\tilde{n}) = \text{Shuffle}(\text{Sibling}(\tilde{n})).$$

  Où le *Shuffle* définit un ordre aléatoire entre les clones frères du même noeud. La nécessité de *Shuffling* sera expliquée par la suite.

- ***Clonage-Path*** : Soit u un $path(n_1, n_2, \ldots, n_k) / n_1, n_2, \ldots, n_k \in N^e_{Source}$ et $(n_1, n_2) \ldots (n_{k-1}, n_k) \in E_{source}$, $\tilde{u}$ représente le clone du u qui est défini par $(\tilde{n}_1,\tilde{n}_2 \ldots \tilde{n}_k) / \tilde{n}_1,\tilde{n}_2 \ldots \tilde{n}_k \in N^e_{View}$ et $(\tilde{n}1,\tilde{n}2) \ldots (\tilde{n}_{k-1},\tilde{n}_k) \in E_{View}$.

Le clonage est une condition préalable et indispensable pour cacher les associations ancêtre-descendant ainsi que les associations de fraternité entre les nœuds. Par contre, l'ordre des nœuds clonés, dans la `Vue`, doit être géré avec prudence pour éviter des problèmes

[20] Les utilisateurs responsables de la gestion d'annuaire de hôpital.

d'inférence de base. Par exemple, d'après la Figure 23, le noeud n_1 s'est dépersonnalisé, d'une façon indépendante, pour les noeuds n_3 et n_5, le noeud n_2 s'est dépersonnalisé, également, pour les nœuds n_7 et n_8. Mais, le fait que les nœuds clonés prennent, à chaque fois, une position très proche par rapport à leur nœud original, cela amène à avoir une structure régulière du document XML. Par conséquent, un utilisateur donné peut, inévitablement, deviner la structure initiale du document.

Pour illustrer ce problème, prenons la règle d'autorisation R_3 de notre politique de contrôle d'accès. Supposons que la `vue` est ordonnée de telle sorte que les instances des deux groupes (`ActesMed`, `Analyses`) et (`Nom, Adresse`) garde le même ordre relatif défini initialement dans le document `Source`. Dans ce cas, leur association de fraternité initiale, qui devrait être cachée par le clonage, est révélée par l'ordre des éléments (i.e., le i^{th} instance de (`ActesMed`, `Analyses`) correspond à la i^{th} instance de (`Nom, Adresse`)). Un problème similaire existe, également, avec la règle R_1 si les *clones* de l'élément service médical sont placés à proximité de leur *original* (e.g., les clones sont des frères directement à droite ou directement à gauche). Donc, l'opération de clonage n'a aucun sens sans mélanger les nœuds clonés entre eux pour éviter l'inférence basée sur l'ordre. Nous appelons cette opération de mélange le "*shuffling des nœuds*". Cela est illustré dans la Figure 24.

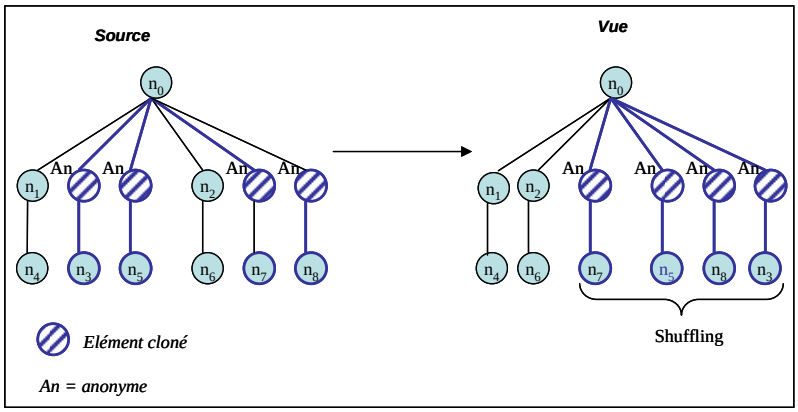


Figure 24 : Mécanisme de shuffling

- ***Shuffling des nœuds*** : est un processus récursif qui s'applique à chaque nœud de la `Vue` contenant un clone comme un nœud fils. Tous les clones, fils d'un nœud donné, sont mélangés ensemble pour éviter l'inférence. Pour un nœud donné, les fils clonés sont groupés après les fils originaux (par convention) et ensuite ils sont mélangés. L'ordre relatif des fils originaux doit être préservé dans la `Vue` puisque l'ordre des nœuds dans XML est important. Par conséquent, la fonction d'ordonnancement φ_{order} doit satisfaire les deux propriétés suivantes :
  - $\forall ni,nj \in N_{Source}$, $\forall ni',nj' \in N_{View}$ / $ni=ni'$ and $nj=nj'$, $\varphi_{order}(ni) < \varphi_{order}(nj) \Rightarrow \varphi_{order}(ni') < \varphi_{order}(nj')$

[21] Par défaut, le noeud cloné hérite le label de son original.

- $\forall ñ \in N_{View}$, $\varphi_{order}(ñ)$ = random(]max (φ_{order}(Parent(ñ)), φ_{order}(iops(ñ))),φ_{order}(ifol(Parent(ñ))[)

Où iops(n) (resp. ifol(n)) est le nœud frère qui précède (resp. suit) directement le nœud n.

2.3 Les autorisations sur les associations d'ancêtres

Les autorisations sur les associations sont introduites pour résoudre les deux problèmes identifiés dans l'introduction appelés la ***divulgation de la classification*** et la ***filiation uniforme*** [50, 51, 52]. Ces deux problèmes violent, comme nous l'avons montré dans notre exemple de motivation, les deux principes fondateurs de ***need to know*** et du ***consentement***. Dans ce qui suit, nous nous intéressons, dans un premier temps, à la divulgation de la classification. Ensuite, nous expliquons la filiation uniforme.

- ***La divulgation de la classification*** : l'objectif est de masquer l'appartenance d'un noeud descendant *n* à une classe donnée enracinée par un nœud ancêtre *a*[22]. À cet égard, deux situations doivent être distinguées : (1) cacher l'appartenance d'un nœud *n* à une classe donnée. Par conséquent, l'information à protéger est la classe du nœud auquel il appartient. (2) cacher le fait qu'un nœud donné *n* est classifié.

Dans le premier cas, l'information à protéger est celle qui révèle l'identification de la classe auquel le nœud *n* est appartient. Cette information est portée par le nœud ancêtre *a*, soit par son label, par un de ses attributs ou soit par un de ses éléments. Le clonage du nœud ancêtre *a* et le chemin qui amène au nœud *n* représente un moyen pour protéger cette information sensible. Bien évidement, le label de l'ancêtre *a*, les attributs et les sous-éléments qui ne participent pas dans l'opération de clonage peuvent, également, être anonymisés s'ils révèlent l'identification de la classe. Cela amène à définir une première classe d'autorisation appelée "***dépersonnalisation des ancêtres***". L'impact de cette première classe est représenté dans la Figure 25.

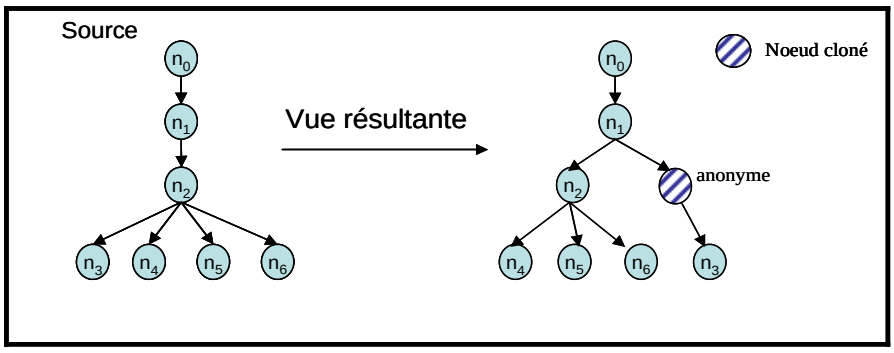


Figure 25 : Dépersonnalisation des ancêtres

[22] Dans la suite, l'appartenance à une classe est considerée comme sysnonyme de l'appartenance à un sous arbre.

La sémantique des autorisations "*dépersonnalisation d'ancêtre*" est définie, en terme de leur impact attendu sur la Vue, comme suit : soit $n \in N_{Source}$. Soit $a, u \in N^e_{Source}$ / $a \in Anc(n)$ et u = Parent(n).

- ***Dépersonnalisation des ancêtres***
  - ***Si*** $a = u$ ***Alors*** Clonage_Element (a) ***Sinon*** Clonage_Path (a, u);
  - Parent($\tilde{a}$) ← Parent(a);
  - Parent(n) ← $\tilde{u}$; // ***Si*** $a = u$ ***Alors*** $\tilde{a} = \tilde{u}$

Dans le deuxième cas, l'information qui doit être protégée est la présence de l'ancêtre a lui-même dans le chemin qui amène au nœud n. Cette situation apparaît à chaque fois qu'une appartenance à une classe révèle une exception par rapport à une situation générale. Par exemple, la règle R_2 de notre exemple de motivation qui dit "*cacher pour les pharmaciens le fait que certaines prescriptions participent à un protocole* " illustre ce point. Le clonage du chemin entre le nœud ancêtre a et le nœud n permet de supprimer le nœud a de ce chemin. Cela amène, également, à définir une deuxième classe d'autorisation appelée "***réduction de chemin***". L'impact de cette deuxième classe est représenté, également, dans la Figure 26 (reduction de chemin pour le neoud n_3).

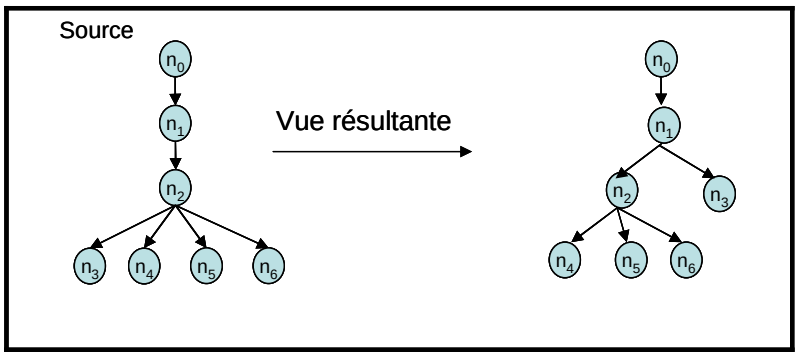


Figure 26 : Réduction de chemin

La sémantique des autorisations "*réduction de chemin*" est définie, en terme de leur impact attendu sur la Vue, comme suit : soit $n \in N_{Source}$. Soit $a, u \in N^e_{Source}$ / $a \in Anc(n)$ et u = Parent(n) .

- ***Réduction de chemin***
  - ***Si*** $a = u$ ***Alors*** Parent(n) ← Parent(a) ***Sinon*** {

    Soit $e \in N^e_{Source}$ / e = Child(a) ∩ Anc(n)
  - ***Si*** $e = u$ ***Alors*** Clonage_Element (e) ***Sinon*** Clonage_Path (e,u);
    - Parent($\tilde{e}$)←Parent(a);
    - Parent(n) ← $\tilde{u}$; // ***Si*** $e=u$ ***Alors*** $\tilde{e} = \tilde{u}$ }

Nous remarquons que *n* (et son potentiel sous-arbre) sont toujours attachés à l'extrémité du chemin cloné. Il est important à noter que les autorisations sur les nœuds, dans les modèles existants, suivent une approche "*top-down*" puisque les autorisations sur les nœuds se propagent, généralement, à tous ses descendants dans la hiérarchie XML. Inversement, la dépersonnalisation d'ancêtres et la réduction de chemin suivent une approche "*bottom-up*" ayant un impact sur l'association entre les nœuds descendants et leurs ancêtres.

- ***La filiation uniforme*** : l'objectif est de permettre aux nœuds descendants d'exprimer des exigences différentes en terme de la révélation de la classification. En d'autres termes, chaque descendant possède une vue différente par rapport à son ancêtre.

Par conséquent, les deux problèmes doivent être pris en considération ensemble. Le clonage donne une solution uniforme pour ces deux problèmes. En effet, le clonage du chemin reliant un ancêtre à un de ses descendants donne, également, l'opportunité de personnaliser la vue autorisée de chaque association d'ancêtre.

2.4 Impact sur les associations de fraternité

Lorsque nous masquons l'appartenance d'un nœud donné à une classe donnée, une attention particulière doit être donnée à la corrélation entre ce nœud et ses frères initiaux. La sémantique introduite dans la section précédente casse implicitement les associations de fraternité. Cet effet de bord peut violer, dans certaines situations, le principe de *need to know*. Inversement préserver implicitement toutes les associations de fraternité lorsque nous dissociant un nœud d'une classe peut violer le principe de *consentement*. Pour cette raison, nous avons besoin d'une décorrélation sélective des frères pour supporter le genre de situation où le nœud doit garder certaines associations avec ses frères initiaux lors de sa dissociation. La sémantique d'une décorrélation sélective est définie comme suit :

- **Caractérisation des associations de fraternité par rapport à l'association d'ancêtre**

Soit $n \in N_{Source}$ et *n'* son image dans N_{Vue}. Soit g1, g2 $\subset N_{Source}$ / $\forall$e1, e2 $\in$ g1$\cup$g2, Parent(e1) = Parent(e2) = Parent(n). La préservation de l'association de fraternité entre le nœud *n* et les éléments du g1 a l'impact suivant sur le document `Vue` :

- $\forall$e1 $\in$ g1, Parent(e1') $\leftarrow$ Parent (n') : attacher *n* et les éléments de *g1* au même parent dans la `Vue`, tandis que les éléments de g2 restent attacher à leur parent original dans la `Vue`.

La Figure 27 illustre la dissociation des nœuds n_4 et n_6 par rapport aux nœuds n_3 et n_5.

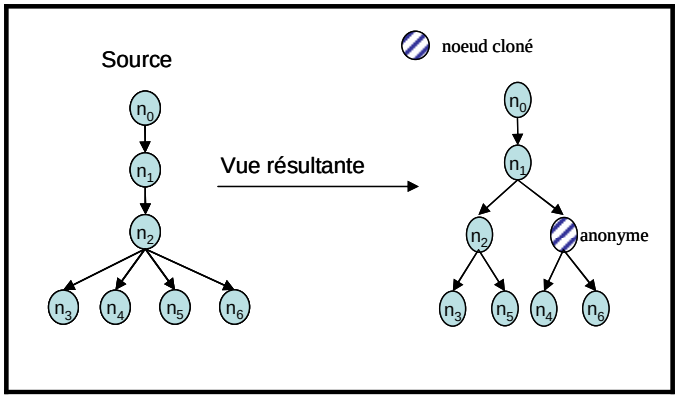


Figure 27 : Décorrelation sélective des associations de fraternité

En conclusion, le traitement de la divulgation de la classification et de la filiation uniforme introduit un problème à deux dimensions : (1) comment les associations entre un nœud donné et ses ancêtres doivent être mappés dans la `Vue` ; (2) quelles sont les associations de fraternité qui doivent être préservées pour ce nœud dans la `Vue`.

Dans la section suivante nous présentons comment les règles d'autorisation sur les associations sont définies et comment elles interagissent avec les règles d'autorisation définies sur les nœuds. La résolution de conflits entre ces différentes règles fait, également, l'objet de la section suivante.

3 Modèle de contrôle d'accès intégrant les associations

3.1 Préliminaires

Une vue autorisée `Vue` peut être vue comme étant le résultat d'un processus de sélection globale et de restructuration appliqué sur le document `Source`. À première vue, un langage tel que XQuery peut être considéré comme un moyen approprié pour définir un tel processus. Bien évidement, cette solution est possible puisque XQuery est un langage turing complet. Cependant, il a été prouvé [20] que, même, pour des restructurations simples du document XML, tel qu'une suppression d'un sous arbre ou d'un élément, cela nécessite des expressions XQuery relativement complexe. Les expressions ressemblent plutôt à des programmes itératifs qu'à des expressions déclaratives. Par conséquent, implémenter un modèle de contrôle d'accès en utilisant un programme XQuery rend sa lisibilité, sa gestion et sa validité difficiles.

Alors qu'un modèle de contrôle d'accès devrait être conçu pour satisfaire les propriétés suivantes :

- *Expressivité* : un sous ensemble de situations pratiques doit être exprimable.

- Concision : une politique de contrôle d'accès doit être exprimées par un ensemble minimum possible de règles d'autorisation.
- Validité : la `vue` résultant doit traduire exactement la sémantique d'une politique de contrôle d'accès donnée.
- Maniabilité : une politique de contrôle d'accès doit être compréhensible par un humain. Son évolution doit être également facile à gérer.

Afin de vérifier ces propriétés, les modèles de contrôle d'accès existants proposent d'exprimer la politique de contrôle d'accès par un ensemble de règles d'autorisation. Un moteur de règles réalise la résolution de conflits entre les règles et traduit le document `Source` en une `Vue` en se basant sur les règles d'autorisation. Grâce à sa simplicité et sa puissance d'expression, XPath est généralement le langage utilisé pour identifier les nœuds cibles dans chaque règle. Nous allons suivre, dans notre modèle, la même démarche adoptée par les modèles de contrôle d'accès pour exprimer les règles d'autorisation sur les associations.

Puisque chaque modèle de contrôle d'accès adopte de petites variantes concernant les règles de propagation et de résolution de conflit, nous introduisons, dans un premier temps, notre modèle de référence pour définir les règles d'autorisation sur les nœuds qui capture les bases communes des modèles de contrôle d'accès existants. Ensuite, nous proposons une extension de ce modèle de référence pour le support des autorisations sur les associations tout en préservant les quatre propriétés précédemment citées. Plutôt que de proposer un nouveau modèle de contrôle d'accès XML, nous montrons que l'approche proposée permet une intégration des autorisations sur les associations dans les modèles de contrôle d'accès existants.

3.2 Modèle de référence pour les autorisations sur les nœuds

Les modèles de contrôle d'accès existants partagent des similarités fortes dans leurs conceptions, leurs différences se situent dans les types de propagation utilisés, les politiques de résolution de conflits adoptés… Généralement, la règle d'autorisation est définie sous forme d'un quadruplet <*Sujet, Objet, Action, Signe*>. Tout dépend des modèles, le *Sujet* peut prendre plusieurs formes (utilisateur, groupe, rôle, adresse IP…etc.). L'*objet* caractérise la portion du document ciblée par la règle d'autorisation. L'*action* représente les opérations d'accès (lecture, mise à jour, suppression et ajout) que le sujet peut effectuer sur l'objet. Finalement, le *Signe* prend soit la valeur positive (+), ce que signifie que la permission est accordée, ou soit la valeur négative (-) pour une interdiction. Dans ce qui suit, nous ne nous s'intéressons pas à la façon de gérer les sujets, et nous traitons uniquement l'opération de lecture puisque nous nous intéressons, dans notre travail, à la confidentialité des données. Par conséquent, la règle d'autorisation sur les nœuds prend la forme simplifiée. Cela nous

permet de nous focaliser sur les aspects fondamentaux de notre contribution qui consiste à caractériser les objets à protéger. Ainsi, la règle d'autorisation sur les nœuds *AN* est définie comme suit : <*Sujet, Objets, Signe*>.

- Sujet : est une entité abstraite,
- Objet $\subseteq N_{Source}$,
- Signe $\in$ {+, -}

Objet correspond aux attributs et aux éléments d'un document source, identifiés par une expression XPath. La puissance d'expression d'un modèle de contrôle d'accès, et donc la granularité de partage, est liée directement au sous-ensemble de XPath supporté. Nous considérons, par la suite, un sous-ensemble significatif de XPath désigné par $XP^{\{[],*,//\}}$ [86]. Ce sous-ensemble, largement utilisé en pratique, englobe l'utilisation de tests sur les nœuds, l'axe parent-enfant (/), l'axe de descendance (//), de jokers (*) et de prédicats [].

La combinaison des règles d'autorisation positives et des règles d'autorisation négatives représente un bon moyen pour gérer les exceptions [72]. Afin de respecter le principe de *moindre privilège* nous adoptons la politique fermée. Nous rappelons que la politique fermée signifie qu'une règle d'autorisation négative s'applique, par défaut, à tout le document. En d'autres termes, l'accès à l'objet qui n'est pas explicitement autorisé est interdit.

Nous supposons que la propagation des règles d'autorisation positives et des règles d'autorisation négatives est implicite, dans notre modèle, ce qui signifie qu'une règle se propage d'un noeud à tous ses descendants dans la hiérarchie XML. Ce mode de propagation correspond à l'option *cascade* présente dans les modèles les plus connus [8, 54, 37]. Etant donné cette politique de propagation et le fait que plusieurs règles (positives et négatives) peuvent être définies pour un même utilisateur sur un même document, l'utilisation des politiques de résolution de conflit est indispensable. Ces politiques sont "l'*interdiction est plus prioritaire*" et *l'objet le plus spécifique est plus prioritaire*".

Considérons deux règles R1 et R2 de signe opposé. Ces règles peuvent être en conflit soit parce qu'elles sont définies sur le même nœud, soit parce qu'elles sont définies respectivement sur deux nœuds différents n1 et n2, reliés par une relation ancêtre-descendant (i.e., $n_1 \in Anc(n_2)$). Dans le premier cas, la politique "l'*interdiction est plus prioritaire*" donne la priorité à la règle négative. Dans le second cas, la politique "*l'objet le plus spécifique est plus prioritaire*" donne la priorité à la règle qui s'applique directement sur le nœud (i.e., R2 est plus prioritaire pour le nœud n2). En d'autres termes, les règles d'autorisation se propagent jusqu'à ce qu'une autre règle du signe opposé s'applique.

3.3 Les règles d'autorisation sur les associations

Le challenge est d'étendre le modèle de référence avec les règles d'autorisation sur les associations [50, 51, 52] tout en préservant les propriétés concision, validité et maniabilité

définies précédemment. Nous proposons, dans cette section, la syntaxe d'une règle d'autorisation sur les associations. Ainsi que, des expressions unifiées permettant la déclaration des autorisations sur les associations introduites dans la section 2.

3.3.1 La règle d'autorisation sur les associations

Une règle d'autorisation sur les associations *AA* est définie sous la forme d'un tuple <*Sujet, objet*>, où l'*objet*, à son tour, est défini par un quadruple : <*Anc, Desc, Visibilité, fraternité*> :

- *Anc* et *Desc* caractérisent l'association à protéger entre un ou ensemble de descendant(s) et un de leur ancêtre. *Anc* et *Desc* constituent le dénominateur commun de toutes les autorisations sur les associations. Ils sont identifiés par les expressions XPath.
- *Visibilité* caractérise la visibilité du chemin *u* reliant chaque nœud descendant à son ancêtre. Pour chaque nœud *n* participant dans *u*, *Visibilité* dit si le nœud est préservé ou non dans le chemin cloné *ũ*. Dans le cas où le nœud sera préservé la *Visibilité* dit, également, si le label du nœud *n* sera préservé ou non dans *ñ*.
- *Fraternité* : implicitement, cacher l'association avec l'ancêtre revient à cacher, également, l'association entre le nœud descendant et ses frères. Pour permettre une décorrélation sélective entre les frères, le paramètre *fraternité* caractérise la liste des frères que le descendant doit garder au moment de sa dissociation.

La définition de la règle d'autorisation sur les associations *AA* soulève deux remarques importantes. La première, concerne la concision et la maniabilité, *AA* capture, gracieusement, et d'une façon très simple toutes les formes d'autorisation sur les associations. La deuxième remarque, contrairement à AN, AA ne comporte pas le paramètre *signe*. La raison pour cela est que AA caractérise uniquement les règles négatives qui visent à chaque fois à dissocier le nœud en question.

La sémantique globale de notre modèle est comme suit : les règles d'autorisation sur les nœuds (AN) sont définies selon une politique fermée et retourne en résultat une vue autorisée `Vue' ⊆ Source.` D'une façon générale, les arcs ayant des noeuds extrémités supprimés par une règle AN sont à leurs tours supprimés de la `Vue'.` Tandis que, les règles d'autorisation sur les associations sont définies selon une politique ouverte qui retourne en résultat la vue finale autorisée `Vue`. Par conséquent, si aucune règle d'autorisation sur les associations n'est définie, la sémantique de notre modèle est conforme avec les modèles de contrôle d'accès existants dans la littérature. Par conséquent, l'intégration des autorisations sur les associations dans ces modèles existants peut être effectuée facilement.

3.3.2 Les ancêtres et les descendants

Pour qu'une règle d'autorisation sur les associations soit cohérente, elle doit satisfaire la condition *Desc* ⊆ *Anc*, où ⊆ désigne la relation d'inclusion entre les expressions XPath. Malheureusement, le problème d'inclusion a été montré co-NP complet pour la classe d'expression XP $^{\{[],*,//\}}$ [86]. Afin d'éviter les vérifications d'incohérence, *Desc* est défini comme une expression relative (chemin relatif) par rapport à Anc. Donc, Anc détermine un ensemble d'origines du chemin et Anc/Desc détermine un ensemble d'extrémités de chemin.

3.3.3 La visibilité du chemin

Pour chaque nœud *n* participant au chemin sélectionné par *Anc* et *Desc*, *Visibilité* doit spécifier si *n* participe ou non au chemin cloné dans `Vue.` Si cela est vrai, alors *Visibilité* doit spécifier, également, si le label de *n* est hérité ou non par *ñ*, clone de *n*. Nous définissons quatre possibilités différentes pour le paramètre *Visibilité* : i) sélectionner les nœuds à supprimer de chemin cloné ou à dépersonnaliser leur label original ; ii) préserver tous les nœuds de chemin cloné avec leur label original. Cette option représente l'option par défaut. iii) préserver tous les nœuds de chemin cloné tout en anonymisant leur label original ; iv) enfin, la suppression de tous les nœuds du chemin cloné est envisagée. Le Tableau 1 récapitule les choix possibles pour ce paramètre et leurs sémantiques. La Figure 28 illustre graphiquement sur un chemin de type /A, //C leurs effets. La première ligne de la table donne une syntaxe étendue pour ce paramètre et les autres lignes proposent une abréviation d'expression de la politique sur le chemin.

Visibilité de chemin	**Sémantique de visibilité de chemin**
[$label_1$?,…, $label_n$?]	Donne la liste des nœuds à supprimer (?= †) ou à dépersonnaliser (?= Φ).
[]	Tous les nœuds sont préservés sur le chemin (i.e, tous les nœuds clonés) et leur label original est hérité. Cette option est l'option par défaut.
[Φ]	Tous les nœuds sont préservés sur le chemin mais ils sont dépersonnalisés (i.e, le label de leur clone est égal à " anonyme").
[†]	Tous les nœuds sont supprimés du chemin.

Tableau 1 : La sémantique de Visibilité du chemin

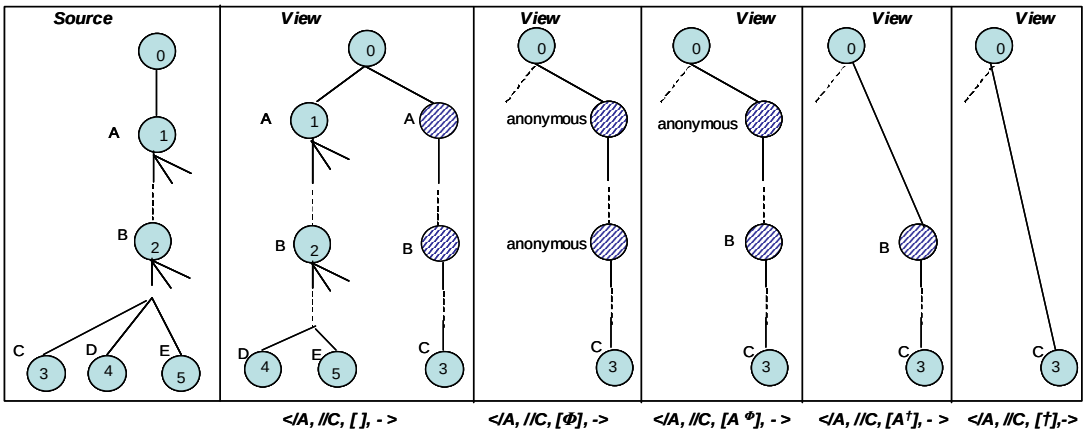


Figure 28 : Exemple de Visibilité du chemin

3.3.4 Les associations de fraternité

Le paramètre *fraternité* est utilisé pour exprimer la décorrelation sélective des frères. Ce paramètre donne la liste des frères qu'un descendant doit préserver lors de sa dissociation. Le nœud descendant peut préserver les associations de fraternité avec certains nœuds ou avec des nœuds ayant le même label. Il peut également se déconnecter de tous ses frères ou préserver tous ses associations de fraternité. Le Tableau 2 récapitule les choix possibles pour ce paramètre avec leurs sémantiques et la Figure 29 illustre graphiquement leurs effets. Encore une fois, la première ligne de tableau donne une syntaxe étendue pour ce paramètre tandis que les autres lignes proposent une abréviation d'expression de la politique.

Fraternité	**Sémantiques du parametre *fraternité***
[$label_1$,… $label_n$]	Tous les noeuds ayant un label appartenant à cette liste doivent préserver leurs associations avec le descendant en question.
[⊥]	Le noeud descendant est déconnecté de tous ses frères. C'est l'option par défaut.
[ψ]	Le noeud descendant préserve ses associations de fraternité avec tous les frères ciblés par la même règle d'autorisation.
[≡]	Le noeud descendant préserve ses associations de fraternité avec tous ses frères.

Tableau 2 : La sémantique de paramètre de *fraternité*

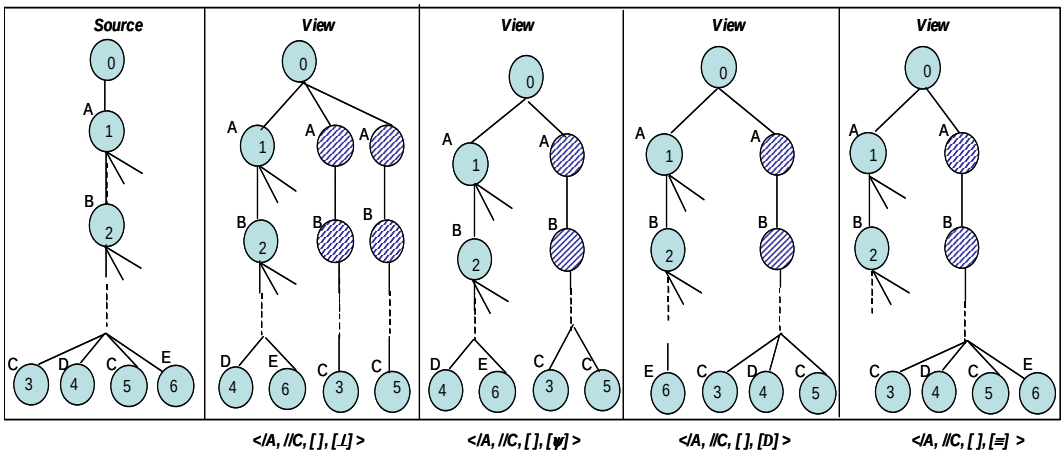


Figure 29 : Exemples de décorrélation sélective

3.4 Résolution des conflits

Dans le modèle de contrôle d'accès étendu aux associations, trois classes de conflits doivent être gérées [50, 51, 52]: (i) les conflits entre les règles d'autorisation sur les nœuds ; (ii) les conflits entre les règles d'autorisation sur les nœuds et les règles d'autorisation sur les associations ;(iii) les conflits entre les règles d'autorisation sur les associations. La résolution des conflits entre les règles d'autorisation sur les nœuds est héritée directement des modèles de contrôle d'accès existants que nous avons déjà expliquée dans la section 3.2. Les conflits entre les règles AN et AA sont évités puisque les règles sur les associations sont définies sur la vue produite après l'évaluation des règles d'autorisation sur les nœuds, selon une politique ouverte par défaut. En d'autres termes, les règles AN sont, toujours, plus prioritaires que les règles AA. Par conséquent, cette section est consacrée à la gestion de la troisième classe des conflits existants entre les règles d'autorisation sur les associations.

Encore une fois, trois classes de conflits entre les règles AA doivent être distinguées : (i) des conflits sur le descendant (des règles différentes et conflictuelles sont définies sur le même nœud descendant); (ii) des conflits sur la visibilité du chemin (des règles différentes sont définies sur la même association ancêtre-descendant avec des options différentes de visibilite); (iii) les conflits sur la fraternité (différentes règles sont définies sur la même association ancêtre-descendant avec des options différentes de décorrélation). Dans ce qui suit nous allons détailler les principes de résolution des conflits de ces trois classes.

3.4.1 Les conflits sur le descendant

Les conflits sur le Desc se produisent lorsque deux règles AA1 et AA2 sont définies sur le même descendant tel que AA1.Anc $\subseteq$ AA2.Anc et AA1.Desc = AA2.Desc, où $\subseteq$ désigne l'inclusion des requêtes XPath.

Indépendamment des valeurs des paramètres *fraternité* et *visibilité* définis dans les

règles, la règle AA1 dit qu'un nouveau chemin doit être crée pour tout couple de nœuds (parent(n1), n2) tel que n1∈ AA1.Anc et n2∈ AA1.Desc, en conséquence cacher l'association d'ancêtre entre le nœud n1 et le nœud n2. D'une façon similaire, la règle AA2 dit qu'un nouveau chemin doit être crée pour tout couple de nœuds (parent(n3), n2) tel que n3∈ AA2.Anc et n2∈ AA2.Desc. Puisque AA1.Anc⊆AA2.Anc, parent(n3)∈ Anc(parent(n1)). Par conséquent, cacher l'association d'ancêtre entre n3 et n2 revient à cacher l'association d'ancêtre entre n1et n3, due à la transitivité de l'association d'ancêtre. Par conséquent, la AA2 l'emporte car elle représente la règle la plus restrictive.

3.4.2 Les conflits sur la visibilité du chemin

Les conflits sur la visibilité du chemin se produisent lorsque deux règles différentes ciblent la même association ancêtre-descendant (i.e., AA1.Anc = AA2.Anc and AA1.Desc = AA2.Desc) avec des visibilité du chemin différents. Le Tableau 3 récapitule toutes les combinaisons possibles ainsi que leur résolution de conflits associée. La résolution de conflits étant commutative, operande1 et opérande2 représente soit les paramètres de *visibilité* de la règle AA1 ou de la règle AA2. La décision de la résolution de conflit est, toujours, prise en respectant le principe du moindre privilège.

Opérande1	**Opérande2**	**Résolution de conflits**
[ ]	∀[23]	Opérande2
[†]	∀	[†]
[Φ]	[Φ]	[Φ]
[Φ]	[$label_1$?, …, $label_n$?]	∀ noeud ∈ Path(anc,desc) si ϕ_{label} (noeud) ∉ [$label_1$?,..,$label_n$?] alors [$label_1$?,.,$label_n$?] U [ϕ_{label} (noeud)$^{\Phi}$]
[$label_1$?,..., $label_n$?]	[$label'_1$?,…, $label'_n$?]	L = [$label_1$?,…,$label_n$?] U [$label'_1$?,…,$label'_n$?] ∀$label_i$∈ L, $label_i^{\dagger}$∈ L et $label_i^{\Phi}$ ∈ L ⇒ L – [$label_i^{\Phi}$]

Tableau 3 : Résolution des conflits sur la visibilité de chemin

Si une des règles d'autorisation n'a pas imposé des restrictions sur la visibilité du chemin alors la deuxième règle l'emporte (ligne 1). La règle supprimant tous les nœuds du chemin est plus prioritaire (ligne 2). La ligne 3 est évidente. Si une règle dépersonnalise tous les nœuds sur le chemin tandis que l'autre règle sélectionne une liste de labels, alors cette liste de labels doit être étendue avec les labels dépersonnalisés manquants (ligne 4). Enfin, si

[23] ∀ : représente n'importe quelle opérande

chaque règle sélectionne une liste de labels alors l'union de ces deux listes doit être calculée. Dans le cas où le même label est présent dans les deux listes, la destruction du nœud est plus prioritaire que sa dépersonnalisation.

3.4.3 Les conflits sur la fraternité

Les conflits entre les frères se produisent lorsque deux règles différentes ciblent la même association ancêtre-descendant avec deux options différentes de décorrélation des frères. Le Tableau 4 récapitule toutes les combinaisons de fraternité possibles pour ces deux règles, ainsi que leur résolution des conflits associée. Encore une fois, la résolution des conflits proposée étant commutative, opérande1 et opérande2 représentent soit les paramètres de *fraternité* de la règle AA1 ou de la règle AA2. La résolution des conflits est faite en respectant, toujours, le principe du moindre privilège.

Opérande1	**Opérande2**	**Résolution des conflits**
$[\perp]$	$\forall$	$[\perp]$
$[\equiv]$	$\forall$	Opérande2
$[\psi]$	$[label_1,\ldots label_n]$	$[\psi]$ si ϕ_{label} (desc) $\in [label_1, \ldots label_n]$, $[\perp]$ autrement
$[label_1,\ldots label_n]$	$[label'_1,\ldots label'_n]$	$[label_1, \ldots label_n] \cap [label'_1, \ldots label'_n] = A$, $[\perp]$ si $A = \varnothing$

Tableau 4 : Résolution de conflits sur les frères

La ligne 1 dit que $[\perp]$ est plus prioritaire que $[\psi]$, $[label_l,\ldots, label_n]$ et $[\equiv]$, puisqu'elle représente la politique la plus restrictive (le descendant est dissocié de tous ses frères). De la même façon, $[\psi]$ et $[label_l,\ldots, label_n]$ sont plus prioritaires que $[\equiv]$, puisque $[\equiv]$ est la politique la moins restrictive (ligne 2). Le conflit entre $[\psi]$ et la liste $[label_l,\ldots, label_n]$ (ligne 3) se résoud de telle sorte que si le label de descendant est inclus dans la liste $[label_l,\ldots, label_n]$ alors $[\psi]$ l'emporte. Sinon $[\perp]$ l'emporte et, par conséquent, le nœud se dissocie de tous ses frères. Enfin, la résolution de conflit entre deux listes de labels consiste à calculer l'intersection entre ces deux listes. Si cette intersection est vide alors $[\perp]$ devient la décision finale.

Il est à noter, également, que les conflits entre frères peuvent survenir indirectement si le même nœud est sélectionné pour participer dans des groupes de frères différents. Par exemple, AA1 :<-,/B,-,[A]> et AA2:<-,/C,-,[A]>. Dans AA1, le nœud descendant B veut préserver son lien de fraternité avec le nœud A. Dans la règle AA2, le descendant C veut, également, préserver son lien de fraternité avec le nœud A. Nous remarquons que le nœud A est sollicité par deux nœuds frères différents (B et C). A première vue, l'intuition peut

amener à dupliquer les éléments A afin de les intégrer, à la fois, dans les frères de B et les frères de C. Cependant, la présence de même(s) élément(s) A dans les deux groupes est une source d'inférence concernant l'association entre les éléments B et les éléments C. A titre d'exemple, la règle AA1 a pour objectif de décorréler le neoud *Nom* du patient et le noeud *actesMed*. La règle AA2 a pour objectif de décorréler le nœud *Nom* du patient et le nœud *Analyses*. Le faite que le même nom du patient est présent dans les deux groupes, l'utilisateur peut inférer facilement la structure initial du document et faire le lien entre les actes médicaux et les analyses. Pour cette raison, le conflit est résolu en appliquant "[⊥]" comme paramètre de *fraternité* dans les règles conflictuelles.

3.5 Exemple d'expression des politiques de contrôle d'accès

Nous montrons dans cette section comment le modèle de contrôle d'accès étendu aux règles d'autorisation sur les associations est utilisé pour exprimer les règles de contrôle d'accès introduites dans notre exemple de motivation défini au chapitre III. Chacune d'elles comprend un mélange entre des règles d'autorisation sur les nœuds et des règles d'autorisation sur les associations. Certaines règles AN et AA font référence au consentement de l'utilisateur. Nous avons supposé que le consentement de l'utilisateur est matérialisé par un élément `Consentement` présent dans chaque dossier médical du patient. L'élément `Consentement` est décomposé, à son tour, en sous-éléments (`Repertoire`, `commercialisation`) qui exprime chaque dimension du consentement de l'utilisateur.

Pour exprimer la règle R1, trois règles AN sont nécessaires. AN1 et AN2 et AN3 capturent les informations strictement nécessaires pour le groupe responsable de la gestion d'annuaire afin d'accomplir leur tâche de travail. Par conséquent, les éléments `ActesMed` et `Analyses` sont supprimés. La règle d'autorisation sur les associations AA1 dépersonnalise ([Φ]) l'ancêtre service médical (/* identifie tous les éléments service médical) de chaque dossier médical du patient qui n'a pas consenti à divulguer son service médical et, par conséquent, elle dissocie le dossier médical de tous ses frères ([⊥]).

Règle d'autorisation R1 :

AN1: < Groupe-annuaire , /Hôpital, + >

AN2: < Groupe-annuaire, //ActesMed, - >

AN3: < Groupe-annuaire, //Analyses, - >

AA1: <Groupe-annuaire, /*, /Dossier[./Consentement/Répertoire = 'no visible'], [Φ], [⊥]>

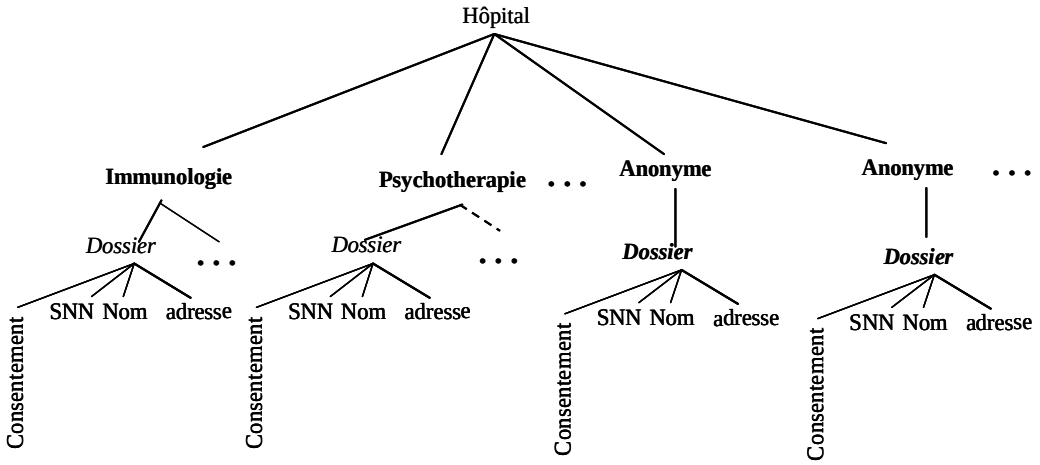


Figure 30 : Vue retornée pour la règle R1

L'expression de la règle R2 se fait également par un mélange de règles AN et AA. La règle AA2 exprime une réduction du chemin supprimant le parent des éléments `Acte ("Protocole")`. En plus, le pharmacien n'a pas le droit du voir les analyses, les diagnostics, ainsi que les symtomes. Cela est exprimé par les règles AN5, AN6 et AN7 suppriment respectivement les éléments `Analyses`, `diagnostics et symptômes.`

Règle d'autorisation R2

AN4 : <Pharmacien, //Hôpital, + >

AN5 : <Pharmacien, //Analyses, - >

AN6 : <Pharmacien, //diagnostics, ->

AN7 : <Pharmacien, //symptômes, ->

AA2 : <Pharmacien, //ActesMed/Protocole, /Acte, [†], [⊥] >

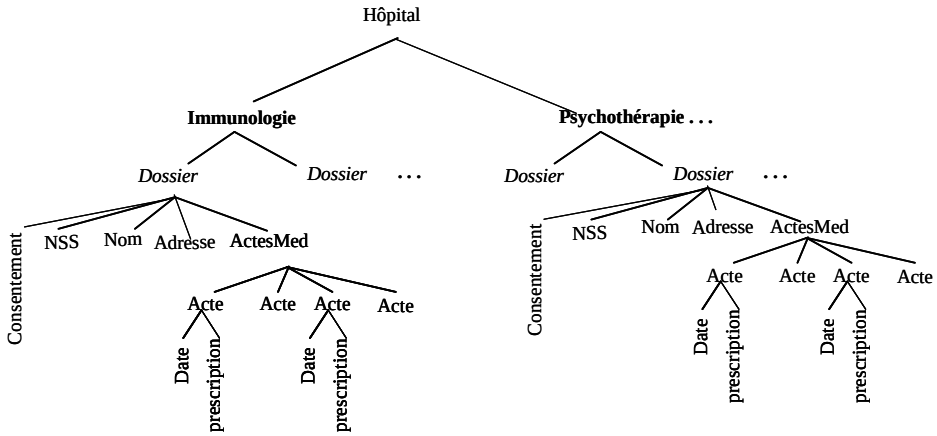


Figure 31 : Vue retornée pour la règle R2

Enfin, pour la règle R3, deux règles sur les nœuds (AN9 et AN10) interdisent au laboratoire médical d'accéder aux noms, aux adresses des patients qui n'ont pas donné leur consentement pour des raisons de commercialisation. La règle AN11 supprime tous les nœuds NSS. Pour les patients donnant leur consentement, AA3 empêche l'inférence entre les informations d'identification (`nom`, `adresse`) et le reste du dossier médical. Nous presentons dans la Figure 32 le résultat pour un seul service « Immunologie »[24].

Règle d'autorisation R3

AN8 : < lab Médical, //Hôpital, + >

AN9:<lab Médical,//Dossier[./Consentement/Commercialisation='no-visible']/nom, ->

AN10 :<lab Médical, //Dossier[./Consentement/Commercialisation = 'no-visible']/Adresse, ->

AN11 : :< lab Médical, // NSS, ->

AA3 : < lab Médical, //Dossier, /nom, [ ], [Adresse]>

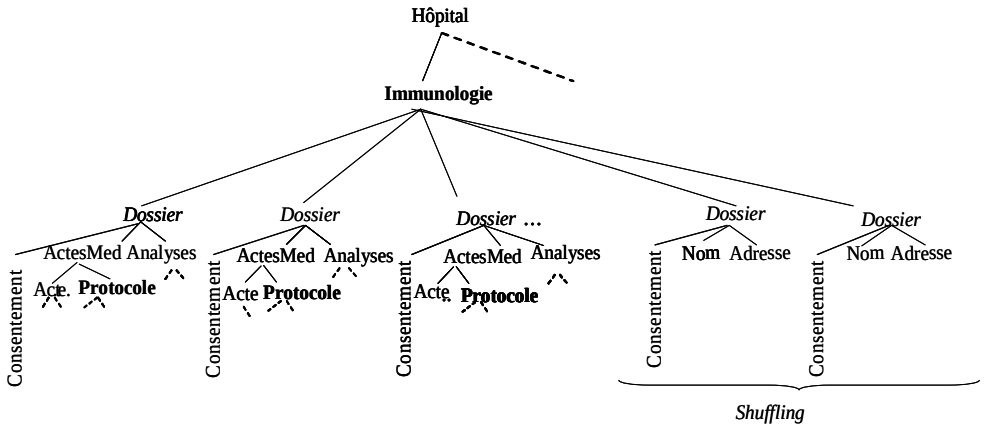


Figure 32 : Vue retornée pour la règle R3

4 Conclusion

Divers modèles de contrôle d'accès ont été proposés dans la littérature pour réguler l'accès aux documents XML. En focalisant le contrôle d'accès, uniquement, sur les nœuds et en ne prenant pas en compte les associations entre eux, ces modèles de contrôle d'accès échouent à répondre aux problèmes de ***divulgation de la classification*** et de la ***filiation uniforme***. La conséquence est la violation des principes de ***consentement*** et ***need-to-know*** (les deux principes fondateurs des législations relatives à la protection des données à caractère personnel).

[24] problème d'espace

Afin de répondre aux deux problèmes précédemment cités, notre approche défend l'intégration des associations d’ancêtres et de fraternité entre les nœuds comme des éléments de première classe dans les modèles de contrôle d'accès. Nos contributions portent sur : (1) la caractérisation de deux classes d’autorisation sur les associations (***dépersonnalisation des ancêtres*** et la ***réduction de chemin***) en prenons en compte la dimension des associations de fraternité. Deux mécanismes (clonage et shuffling) sont introduits pour traduire exactement ces associations en une vue autorisée ; (2) la définition d’un formalisme à base de règles pour exprimer ces classes d’autorisation sur les associations en permettant leur intégration d’une façon très simple dans les modèles de contrôle d’accès existants.

La description de notre prototype fera l’objet du chapitre suivant.

Chapitre V – Implémentation et Etudes de Performances

1 Introduction

Dans ce chapitre, nous décrivons le prototype que nous avons réalisé ainsi que les tests d'évaluation de performance.

Notre prototype s'appuie sur le seul prototype existant dans le domaine public concernant la gestion des droits d'accès XML. Le choix de repartir d'un prototype déjà existant a été motivé par l'envie de montrer que notre modèle de contrôle d'accès pouvait se concevoir comme une extension des modèles de contrôle d'accès existants.

L'objectif principal de notre étude est de mesurer, plus particulièrement, le surcoût des règles d'autorisation sur les associations par rapport au coût des règles d'autorisation sur les nœuds et de voir si ce surcoût est raisonnable ou pas. Pour cela nous avons suivi une méthodologie qui nous a permis de tirer des conclusions pertinentes.

L'organisation de ce chapitre est comme suit. Dans une première section, nous décrivons les caractéristiques de notre prototype. La deuxième section est consacrée à l'étude des performances. Cette section présente notre démarche et les résultats des tests. La troisième section décrit certaines améliorations apportées au prototype. Enfin, une conclusion termine ce chapitre.

2 Le prototype

Afin d'implémenter notre modèle de contrôle d'accès, nous avions deux choix possibles : i) soit développer un prototype à partir de zéro, ii) soit étendre un prototype déjà existant. Nous avons choisi la seconde approche pour les raisons suivantes : i) nous voulions montrer que les règles d'autorisation sur les associations peuvent être intégrées facilement dans un modèle existant quel qu'il soit ; ii) nous voulions calculer le surcoût des règles d'autorisation par rapport aux règles d'autorisation sur les nœuds. Pour cela, nous nous sommes basés sur le seul prototype rendu public et développé par une équipe italienne. Ce prototype appelé ici par le nom de son concepteur Damiani [40] a été écrit en java.

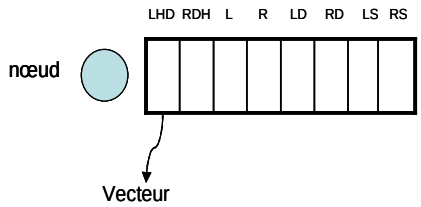
Dans cette section, nous commençons par rappeler les différentes étapes de l'algorithme

de Damiani [37]. Puis dans un second temps, nous décrivons les caractéristiques de notre algorithme supportant les règles d'autorisation sur les associations.

2.1 L'algorithme de Damiani

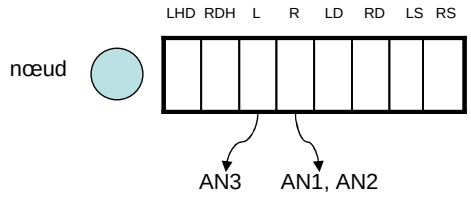
L'algorithme de Damiani est basé sur l'approche "Matérialisation de la vue autorisée" (présentée dans le chapitre II) pour calculer la vue autorisée. Cet algorithme évalue les politiques de contrôle d'accès composées uniquement des règles d'autorisation sur les nœuds. Il fonctionne en quatre phases qui peuvent être décrites indépendamment comme suit :

1. *Phase de construction de l'arbre DOM* : l'algorithme parse le document source et en construit une représentation DOM en utilisant l'outil Xerces. Une allocation des structures de données allouées est faite à cette étape. Ces dernières servent pour le marquage des nœuds par les règles d'autorisations qui les concernent. A chaque nœud *n* est associé un *tableau* de 8 cases qui correspond aux différents types d'autorisation adoptées par le modèle de Damiani (LDH (local hard authorization)[25] > RDH (recursive hard authorization) > L (local authorization) > R (recursive authorization) > LD (local authorization over the DTD) > RD (recursive authorization over theDTD) > LS (local soft authorization) > RS (recursive soft authorization)). Chaque case du tableau contient un *vecteur* qui sert à stocker toutes les autorisations du même type définies pour le nœud *n*. Une représentation du nœud avec sa structure de données est schématisée ci-dessous.



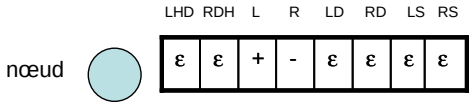
2. *Phase de marquage de l'arbre* : cette étape identifie les autorisations qui concernent le demandeur d'accès (un sujet donné). Les nœuds sont marqués par les règles d'autorisation qui s'appliquent sur eux. Plus précisément, à chaque fois qu'un nœud est ciblé par une expression XPath définie par l'objet de la règle d'autorisation, il sera marqué par cette règle. Bien évidement, la règle d'autorisation sera ajoutée à la case du tableau qui correspond à son type. Par exemple, si trois règles d'autorisation ont été définies pour le noeud *n*, deux règles AN1 et AN2 de type récursif (R) et une autre AN3 de type local. Le résultat du marquage sera comme suit :

[25] " >" : Représente le symbole de priorité entre les règles d'autorisation



3. *Phase de résolution des conflits et de propagation* : cette phase consiste à résoudre les conflits entre les règles d'autorisation définies sur le même nœud. Dans un premier temps, l'algorithme utilise la politique de résolution de conflit, "*le sujet le plus spécifique est plus prioritaire*" pour les règles d'autorisation conflictuelles du même type (récursives, par exemple). Dans le cas où le conflit persiste, l'algorithme applique la politique de résolution de conflit "*la règle d'autorisation négative est plus prioritaire*". Par exemple, le marquage après la résolution de conflit peut être comme suit :

   "ε" : Signifie qu'aucune règle de type (LHD) n'a été spécifiée pour ce nœud.



   Une fois les conflits résolus, les règles d'autorisation de type récursif se propagent à tous les descendants. A la fin de la phase de propagation, chaque nœud sera marqué par un seul signe qui détermine son accessibilité. Comme les types de règles sont ordonnés par ordre de priorité, le premier signe (+ ou -) l'emporte. Par rapport à l'exemple précédant, c'est le signe "+" qui l'emporte. Dans le cas où il n'existe aucun signe explicite (c-à-d il n'existe que des signes "ε") dans le tableau, le nœud sera marqué par le signe "-" (politique *fermée* par défaut).

4. *Phase de suppression des nœuds* : cette phase supprime tous les noeuds marqués négativement e n'ayant pas de descendant positif. S'il existe un descendant positif, l'algorithme supprime uniquement les attributs du nœud en question s'ils existent.

2.2 Notre Algorithme

L'extension de l'algorithme de Damiani par des règles d'autorisation sur les associations peut se faire d'une manière très simple. Nous décrivons, dans ce qui suit, cette extension tout en précisant les phases étendues et les phases propres à chaque algorithme.

5. *Phase de construction de l'arbre DOM* : afin de marquer les nœuds par les règles d'autorisation sur les associations, nous avons associé à chaque nœud un *vecteur* qui stocke les règles sur les associations s'appliquant sur lui. Par

conséquent, un nouveau vecteur a été ajouté chaque nœud. En total, chaque nœud possède un vecteur pour les règles d'autorisation sur les nœuds et un vecteur pour les règles d'autorisation sur les associations.

6. *Phase de marquage de l'arbre* : cette phase est étendue par l'évaluation des règles d'autorisation sur les associations. Chaque nœud (i.e . descendant) ciblé par une expression XPath correspondant à la concaténation des expressions XPATH (i.e. XPATHAnc/XPATHDesc) spécifiées dans la règle sur les associations, est marqué par cette règle. Par exemple, si le noeud *Desc* (présenté dans le schéma ci-dessous) est ciblé par deux règles AA1 et AA2, le vecteur associé au nœud descendant contient ces deux règles d'autorisation (AA1 et AA2).

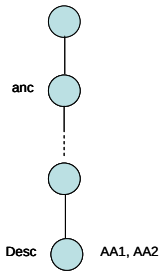


Figure 33 : Marquage de nœuds

7. *Phase de résolution de conflit* : cette phase est également étendue par la gestion des conflits entre les règles d'autorisation sur les associations ciblant le même nœud. Les politiques de résolution de conflit décrites dans le chapitre IV (section 3.4) sont appliquées ici. Après cette phase, chaque nœud est marqué par une et une seule règle d'autorisation sur les associations. Par exemple, d'après la Figure 33, si la règle AA2 est prioritaire, le marquage devient comme suit (Figure 34) :

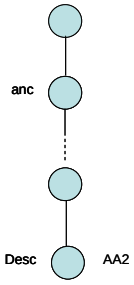


Figure 34 : Résolution de conflit

8. *Phase de reconstruction de l'arbre* : représente la dernière phase, après la suppresion des nœuds, de l'algorithme qui est responsable de la reconstruction de la vue autorisée du demandeur d'accès. C'est une phase indispensable pour appliquer les deux mécanismes de base "*le clonage* et *le shuffling*" (la

description de ces deux mécanismes est présentée dans la Figure 35). Dans la Figure 35, nous supposons que le processus de reconstruction a été déjà fait pour les nœuds n_5, n_9 et n_{10}. Ce processus s'effectue comme suit. Dans un premier temps, pour chaque nœud marqué (e.g, le noeud n_6) par la règle d'autorisation sur les associations, nous identifions son ancêtre spécifié dans la règle (e.g, n_1). Nous clonons, ensuite, le chemin entre le nœud descendant et le père de son ancêtre. Cela consiste, tout simplement, à créer de nouveaux éléments. Le nombre d'éléments à créer est égal au nombre de nœuds existants sur le chemin entre le descendant et le père de son ancêtre (e.g n_0). Chaque nœud cloné est marqué par l'attribut = "*clone*" pour les distinguer par la suite des nœuds originaux. Dans notre schéma (Figure 35), nous présentons les nœuds clonés par des nœuds hachurés. Une fois le nouveau chemin créé, nous dissocions le nœud descendant (e.g, n_6) et nous le rattachons au chemin cloné (étape 1 de la figure).

Dans un second temps, nous rattachons le chemin cloné au père du nœud ancêtre (n_0). La politique de rattachement spécifie que les chemins clonés sont rattachés à la suite des nœuds-fils originaux de l'ancêtre. Afin d'éviter les problèmes d'inférence, une position ***aléatoire*** parmi les nœuds descendants clonés (étape 2 de la figure) est calculée. Ce dernier mécanisme est appelé le ***shuffling***. C'est une fonction qui cherche une position aléatoire parmi les clones (ou les nouveaux éléments) pour rattacher le nouveau chemin conduisant au nœud descendant dissocié.

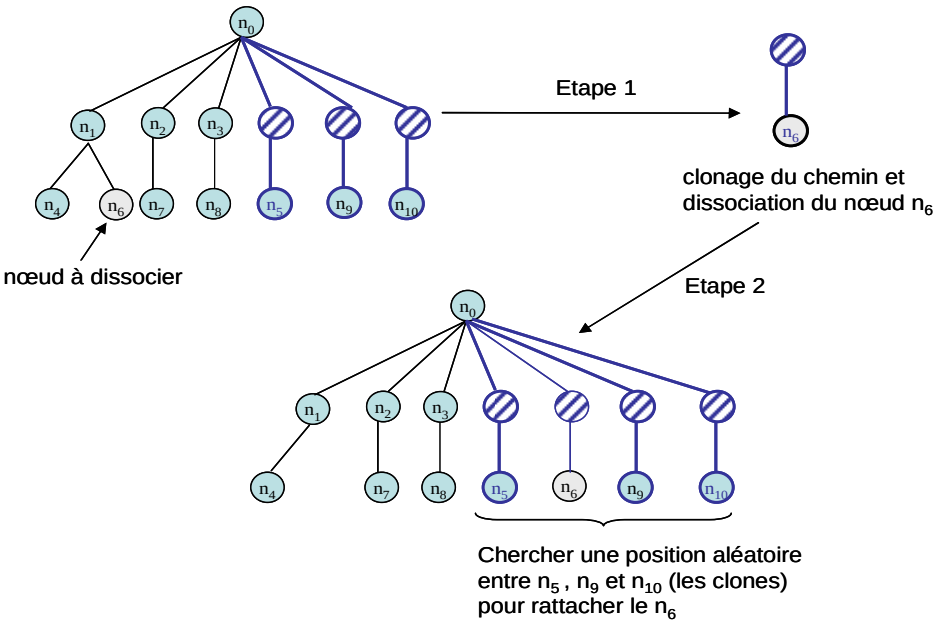


Figure 35 : Mécanismes de clonage et de shuffling

Concernant les fonctionnalités implémentées, le prototype implémente les principales

politiques de résolution de conflit sur la visibilité de chemin et sur les associations de fraternité.

Les politiques de résolution de conflits implémentées sur la visibilité de chemin sont présentées dans Figure 36 qui correspond partiellement au tableau 3 de chapitre IV.

Opérande1	Opérande2	Resolution de conflit
anc1	anc2	Si *anc1* est l'ancêtre de *anc2* alors *anc1* qui l'emporte sinon anc2
[ ]	∀	Opérande2
[†]	∀	[†]
[Φ]	[Φ]	[Φ]

Figure 36 : Politiques de résolution de conflits implémentées pour la visiblité

Pour les associations de fraternité, la seule politique de résolution de conflit implémentée est la "*dissociation de toutes les associations de fraternité de n'importe quel autre type de lien de fraternité*". Cela correspond à la première ligne du Tableau 4 de chapitre IV.

Opérande1	**Opérande2**	**Résolution des conflits**
[⊥]	∀	[⊥]

Figure 37 : Politiques de résolution de conflits implémentées pour la fraternité

Les différentes opérations de reconstruction, présentées dans le chapitre IV (Figure 28 et Figure 29), ont toutes été implémentées.

3 Etude des performances

Différentes expérimentations ont été réalisées sur le prototype. Afin d'obtenir des résultats significatifs, nous avons suivi une certaine méthodologie de test que nous décrivons dans une première section. Ensuite, nous donnons les résultats et conclusions obtenus sur nos tests. Chacune des phases de l'algorithme décrites précédemment est mesurée.

3.1 Plateforme de test

Nos jeux de tests ont été effectués sur une machine possédant un CPU cadencé à 3Ghz et 1 GB de RAM. Nous avons généré 15 documents XML de taille allant de 250 Ko jusqu'à 3,5

Mo qui correspond respectivement à 10.000 nœuds (10 Knœuds) jusqu'à 150.000 nœuds (150 Knœuds). Nous nous sommes arrêtés à cette taille du document (3,5 Mo) à cause de la saturation de l'espace mémoire générée par la représentation DOM du document.

Les documents sont générés grâce à l'outil Toxgene[111]. Cet outil donne à l'utilisateur le contrôle total sur la structure et sur le contenu des documents XML qu'il produit. Il peut, par exemple, fixer le nombre moyen d'apparition des éléments dans le document. Il peut, également, contrôler la génération de littéraux CDATA (e.g, la longueur de la chaîne de caractère), etc.

Les documents que nous avons générés partagent une structure conforme à notre exemple de motivation décrit dans le chapitre III. La distribution est uniforme pour le contenu de chaque dossier médical. Nous avons fait varier le nombre de dossiers medicaux dans le document. La Figure 38 présente un extrait de nos bases de test[26]. Le prototype a été implémenté en java en utilisant APIDOM, Xerces 1.2.1 comme constructeur de document DOM, et Xalan (1.x.x)[27] comme évaluateur de requête XPath.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.3 U (http://www.xmlspy.com) by boisson (gandalf) -->
<!-- generated by ToXgene Version 2.2 on Mon Nov 08 16:05:31 GMT 2004 -->
<root>
    <Service>
        <Folder>
            <Admin>
                <SSN>6592697667</SSN>
                <tel>ruthl</tel>
                <tel>beans</tel>
                <fax>even </fax>
                <email>sly, ironic decoys behind the somas try </email>
                <address>blithely daring escapades will boost rut</address>
                <comments>bravely ironic sheaves boost fluffily outside the gifts!blithe, blithe
                    dinos hang boldly silent multipliers;warthogs of the regularly busy
                    forges lose closely bravely bold asymptotes:bravely silent
                    de</comments>
                <Fname>idly thin sauternes with the idly busy b</Fname>
                <Lname>ironic forges shall solve bravely somas:</Lname>
                <Lname>warthogs do promise!furious warhorses am</Lname>
                <Age>33</Age>

            </Admin>

            <MedActs>
                <Act>
                    <RPhys>silently fluffy epitaphs against the sly</RPhys>
                    <Date>109</Date>
                    <Dummy>caref</Dummy>
                    <Dummy>foxes</Dummy>
                    <Presc>enticing, furious pinto shall have to pr</Presc>
                    <Notes>brave sheaves outsid</Notes>
                    <VitalSigns>7865607471</VitalSigns>
                    <Detail>
                        <Sympt>2250187919</Sympt>
```

[26] Le contenu du document est généré d'une manière aléatoire. Ce qui justifie un contenu incompréhensible.

[27] Il s'agit d'une API " *propriétaire* " adaptée par l'équipe de Damiani à partir des exemples fournis par Xalan.

```
                                        <Diag>8571666728</Diag>
                                        <Comments> *-iqB(/B]9v$3t|][Q= Tim3]aoEmM}dL4~8
                                        @;@#MpoGLSa-#3z r </Comments>
                                    </Detail>
                                </Act>
                                <Protocol>
                                    <Act>
                                        <RPhys>silently fluffy epitaphs against the sly</RPhys>
                                        <Presc>gtd jhf, hubdb dfjnif to shall have to pr</Presc>
                                    </Act>
                                </Protocol>
                            </MedActs>
                            <Analysis>
                                <Comments>busy players instead of the final, slow realms wake at the blithe
                                    realms--silent tithes beside the slow gifts must are ruthless players--
                                    furious, stealthy asymptotes to the bravely slow tithes doze
                                    qu</Comments>
                                <LabResults>
                                    <G1>
                                        <G1A>200</G1A>
                                        <G1B>62</G1B>
                                        <G1C>84</G1C>
                                        <G1D>103</G1D>
                                        <G1E>16</G1E>
                                    </G1>
                                </LabResults>
                            </Analysis>
                        </Folder>
                    </Service>
                </root>
```

Figure 38 : Extrait des bases de test

3.2 Etude de performances

Nous présentons dans ce qui suit les conclusions obtenues en mesurant les performances de chacune des phases de l'algorithme décrit en section 2:

3.2.1 Démarche et conclusions

- ***Construction de l'arbre DOM (phase 1 et phase 5)*** : les tests ont été réalisés sur plusieurs documents de taille différente allant de 10 Knœuds jusqu'a 150 Knœuds. Comme cela était prévisible nous avons constaté que les performances de cette phase évoluent linéairement en fonction de la taille du document. Nous avons mesuré un ratio de 4.4ms/Knoeud.
- ***Les phases de marquage de l'arbre*** (***phases 2 et 6***) : cette phase consiste essentiellement en l'évaluation de requêtes XPath présentes dans les règles sur les nœuds ou les règles sur les associations. L'objectif consiste à marquer les nœuds ciblés par une expression XPATH (i.e . remplissage des vecteurs). Dans un premier temps, nous avons étudié l'impact des différents types de requêtes XPath possibles et ce quel que soit le type de règles d'autorisation. Nous avons fait plusieurs tests : (1) requêtes de chemin absolu vs chemin relatif (e.g, /root/service/folder/MedActs/detail vs //detail), (2) requêtes avec prédicats (e.g, MedAct vs //MedAct[./ssn]), (3) requêtes avec un degré de sélectivité différent

c'est-à-dire ciblant un nombre de nœuds plus ou moins grand. L'objectif était d'étudier si la complexité des règles avait un impact important sur les performances. Intuitivement, nous pensions que les requêtes sur les associations étaient plus complexes, donc plus coûteuses à évaluer. Les résultats des tests ont montré que le coût de l'exécution des requêtes XPath est indépendant de leur nature. Le coût global de cette phase dépend principalement du nombre total de règles d'autorisation participante dans la politique de contrôle d'accès plutôt qu'à leur nature. Ce coût s'avère prépondérant lorsque le nombre de règles est supérieur à 8.

– ***Les phases de résolution de conflits* (*phases 3 et 7***) : pour étudier l'impact de la phase 3 concernant les noeuds, nous avons testé, dans un premier temps, plusieurs combinaisons de règles d'autorisation en conflit et sans conflit (Figure 39). Dans les deux premiers cas, nous avons inversé l'ordre des règles positives et négatives par rapport à la hiérarchisation (sur les noeuds folder et détail). Aucun conflit n'existe dans le troisième cas. Dans le quatrième et le cinquième cas, nous avons essayé d'augmenter le nombre de règles en conflit sur le même nœud (détails).

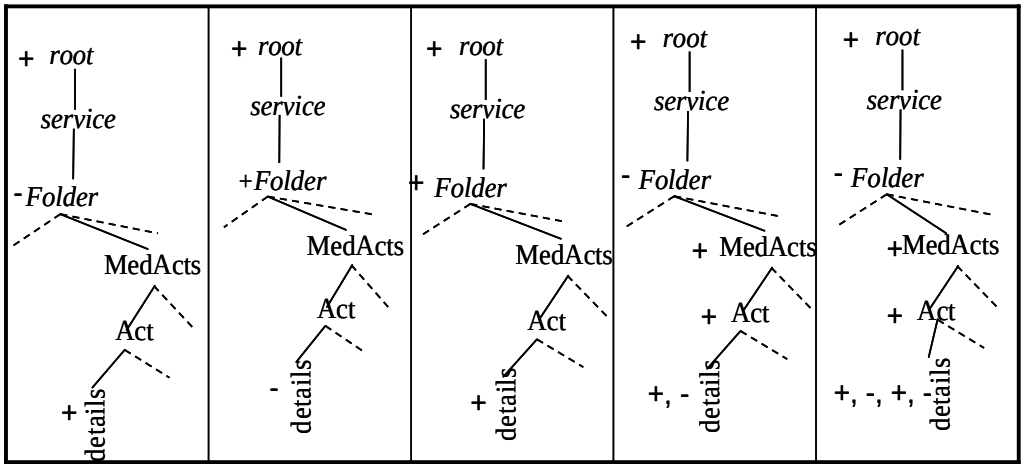


Figure 39 : Différentes combinaisons entre les règles d'autorisation

Nous avons constaté, d'après les tests, que l'ordre des règles négatives et des règles positives définies au niveau de la hiérarchie n'a aucun impact sur la résolution des conflits. Une légère augmentation du coût a été constatée lorsque *plusieurs* règles conflictuelles du même type (locale/récursive) sont définies de manière explicite sur le même nœud. Par exemple, <//Details, -, récursive >, <//Details, +, récursive >, <//Details, -, récursive > et <//Details, +, récursive > (dernier cas de notre figure).

Des tests complémentaires réalisés sur des documents de taille différente (allant de 10Knœud jusqu'à 150 Knœud), ont montré que le coût de cette phase était lié, plus particulièrement, au parcours de l'arbre DOM, ainsi qu'au parcours des

structures de données contenant le marquage. Le coût est linéaire en fonction du nombre de nœuds dans le document.

Pour la phase 7 concernant les associations, nous avons également fait des tests avec des règles conflictuelles au niveau de la *visibilité* et du *lien de fraternité*. Bien que les politiques de résolution de conflit des règles sur les associations soient sémantiquement plus complexes que celles sur les nœuds, la différence en terme de performance n'est pas significative. L'augmentation du coût total est de moins de 5%.

- ***La phase de suppression (phase 4)*** : les tests effectués font varier le nombre de nœuds supprimés : nous considérons la suppression de tous les nœuds de l'arbre <sergio, /Root, -> (c'est-à-dire la vue autorisée est vide); la suppression d'aucun nœud (c'est-à-dire la vue autorisée correspond au document source) <sergio, /Root, +>. Nous avons remarqué que la performance de cette phase évolue linéairement avec la taille du document. Elle peut être exprimée en terme d'un ratio de 0,4ms/Knœuds.
- ***La phase de reconstruction (phase 9)*** : puisque le rôle fondamental de cette phase consiste à créer des clones, nous avons fait varier le nombre de nœuds à dissocier et le nombre de nœuds à cloner. Nous avons remarqué que la phase de reconstruction dépend de la taille du document obtenu après la phase de suppression des nœuds, du nombre de nœuds ciblés par les règles d'autorisation sur les associations (i.e Desc) et du nombre de nœuds à cloner. Par exemple, le coût de la dépersonnalisation de toutes les Prescriptions (3640 éléments) de l'ancêtre *service médical* dans un document de 150Knoeuds est égal à 840ms. Cette règle engendre un très grand nombre de clones (14560 clones correspondant à 3640*4, 4 étant la profondeur du chemin cloné). Le coût de la reconstruction entre la dépersonnalisation des ancêtres et la réduction du chemin n'est pas significatif. La décorrélation sélective engendre une légère augmentation du coût liée au parcours de tous les frères du nœud descendant.

La Figure 40 montre les tests de performance de chacune des phases de l'algorithme ainsi que le coût supplémentaire lié à l'intégration des autorisations sur les associations. Afin de faciliter la compréhension des règles d'autorisation nous avons choisi de conserver les règles de notre exemple de motivation. Nous détaillons donc uniquement les chiffres correspondant aux politiques de contrôle d'accès des règles définies dans le chapitre IV (section 3.5). Les politiques R1, R2, et R3, rassemblent un ensemble de règles sur les nœuds et sur les associations. Les politiques sont évaluées sur des tailles de documents qui varient entre 50 Knœuds et 150 Knœuds. Afin de faciliter la lecture de la Figure 40, un curseur (à droite de chaque histogramme) sépare le coût induit par de l'évaluation des règles d'autorisation sur les nœuds AN (au-dessous du curseur) du coût supplémentaire induit par les règles d'autorisation sur les associations AA (au-dessus du curseur).

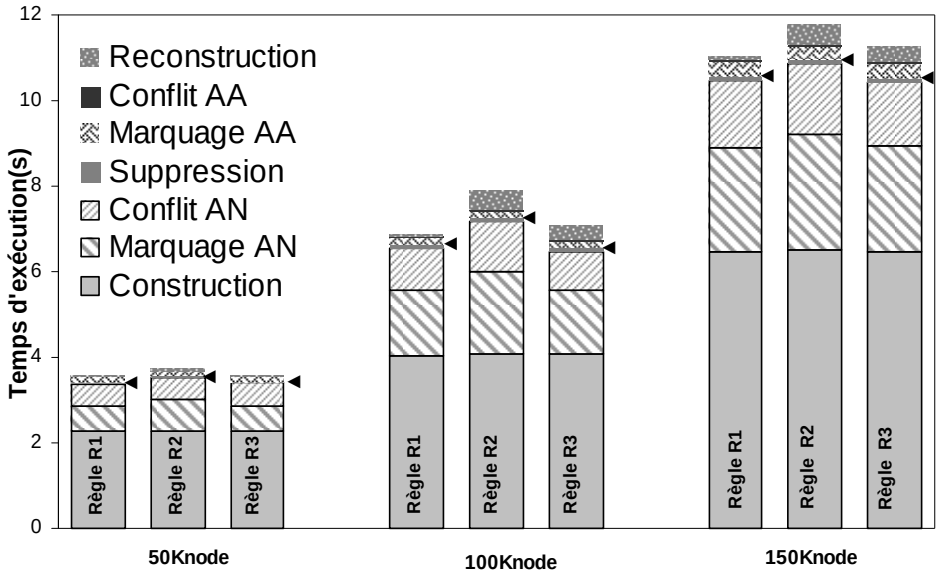


Figure 40 : L'impact des autorisations sur les associations de l'exemple de motivation

Le coût total de AN correspond aux coûts (1) de construction de l'arbre DOM, (2) du marquage de l'arbre par AN, (3) de la résolution de conflit sur les nœuds et (4) de la suppression des nœuds. Tandis que le coût supplémentaire du AA correspond aux coûts (1) du marquage de l'arbre par AA, (2) de la résolution de conflits sur les associations et (3) de la reconstruction de l'arbre.

D'après la Figure 40, nous remarquons que le coût de construction DOM domine le coût total de l'algorithme. Concernant la phase de reconstruction, c'est la règle R2 qui induit le coût le plus élevé. Cela est dû au coût engendré par le clonage important des nœuds. Les règles R1 et R3 ciblent le même nombre de nœuds descendants (120, 240, 360 pour respectivement 50 Knœuds, 100 Knœuds et 150 Knœuds) puisqu'il existe un seul élément `nom` dans chaque `Dossier`. La règle R2 quant à elle cible trois fois plus d'éléments puisque chaque élément `Protocole` (un par dossier) contient en moyenne trois éléments fils `Act`. L'opération de shuffling, en elle-même, n'est pas significative (ce n'est qu'un appel à une fonction aléatoire).

En général, une règle d'autorisation sur les nœuds est plus coûteuse que celle sur les associations. Cela due au coût de résolution de conflit sur les noeuds. Par contre, dans le cas de la décorrelation, le coût de règle d'autorisation sur les associations est proche de celle sur les nœuds.

3.2.2 Quelques détails des tests

Dans cette section, nous allons détailler la phase de **marquage de l'arbre**, car les résultats

nous ont un peu surpris dans la phase de tests.

Comme nous l'avons dit précédemment dans la section 3.3, le processus de marquage s'appuie principalement sur l'évaluation des requêtes XPath. Bien que les règles sur les nœuds soient différentes des règles sur les associations, en pratique ces deux phases sont totalement similaires en terme de marquage. En effet, pour chaque règle d'autorisation sur les associations nous évaluons une seule requête XPath qui cible les descendants. Donc quel que soit le type de règles, une seule requête Xpath est évaluée. C'est pourquoi dans ce qui suit, nous ne faisons plus la différence entre le marquage des nœuds et le marquage des associations.

Comme nous l'avons dit plus haut, nous nous sommes intéressés, dans un premier temps, à étudier l'impact des différents types de requêtes XPath possibles. Les résultats des tests ont montré que le coût d'exécution des requêtes XPath est indépendant de leur nature. Afin de ne pas répéter plusieurs fois les mêmes courbes, pour chaque type de requêtes, nous présentons, uniquement, deux courbes qui confirment nos conclusions. Les tests ont été faits sur un document de 25 Knœuds.

1. La Figure 41 présente le résultat d'une exécution répétée de deux requêtes XPath. Ces dernières diffèrent dans la longueur de leur chemin :
   - /*root* : cette requête est appelée *chemin_court*
   - */root/Service/Folder/MedActs* : cette requête est appelée *chemin_moyen*.

Nous constatons que le coût est indépendant de la longueur de chemin testé.

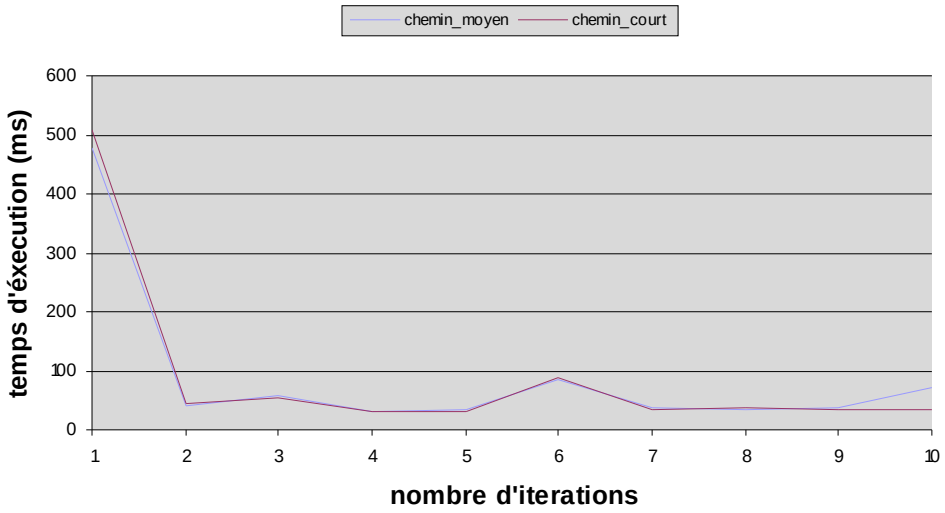


Figure 41 : Courbe d'exécution de deux requêtes différentes

2. La Figure 42 montre le résultat d'une exécution répétée de trois requêtes XPath différentes qui sont :

- //Service : *requête peu profonde* (i.e positionnement du nœud service par rapport à la racine root) qui cible ***peu de nœuds***.
- //Act : *requête peu précise* (le chemin exact conduisant à ce nœud n'est pas précisé) qui cible ***beaucoup de nœuds***.
- /Root/Service/Folder/MedActs//Act : *requête précise* (le chemin exact conduisant à ce nœud est plus précis) qui cible ***beaucoup de nœuds***.

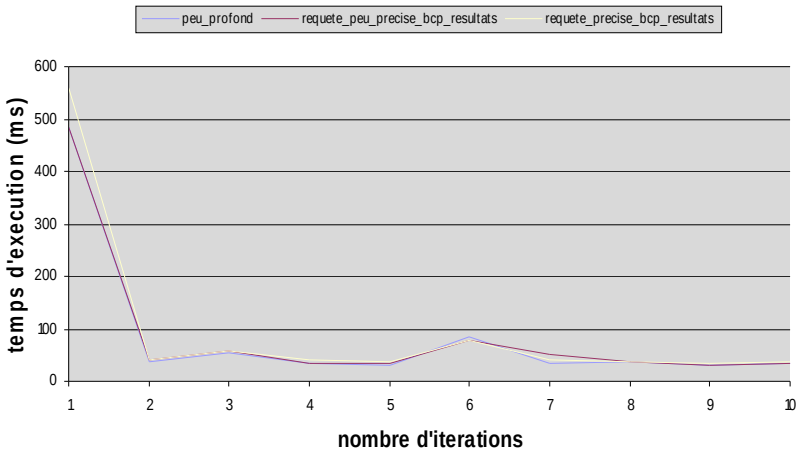


Figure 42 : Courbe d'exécution de trois requêtes différentes

Encore une fois, la courbe montre que le coût d'exécution est indépendant de la nature des requêtes Xpath, ainsi que du nombre de nœuds ciblés. Nous constatons également que les deux courbes sont exactement similaires.

Néanmoins, deux phénomènes remarquables ont été constatés :

- L'exécution de la première requête est toujours longue car au début du processus de marquage, il y a une phase d'initialisation de l'API Xpath indépendante de la politique de contrôle d'accès mise en œuvre.
- Les pics dans la durée de l'exécution des requêtes XPath correspondent au déclenchement intempestif du ramasse miette (garbage collector) de Java.

Le phénomène est resté identique malgré les différents tests réalisés avec des versions différentes de Xalan ou de Xerces.

Les informations recueillies dans les documentations Xalan nous fournissent des éléments d'explication de ces deux phénomènes. Afin d'exécuter une requête XPath, Xalan génère un grand nombre d'objets qui sont susceptibles d'êtres réutilisés, mais qui ne sont pas prévus dans ce sens. L'API Xalan étant conçue pour un usage unique, elle reconstruit son environnement à chaque appel.

Nous pouvons toutefois conclure à la vue des courbes ci-dessus que si lors de la première exécution, il y a génération de l'environnement de travail de Xalan (ce qui est conforme aux informations fournies par Apache), lors de la deuxième exécution et des exécutions suivantes, cet environnement est réutilisé au moins pour partie, et ce jusqu'aux pics que nous constatons dans les temps d'exécution successifs.

Ces pics s'expliquent alors par le fait que l'environnement de Xalan, très gourmand en mémoire et laissant derrière lui au fil de son exécution des objets non utilisés (provisoirement, ou du moins jusqu'au prochaine appel à l'API). Cela provoque le passage du ramasse miette (dans le graphe ci-dessus entre l'exécution de le n°4 et le n° 5) et oblige donc l'API Xalan à recréer des objets qui sont nécessaires et que le ramasse miette a éliminé.

Le pic entre l'exécution n°2 et n°3, plus petit et juste consécutif à la phase d'initialisation du contexte, peut s'expliquer de la même manière, ou bien correspond à l'activation d'objets supplémentaires rendus nécessaires par la répétition d'un traitement par le même contexte de travail.

Dans la section suivante, nous présentons les améliorations apportées, plus particulièrement, à cette phase de marquage.

4 Les améliorations apportées au prototype de base

Les tests effectués sur le prototype rendu public ont montré que les deux phases dominent en terme de coût sont les phases de construction et de marquage de l'arbre. Puisque ce prototype utilise les anciennes versions de Xerces et de Xalan, nous avons essayé, dans un premier temps, des versions plus récentes de Xerces (1.2.1 en 1.4.4) et de Xalan (1.x.x en 2.6.0). Malheureusement, aucune amélioration n'a été constatée au niveau des performances que ce soit au niveau de la construction de l'arbre DOM ou soit au niveau du marquage.

Par contre, l'utilisation de l'API de Xalan 2.6.0 ouvre une voie intéressante pour contrôler et réutiliser le contexte et les objets créés au moment de l'exécution de la première requête XPath. Cette réutilisation, dans les versions anciennes, se produit de manière anarchique et les objets sont susceptibles d'être libérés ou recréés au fil de l'exécution, ce qui explique les allongements constatés des temps d'exécution lors de l'exécution des requêtes répétitives et l'apparition des pics. Xalan offre dans sa version 2.6.0 une API alternative à XPathAPI, qui contrôle cette réutilisation de contexte et dont les fonctionnalités sont identiques à XPathAPI : il s'agit de ***CachedXPathAPI***. Afin de vérifier ce phénomène et de comparer le temps entre la durée d'exécution des requêtes avec cache et la durée d'exécution des requêtes sans cache, nous avons effectué des tests avec les mêmes séquences de requêtes XPath sur le même document (100 Knœuds) en utilisant les deux XPath API (avec cache et sans cache). Les résultats des tests sont schématisés dans les Figure 43.

Nous constatons encore fois que la première exécution est longue, d'une durée

d'exécution pratiquement similaire dans les deux situations (cache ou sans cache). Par contre, le résultat intéressant devient visible à partir de la deuxième exécution. Ce temps d'exécution devient faible et les pics disparaissent presque entièrement dans le cas de l'utilisation de cache. La figure montre une stabilité dans le temps d'exécution à partir de la deuxième requête XPath.

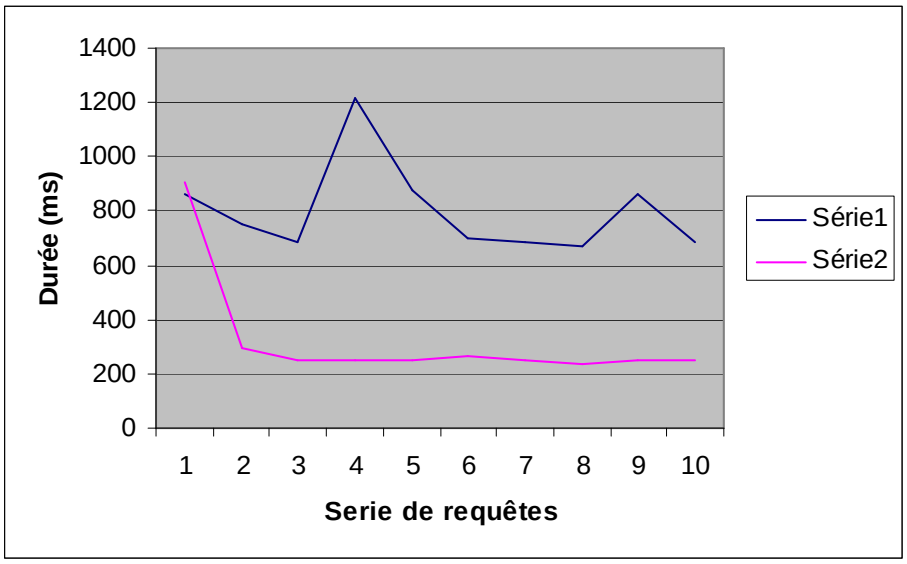


Figure 43 : Exécution des requêtes sans et avec l'utilisation de cache

La disparition des phénomènes de pics avec l'utilisation d'une API contrôlant la réutilisation des objets et du contexte confirme ainsi les caractéristiques fonctionnelles identifiées de XPathAPI.

- Une phase d'initialisation (ou précompilation) dans laquelle sont créés le contexte et les objets nécessaires à l'exécution des requêtes suivantes.
- Les requêtes XPath successives (identiques ou différentes) s'exécutent dans un temps plus petit qu'avec la version sans cache.

Nous rappelons, également, que ces constatations sont toujours indépendantes de la nature des requêtes exécutées.

De plus la phase de "précompilation" peut être dissociée de l'exécution des requêtes réelles décrites dans la base des règles d'autorisation. Elle peut être exécutée préalablement sur une requête arbitraire indépendamment des requêtes XPath réelles des règles d'autorisation. Par exemple au moment où l'utilisateur fournit son login pour accéder au document, une requête arbitraire est excecutée par le programme. La création du contexte se fera alors en temps masqué et le traitement des règles d'autorisation aura un temps faible.

5 Conclusion

Dans ce chapitre, nous avons décrit notre prototype et bien montré que notre modèle peut être vu comme une extension de modèles de contrôle d'accès XML existants, comme celui de Damiani disponible dans le domaine public. Nous avons, également, estimé que le coût de l'intégration des règles d'autorisation sur les associations reste très raisonnable. Des améliorations ont été apportées au prototype initial concernant la phase de marquage de l'arbre en utilisant cachedXpathAPI.

Chapitre VI – Sécurisation des documents XML basée sur le chiffrement

1 Introduction

L'étude effectuée dans ce chapitre est indépendante des chapitres précédents, elle concerne un aspect mentionné au chapitre III, la protection des données par des techniques de chiffrement. L'objectif est de pouvoir garantir une protection effective des données lorsque celles-ci sont gérées dans un environnement qui n'est pas totalement de confiance.

Traditionnellement, le partage des fichiers de données se fait grâce à un serveur de confiance à qui nous déléguons la gestion des droits d'accès. Cependant, ces serveurs sont de plus en plus souvent attaqués par des pirates ou, d'une manière plus grave, par les administrateurs de bases de données [25]. Il est donc difficile de faire confiance à un système de gestion de données dans ces conditions. Si nous n'avons pas confiance dans le serveur de données, nous ne pouvons pas stocker en clair les données, et encore moins lui faire confiance pour qu'il réalise la gestion des droits d'accès. La seule façon de protéger ces données contre toute attaque malveillante "*interne et externe*" est de chiffrer les données avant de les déposer sur le serveur de non-confiance et d'effectuer la gestion des droits d'accès côté client. La question qui se pose alors est comment traduire une politique de contrôle d'accès par des règles de chiffrement et de distribution des clés de telle sorte que chaque utilisateur déchiffre uniquement les parties qui lui sont autorisées par la base de règles d'autorisation.

Dans la littérature plusieurs techniques ont été proposées : Bertino [11, 12], G.Miklau [87] et Ray [99, 100]. Les deux premières l'ont été dans un contexte XML, pas la dernière. Le choix d'étudier ces trois modèles de contrôle d'accès est lié aux approches très différentes utilisées pour chiffrer le document. Bien que le modèle de Ray n'ait pas été proposé dans le cadre de XML, nous l'avons adapté. L'analyse approfondie de ces modèles a permis de mettre en évidence certaines carences et contre-exemples qui montrent que ces modèles peuvent être mis en échec dans certains cas. Afin de résoudre ces problèmes, nous proposons des améliorations de l'algorithme de chiffrement.

Les approches existantes permettent d'exprimer une situation de partage à l'instant *t* et sont donc très statiques. Dans une deuxième partie, nous étudions l'impact des mises à jour

afin de faire face au contexte plus dynamique du dossier médical. Une première ébauche de solution est proposée afin de limiter le rechiffrement tout ou partie du document.

Le chapitre est organisé comme suit. Dans une deuxième section, nous abordons les problèmes liés à la mise en œuvre d'un modèle de contrôle d'accès XML à base de chiffrement. Nous dressons tout d'abord, l'état de l'art des techniques existantes, puis nous donnons leurs limites, enfin nous présentons notre approche. Dans une troisième section, nous abordons les problèmes liés à la mise à jour de la politique de contrôle d'accès et étudions l'impact sur les techniques de chiffrement. Nous terminons par une conclusion.

2 Modèles de contrôle d'accès pour XML à base de chiffrement

La mise en œuvre de techniques de chiffrement pour la gestion des droits d'accès se résume en trois points : i) identifier les parties du document à chiffrer avec des clés différentes, ii) identifier les clés à distribuer aux utilisateurs appropriés afin que chaque utilisateur ne puisse déchiffrer que les parties qui lui sont, initialement, autorisées par la base de règles d'autorisation, iii) choix du mode de distribution des clés de déchiffrement aux utilisateurs appropriés.

Nous avons retenu pour notre étude trois approches car elles sont représentatives des techniques mises en œuvre dans la littérature. Le modèle de Bertino [11, 12] génère pour chaque ensemble de nœuds partageant le même ensemble de règles d'autorisation une clé différente de chiffrement. Le modèle de Ray [99, 100] est basé sur la théorie des clés compatibles. Le modèle de G.Miklau [87] adopte une technique de chiffrement différente des autres, la génération des clés de chiffrement est faite à partir de certaines valeurs contenues dans le document XML. Afin de comprendre plus en détail ces différents modèles, nous commençons par rappeler les définitions liés aux techniques de chiffrement.

2.1.1 Préliminaires

Différentes techniques de chiffrement existent dans la littérature :

- *Chiffrement symétrique* : consiste à utiliser la même clé pour le chiffrement et pour le déchiffrement. Cette clé doit être communiquée de manière sécurisée de l'émetteur au destinataire. Ce type de chiffrement est appelé, également, chiffrement à clé secrète ou chiffrement à clé privée [58, 84, 71].
- *Chiffrement Asymétrique :* consiste à utiliser une paire de clés. Une clé publique pour le chiffrement et une clé privée pour le déchiffrement. Lorsqu'un utilisateur désire envoyer un message, par exemple, à un autre utilisateur, il lui suffit de chiffrer le message à envoyer au moyen de la clé publique du destinataire (qu'il trouvera par exemple dans un serveur de clés tel qu'un annuaire LDAP). Ce dernier déchiffrera le message à l'aide de sa clé privée (qu'il est seul à connaître) [58, 84, 71].

- *Chiffrement à base de la théorie des clés compatibles :* ce type de chiffrement est basé sur le chiffrement asymétrique. L'idée de base est de combiner plusieurs clés de chiffrement pour chiffrer une donnée. Par exemple, pour deux paires de clés différentes : (K_i, K_i^{-1}) et (K_j, K_j^{-1}), la donnée pourra être chiffrée par $K_i \times K_j$ (K_i étant compatible avec K_j). Par conséquent, et d'après la théorie des clés compatibles, la donnée pourra être déchiffrée par Ki^{-1} et K_j^{-1} [99].

2.1.2 Modèle de contrôle d'accès de Bertino et al

Le chiffrement du document XML dans le modèle de Bertino [11, 12] repose sur le processus de marquage du document décrit au chapitre II. Le but de ce marquage est d'identifier les nœuds (éléments/attributs) ayant le même *ensemble de règles d'autorisation*. Chaque ensemble de nœuds partageant le même ensemble est chiffré par une même clé. Les utilisateurs reçoivent, par la suite, les clés de déchiffrement appropriées.

Dans ce qui suit, nous illustrons, brièvement, le processus de chiffrement par un exemple. La Figure 44 présente un exemple simplifié de base de règles d'autorisation[28] régulant l'accès au document XML présenté dans la Figure 45. La base de règle indique que tous les utilisateurs (secrétaire, médecin, laboratoire d'analyses) ont le droit d'accéder à toutes les informations de l'*Hôpital*. La secrétaire n'a pas le droit d'accéder aux informations concernant les *actes médicaux*. Le laboratoire d'analyses n'a pas le droit d'accéder aux éléments *commentaire* et *prescription*.

< ----------base de règles ---------- >
R_1 = </utilisateurs, /Hôpital, + >
R_2 = </secrétaire, //actsMed, - >
R_3 = </LabAnalyse, //commentaire, - >
R_4 = < /LabAnalyse, //prescription, ->
< ----------base de règles ---------- >

Figure 44 : Exemple de base de règles d'autorisation

Le processus de chiffrement se déroule en deux étapes décrites ci-dessous :

1. ***Première étape*** : consiste à marquer les nœuds par l'ensemble de règles d'autorisation qui s'appliquent sur eux. Le marquage du document est présenté dans la Figure 45:

[28] La politique de propagation est cascade pour toutes les règles.

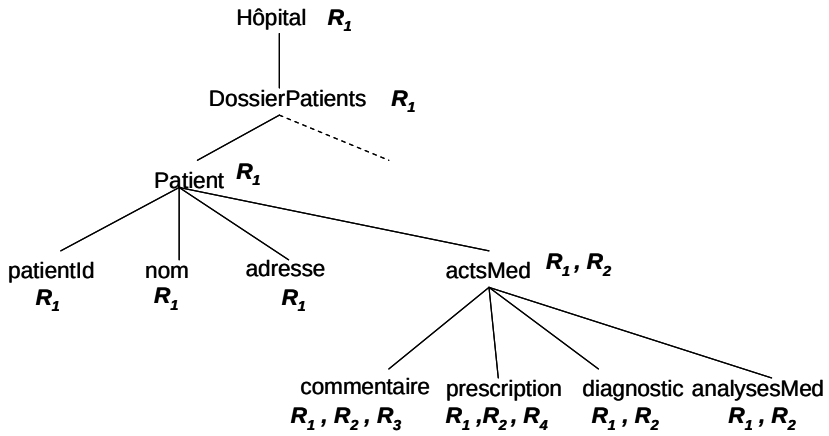


Figure 45 : Document XML marqué

2. ***Deuxième étape*** : consiste à identifier les groupes d'éléments-attributs partageant le même ensemble de règles d'autorisation. Pour chaque groupe identifié, une clé différente sera générée. Par exemple, les éléments actsMed, disgnostic, analysesMed forment un groupe car ils partagent le même ensemble de règle d'autorisation $\{R_1, R_2\}$. Le Tableau 5 résume l'ensemble des groupes d'éléments[29] identifiés dans le document XML présenté dans la Figure 45.

Membres du groupe	Clés
Hôpital, dossierPatients, Patient, PatientId, nom, adresse	K_1
actsMed, diagnostic, analysesMed	K_2
commentaire	K_3
Prescription	K_4

Tableau 5: Table de groupes

Ensuite, une table de clés est générée ; elle identifie pour chaque règle d'autorisation les clés correspondantes.

L'identification des clés pour chaque règle se fait de la manière suivante. Pour chaque règle, les groupes d'éléménts auxquel elle s'applique sont identifiés. Ensuite, les clés des groupes identifiés sont attribuées à la règle correspondante. A titre d'exemple, la règle R_1

[29] En réalité le regroupement habituellement se fait sur l'identifiant unique du nœud, mais afin de faciliter la lecture des tableaux nous avons préféré de garder les noms des tags.

s'applique sur tout le document, donc elle concerne tous les groupes d'éléments. Les clés correspondantes à cette règle sont les clés de tous les groupes {K_1, K_2, K_3, K_4}. Par contre, la règle R_4 cible uniquement le groupe contenant l'élément *prescription*. La clé nécessaire, donc, à cette règle est K_4. Le même raisonnement pour les autres règles d'autorisation. La table des clés résultante est présentée dans Tableau 6.

Règles	Composants	Clés
R_1	Hôpital, dossierPatients, Patient, PatientId, nom, adresse, actsMed, diagnostic, analysesMed, commentaire, Prescription	{K_1, K_2, K_3, K_4}
R_2	actsMed, diagnostic, analysesMed, commentaire, Prescription	{K_2, K_3, K_4}
R_3	commentaire	{K_3}
R_4	Prescription	{K_4}

Tableau 6 : Table de clés

2.2 Les limites du modèle de Bertino

La présence des règles d'autorisation négatives dans la base de règle amène à deux problèmes importants :

1. ***Distribution des clés de déchiffrement*** : la distribution des clés s'effectue en fonction de la table des clés et des règles d'autorisation qui s'appliquent aux utilisateurs. Chaque utilisateur reçoit les clés qui correspondent à ses règles d'autorisation. Dans le cas où la base de règle contient uniquement des règles d'autorisation positives, la distribution des clés ne pose aucun problème (chaque règle d'autorisation correspond à l'attribution des clés). Par contre, la présence des règles d'autorisation positives et négatives dans la base de règle pose certains problèmes non éclaircis. Le principe proposé ne prend pas en compte la gestion des conflits entre les règles d'autorisation. Nous illustrons ce problème par un exemple. La base de règle d'autorisation indique que la secrétaire a le droit d'accéder à tout le dossier médical du patient sauf aux actes médicaux (règles R_1 et R_2). D'après la table des clés, la secrétaire reçoit les clés {K_1, K_2, K_3, K_4} alors qu'elle ne devrait recevoir uniquement la clé K_1. Une solution possible pour résoudre le problème consiste à gérer les conflits en moment de la distribution des clés.

2. ***Problème de chiffrement :*** nous illustrons ce problème par un exemple simple

démontrant que le principe proposé de chiffrement à base de règles peut engendrer des incohérences en cas de document dont la structure est récursive (Figure 46). Supposons deux règles d'autorisation R_1 : <//A, +> et R_2 : <//B, -> pour le même sujet. Le marquage par ces deux règles d'autorisation montre que les deux éléments A et B marqués par $\{R_1, R_2\}$ forment un seul groupe d'éléments qui va être chiffré par une clé. Cependant on note que ce groupe d'éléments est incompatible car il contient à la fois un élément accessible et un élément inaccessible pour le même sujet. Au niveau de l'élément B (nœud n_2), la règle R_2 est prioritaire (l'accès est interdit) ; alors qu'au niveau de l'élément A (nœud n_4), la règle R_1 est prioritaire (l'accès est autorisé). D'où, l'identification des groupes d'éléments à chiffrer avec la même clé, en présence des règles d'autorisation positives et des règles d'autorisation négatives, doit se faire avec prudence.

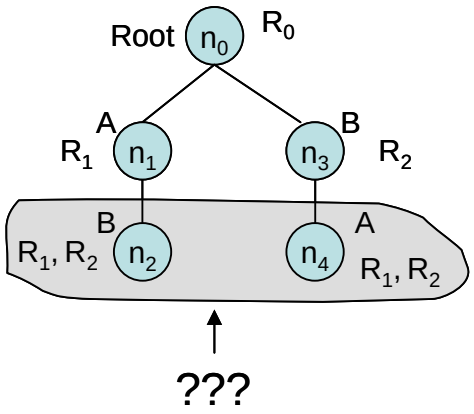


Figure 46 : Exemple d'un document XML marqué

Pour cette raison, dans la section suivante, nous proposons une solution possible pour résoudre le problème de chiffrement découvert dans le modèle de Bertino. Ensuite, nous précisons la façon d'identifier les clés à distribuer aux différents utilisateurs. Ces clés permettent aux utilisateurs de déchiffrer les parties qui leur sont autorisées au départ par la base de règles d'autorisation.

2.3 La solution proposée

Notre solution est basée, également, sur le processus de marquage. Chaque nœud sera marqué par l'ensemble de règle d'autorisation qui s'applique sur lui. Contrairement, à l'approche de Bertino, les règles d'autorisation définies sur un nœud sont ordonnées par ordre de priorité en appliquant le principe "*l'objet le plus spécifique est plus prioritaire*"[30]. Dans notre approche, deux ensembles de règles sont identiques si seulement si ***l'ordre*** des règles dans les deux ensembles est ***identique***. Ensuite, les nœuds ayant le même ensemble de

[30] Cela se fait indépendamment des sujets.

règles (***identique***) forme un groupe. Ces nœuds vont être chiffrés, ensuite, par la même clé.

Pour cela, notre solution repose sur le traitement des profondeurs des règles d'autorisation appliquées sur le document au moment du marquage. Cette approche sera appelée dans la suite de ce chapitre par "***l'approche à base de profondeur***". Ce traitement nous permet de gérer à la fois les priorités et de résoudre les conflits entre différentes règles spécifiées sur un nœud donné. Notre processus de chiffrement s'effectue en plusieurs étapes décrites ci-dessous :

1. ***Première étape*** : consiste à marquer les nœuds par l'ensemble des règles d'autorisation qui les concernent. Pour cela, si une règle d'autorisation R_i est définie d'une manière explicite sur un nœud n_j, alors le nœud n_j est marqué par R^0_i. "0" signifie que la règle n'est pas héritée d'un niveau supérieur dans la hiérarchie XML. Si l'option de propagation de la règle R_i est *cascade* alors tous les descendants de n_j sont marqués d'une manière récursive par R^{-k}_i ; "*k*" représente le niveau de ces descendants par rapport à n_j. Par exemple, les descendants directs nd_l du nœud n_j sont marqués par R^{-1}_i et d'une façon récursive les descendants directs du nd_l sont marqués par R^{-2}_i…etc. Le marquage du document de la Figure 46 est décrit en Figure 47 :

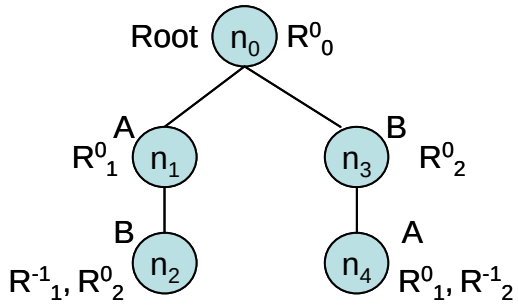


Figure 47 : Nouveau marquage du document XML

2. ***Deuxième étape*** **:** consiste à ordonner l'ensemble de règle d'autorisation par *ordre de priorité* en se basant sur l'indice de profondeur *k*. L'ordonnancement des règles se fait, donc, par ordre décroissant par rapport à l'indice *k*, ce qui veut dire que pour un utilisateur donné, c'est toujours la première règle dans l'ensemble de règles qui le concerne qui l'emporte. Le marquage final, après ordonnancement, est présenté dans la Figure 48. Dans notre approche, deux ensembles de règles sont identiques si seulement si ***l'ordre*** des règles dans les deux ensembles est ***identique***. Par exemple, d'après la Figure 48, les éléments A (nœud n_4) et B (nœud n_2) ont le même ensemble de règles $\{R_1, R_2\}$ mais ils ne sont pas identiques car l'ordre des règles dans l'ensemble est différent. Donc, ces deux éléments doivent être chiffrés par des clés de chiffrement différentes.

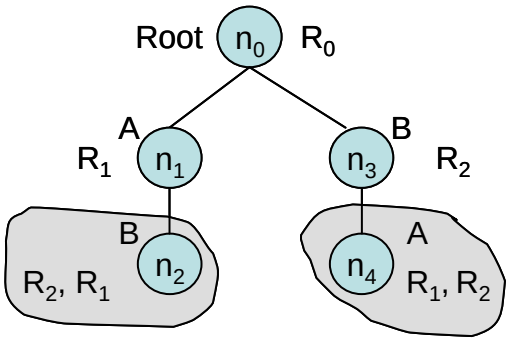


Figure 48 : Marquage final du document XML

3. ***Troisième étape*** : consiste à générer la table de marquage qui regroupe les éléments qui doivent être chiffrés par la même clé de chiffrement.

Groupes d'éléments	Ensemble de règles	Clés de chiffrement
n_0	R_0	k_0
n_1	R_1	k_1
n_2	R_2, R_1	k_2
n_3	R_2	k_3
n_4	R_1, R_2	k_4

Figure 49 : Table de clés

4. ***Quatrième étape*** : consiste à identifier les clés à envoyer aux utilisateurs afin qu'ils déchiffrent les parties qui leur sont autorisées par la base de règles d'autorisation. Pour cela, dans un premier temps, nous identifions quelles sont les règles d'autorisation s'appliquant à un utilisateur donné. Ensuite, nous parcourons séquentiellement la table de marquage pour identifier les parties concernées par les règles de cet utilisateur afin d'extraire les clés appropriées. L'extraction des clés se fait comme suit : puisque les règles sont ordonnées par ordre de priorité, la résolution de conflit se fait d'une façon automatique. La première règle dans l'ensemble des règles qui concerne cet utilisateur l'emporte toujours. Par exemple, si l'utilisateur est concerné par les règles R_1 et R_2, alors d'après la table de marquage, il reçoit la clé k_1 car il est concerné par le deuxième groupe d'éléments. Il ne peut pas recevoir la clé k_2 car dans le troisième groupe d'éléments la règle R_2 précède la règle R_1 (R_2 est donc prioritaire) et puisque R_2 est négative l'utilisateur ne peut recevoir cette clé. On

continue ainsi de suite le procédé sur toute la table de marquage. Au final, l'utilisateur reçoit les clés k_1 et k_4.

Dans la littérature, une extension au modèle de Bertino avec support des règles d'autorisation négatives a été proposée par Carminati [24]. Cependant le processus de chiffrement utilisé n'est pas clairement défini dans leur rapport technique. L'idée de base repose sur un marquage du document par un ensemble de règles d'autorisation actives. Une règle d'autorisation active correspond à une règle qui n'a pas été désactivée par une autre règle conflictuelle plus prioritaire. Comme nous le montrons par un exemple simple, ce marquage peut engendrer des problèmes au moment de la diffusion des clés dans le cas de l'inclusion des sujets. Supposons la base de règle présentée dans la Figure 51. Le groupe utilisateurs est composé de secrétaire, infirmière et médecin.

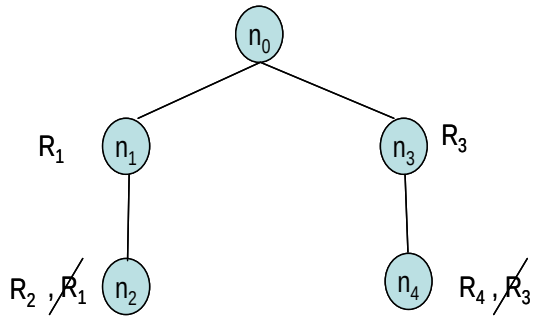


Figure 50 : Document marqué avec les règles actives

R_1 : <utilisateurs, n1, +>

R_2 : <secrétaire, n2, ->

R_3 : <secrétaire, n3, +>

R_4 : <utilisateurs, n_4, ->

Figure 51 : Base de règles d'autorisation

Nous remarquons que la règle R_1 a été désactivée par la règle par la règle R_2 pour le nœud n_2. La règle R_3 a été désactivée par la règle R_4 pour le nœud n_4. La désactivation de R_3 ne pose aucun problème puisque la secrétaire est incluse dans le groupe utilisateur. La secrétaire, implicitement[31], est concernée par la règle R_4 qui désactive la règle R_3. Par contre, la désactivation de la règle R_1 sur le nœud n_2 pose des problèmes. Car la règle R_1 reste toujours active pour les autres membres du groupe infirmière et médecin qui ont normalement l'accès au nœud n_2.

Nous ne pouvons donc pas parlé de règles actives et inactives dans l'absolu. L'activation des règles est propre à un ***sujet*** bien particulier. Par contre, le chiffrement du document se fait pour l'ensemble des sujets. Cela montre donc qu'on ne peut pas désactiver une règle d'autorisation en s'appuyant uniquement sur les nœuds du document.

D'après notre étude, nous avons remarqué que la combinaison des règles d'autorisation positives et des règles d'autorisation négatives rend le modèle de contrôle d'accès à base de chiffrement plus compliqué. La complexité du modèle se situe non seulement au moment du chiffrement (identification des groupes d'éléments qui vont être chiffrés par la même clé),

[31] Car elle appartienne au groupe des utilisateurs.

mais, également, au niveau de l'identification des clés appropriées à chaque utilisateur. Cette identification des clés doit se faire avec une très grande prudence.

Une fois les clés identifiées, il faut choisir un mode de diffusion de ces clés. Dans la littérature, il existe deux modes de diffusion des clés : i) *le mode on line* : les clés de déchiffrement sont envoyées avec le document chiffré ou par email sécurisé pour chaque utilisateur ; ii) *le mode off-line* : les clés sont stockées dans une base de données de confiance ou dans un annuaire LDAP, ensuite chaque utilisateur récupère ses clés au fur et à mesure de ses besoins. Le choix d'une méthode de diffusion dépend de plusieurs facteurs comme le nombre d'utilisateurs, le nombre de clés générées pour chiffrer le document, des préférences explicites des utilisateurs (dans le cas où les utilisateurs exigent un mode de diffusion bien particulièr)…etc.

2.3.1 Modèle de contrôle d'accès basé sur l'approche de Ray

L'approche de Ray [99, 100] est basée sur la théorie des ***clés compatibles*** pour chiffrer les données. Chaque donnée est chiffrée par l'ensemble des clés des utilisateurs ayant le droit d'y accéder. L'objectif principal de la théorie des clés compatibles est que chaque utilisateur a en sa possession *une et une seule clé* lui permet de déchiffrer toutes les données auxquelles il a droit. Il est important de noter que la théorie des clés compatibles n'a pas été étudiée auparavant dans le cadre des documents XML. Dans ce qui suit, nous proposons une adaptation de la théorie des clés compatibles pour les documents XML. Contrairement à l'approche présentée précédemment, le marquage du document se fait par un ensemble d'utilisateurs. Le processus de chiffrement que nous pouvons imaginer se fait, également, en plusieurs étapes :

1. ***Génération des clés*** : Consiste à générer une paire de clés (K, K^{-1}) pour chaque utilisateur. K représente la clé de chiffrement et K^{-1} représente la clé de déchiffrement. Par exemple, nous avons trois utilisateurs secrétaire (S), Médecin (M) et LabAnalyse (L).

Utilisateurs	Clés de chiffrement	Clés de déchiffrement
Secrétaire (S)	K_s	K^{-1}_s
Médecin (M)	K_M	K^{-1}_M
LabAnalyse(L)	K_L	K^{-1}_L

Figure 52 : Table de clés des utilisateurs

2. ***Processus de marquage du document XML*** : consiste à marquer les nœuds par l'ensemble des utilisateurs ayant le droit d'y accéder. Pour cela, nous pouvons imaginer, dans un premier temps, une étape intermédiaire qui consiste à marquer

les nœuds par l'ensemble des règles d'autorisation tel que nous l'avons défini dans notre approche. Notons cependant que la résolution des conflits entre les règles doit se faire dès le marquage et non au moment de la diffusion des clés. Par exemple, l'élément *commentaire* est marqué par l'ensemble de règles {R_3, R_2, R_1}. La règle R_3 s'applique sur LabAnalyse et donc interdit l'accès à l'élément commentaire. La règle R_2 s'applique sur la secrétaire et donc lui interdit, également, l'accès à l'élément commentaire. La règle R_1 donne l'accès à tous les utilisateurs. Comme les deux premières règles sont plus prioritaires, les deux utilisateurs secrétaire et LabAnalyse vont être supprimés de la liste des utilisateurs ayant accès à cet élément. Une fois les conflits résolus nous obtiendrons le marquage représenté dans la Figure 53.

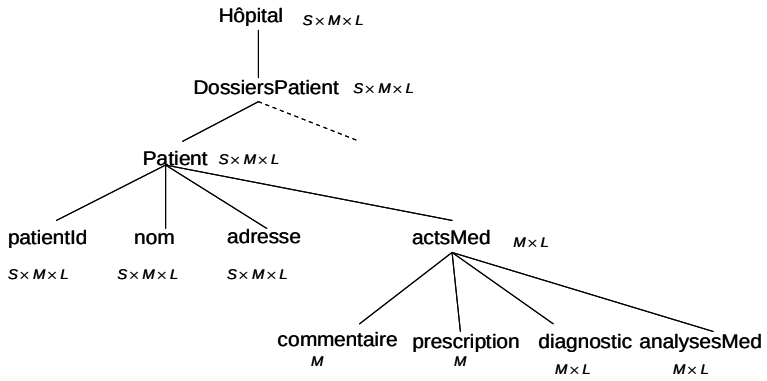


Figure 53 : Document marqué par l'ensemble des utilisateurs

3. ***Processus de chiffrement* :** chaque groupe d'éléments est chiffré par l'ensemble des clés des utilisateurs ayant le droit d'y accéder.

Groupes d'éléments	**Ensemble d'utilisateurs**	**Clés de chiffrement**
Hôpital, DossiersPatient, Patient, PatientId, nom, adresse	S, M, L	$K_S \times K_M \times K_L$
actsMed, diagnostic, analysesMed	M, L	$K_M \times K_L$
Commentaire, Prescription	M	K_M

Figure 54 : Table des clés

L'avantage de cette approche est que la résolution de conflits se fait au moment de marquage. Donc aucun problème au niveau de chiffrement. La gestion des clés compatible permet de réduire le nombre de clé de l'utilisateur. Par contre, elle ne réduit pas le nombre de

clé à considérer dans le système globale.

2.3.2 Modèle de contrôle d'accès basé sur l'approche de Miklau

Traditionnellement, la définition des règles de contrôle d'accès est basée sur la notion de sujet, objets et accès. Miklau [87] définit un modèle de contrôle d'accès indépendamment des sujets et propose un langage propre pour définir les règles de contrôle d'accès. Le modèle est basé sur un ensemble de *règles d'accès conditionnelles*[32]. Les accès aux données ne sont pas déterminés en fonction des sujets (c'est-à-dire en fonction de leur id ou leur position organisationnelle) mais, plutôt, en fonction de leurs propres ***connaissances***. En d'autres termes, l'accès aux données est basé sur un ensemble de règles d'accès conditionnelles spécifiant les valeurs qui doivent être fournies par les utilisateurs pour accéder à des parties du document XML. Nous décrivons, dans un premier temps, brièvement le principe de définition des règles de contrôle d'accès. Ensuite, nous présentons la technique de chiffrement utilisée.

Le modèle est basé sur un ensemble de règles d'accès conditionnelles appelées *CAR*s (*Conditional Access Rules*). Une règle d'accès conditionnelle r est représentée sous la forme suivante : $r ::= C : (\{B\} \rightarrow \{F\})$. Nous définissons, d'abord, les concepts de cette règle, ensuite nous donnons son intuition :

- C : représente le *contexte* de la règle d'autorisation identifié par une expression XPath absolue.
- $\{B\}$: représente l'ensemble des valeurs obligatoires défini sous forme d'expressions XPath relatives à partir du nœud contexte.
- $\{F\}$: représente l'ensemble des valeurs libres défini sous forme d'expressions XPath relatives à partir du nœud contexte.

L'intuition de cette règle d'accès est que l'accès au sous-arbre enraciné par le nœud *contexte* $c \in eval(C, D)$[33] doit être non autorisé. L'accès sera autorisé, uniquement, en fournissant les valeurs de B qui permettent, ensuite, d'accéder au sous-arbre porté par F. Le modèle permet également de gérer des exceptions en définissant d'autres nœuds contexte tel qu'il est schématisé dans la Figure 55.

[32] La notion d'accès conditionnel, dans ce modèle, généralise la catégorisation statique des sujets pour définir les politiques de contrôle d'accès

[33] $eval(C, D)$ désigne l'ensemble de nœuds résultant d'une évaluation de C a partir de la racine sur le document D.

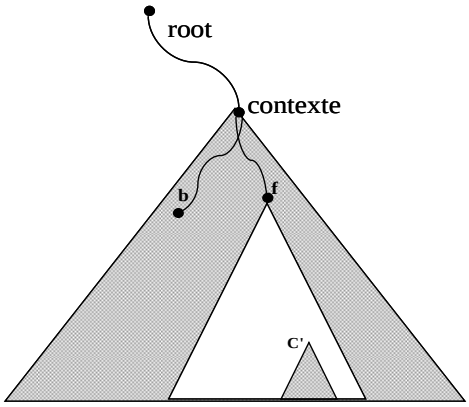


Figure 55 : Illustration de l'effet d'une règle CAR

À titre d'exemple, nous donnons un ensemble de règles conditionnelles CARs sur les dossiers médicaux présentés dans la Figure 56.

1. */Hôpital/DossierPatients/Patient* : ({*PatientId*}→{*pers/nom, pers/adresse*}) : la règle dit que, dans le sous arbre porté par le nœud contexte *patient*, l'accès aux nom et adresse est conditionné par la connaissance de l'identifiant du patient correspondant.

2. /Hôpital/DossierPatients/Patient :({pers/nom}→{med/chambreNum, med/étage}) : la règle dit que, dans le sous arbre porté par le nœud contexte *patient*, l'accès aux numéro de chambre et étage est conditionné par la connaissance du nom du patient correspondant

3. /Hôpital/DossierPatients/Patient : ({patientId, med/diagnostic}→{.}) : la règle dit que, l'accès aux informations du patient est conditionné par la connaissance du l'identifiant du patient et le diagnostic.

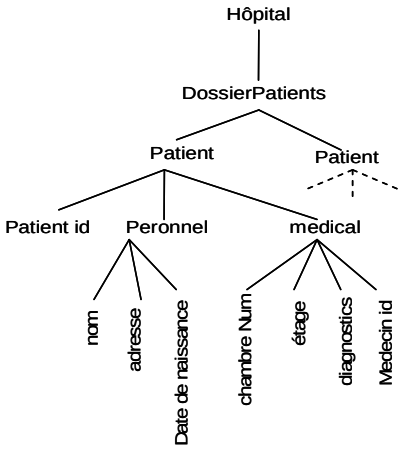


Figure 56 : Exemple de dossiers médicaux

La technique de chiffrement adoptée dans ce modèle est basée sur une technique de

chiffrement adopté dans un cadre relationnel, dont nous rappelons le principe d'une façon générale :

Soit T[A, B] une relation binaire. Alice souhaite publier cette relation, mais elle souhaite également en restreindre l'accès. L'utilisateur doit fournir, d'abord, les valeurs de *A* pour pouvoir extraire les valeurs de *B*. Cette règle de contrôle d'accès appelée *r*, est représentée sous la forme suivante $T : (A \rightarrow B)$. En utilisant les primitives de la cryptographie (la fonction de cryptage *E* et la fonction de hachage *f*), la table de cryptage (ou *access controlled* table) T_r^{ac} est définie comme suit :

$T_r^{ac} = \{(f(a), E_a(b)) \setminus (a, b) \in T\}$; les n-uplets de la table T_r^{ac} contiennent seulement du texte chiffré.

Par analogie, dans le cadre XML, si $r = C : (B \rightarrow F)$ est une règle de contrôle d'accès conditionnelle et *x* est un nœud contexte, la table binaire $T_r(x)$ est définie comme suit :

$T_r(x) = \{(b, f) \setminus b = b_1.b_2 \ldots b_k , f = f_1. f_2 \ldots f_l , bi \in eval(C/Bi, D), i \in [1 \ldots k], fi \in eval(C/F_j, D), j \in [1 \ldots l]\}$, avec *b* correspondent à la concaténation des valeurs obligatoires et *f* à la concaténation des valeurs libres.

Au moment de la construction du document chiffré D_R^{ac}, les nœuds qui ne sont pas des nœuds contextes restent inchangés dans D_R^{ac}. Par contre pour les autres nœuds, nous prenons chaque règle *r* et nous calculons pour chaque nœud contexte *x* la table $T_r(x)$. Ensuite le nœud contexte *x* et ses sous-arbres sont remplacés par une collection des tables $T_r(x)$ qui sont représentées sous forme XML.

Bien que Miklau [87] propose une approche intéressante de contrôle d'accès basée sur le chiffrement, leur modèle de contrôle d'accès souffre d'un inconvénient majeur. Cet inconvénient est lié plus particulièrement à l'absence des sujets dans les règles de contrôle d'accès. La gestion des droits d'accès est basée uniquement sur les connaissances de quelques valeurs de données dans le document XML. Il n'existe aucun moyen de contrôler les connaissances des personnes. Par conséquent, n'importe quelle personne peut avoir ces connaissances d'une façon ou d'une autre ; cela lui permettra d'accéder à certaines informations sensibles du patient d'une façon illégitime.

Nous avons présenté, dans cette section, différents modèles de contrôle d'accès à base de chiffrement. Ces modèles diffèrent dans la façon de chiffrer le document et les techniques de chiffrement utilisées. Dans la section suivante, nous allons étudier l'impact des différentes mises à jour sur le document chiffré dans chacun de ces modèles.

3 Impact des mises à jour sur le document chiffré

L'objectif de cette section est d'étudier l'impact des différents types de mises à jour sur le

document chiffré et d'analyser le comportement de chaque modèle présenté précédemment vis-à-vis de ces mises à jour. Ces dernières peuvent concerner les sujets, les objets (éléments/attributs) ou les règles d'autorisation. La mise à jour des sujets signifie l'ajout/suppression d'un utilisateur ou groupe d'utilisateurs. La mise à jour des objets concerne l'ajout/suppression ou modification d'un attribut, élément. Par contre, la mise à jour des règles d'autorisation consiste à mettre à jour la base de règle en ajoutant ou supprimant d'un n-uplet <***sujet***, ***objet***, ***signe, propagation*** >. À la fin de cette section, nous présentons une piste de réflexion pour protéger le document en utilisant les techniques de chiffrement tout en maintenant de bonnes propriétés ceci afin d'éviter des rechiffrements trop lourds.

3.1 Approche à base de profondeur

Nous étudions, dans ce qui suit, pour chaque type de mises à jour, leurs impacts sur le document chiffré.

3.1.1 Mise à jour des sujets

Nous étendons par la mise à jour des sujets, la mise à jour de la base des sujets sans modifier la base de règles d'autorisation. Cela correspond soit à l'ajout d'un utilisateur à groupe déjà existant, soit à l'ajout d'un utilisateur n'appartenant à aucun groupe prédéfini. Idem, la suppression d'un utilisateur peut être, soit une suppression d'un groupe, soit une suppression, tout simplement, de l'utilisateur n'appartenant à aucun groupe.

1. ***Ajout d'un utilisateur***

   L'ajout d'un utilisateur à la base de sujets nécessite l'interrogation de la table des clés pour identifier les clés qui le concernent. Cela peut se faire comme suit :

   a. Si l'utilisateur est ajouté au groupe déjà existant, il reçoit toutes les clés du groupe.

   b. Si l'utilisateur n'appartient à aucun groupe prédéfini, il reçoit, par conséquent, la clé définie pour la politique par défaut si elle existe.

2. ***Suppression d'un utilisateur*** **:** invalider les clés de cet utilisateur et re-chiffrer les parties concernées.

3.1.2 Mise à jour du document

Le chiffrement du document se fait en deux étapes : le marquage du document par les règles d'autorisation et la génération de la table des clés. Dans la suite de notre étude, nous supposons que la mise à jour est autorisée pour un utilisateur donné. La définition du modèle de contrôle d'accès avec mises à jour dépasse le cadre de notre étude.

Le chiffrement de l'élément/attribut ajouté dépend de l'ensemble des règles

d'autorisation du père. Pour chaque règle du père, nous vérifions l'option de propagation. Si l'option de propagation est cascade alors l'élément/attribut ajouté bénéficiera de cette règle. Une fois l'ensemble des règles du père traité, nous nous intéressons, ensuite, à l'ensemble des règles spécifiques à l'élément ajouté. Deux cas peuvent se présenter :

- Si l'ensemble des règles de l'élément/attribut ajouté existe déjà dans la table de clé alors nous ajoutons cet élément au groupe d'éléments correspondant. Par exemple, l'ajout d'un élément **Age** comme descendant de l'élément *Patient* (Figure 45) amène à ce que cet élément bénéficie de la règle R_1 du père. Il suffit, donc, de l'intégrer dans le premier groupe de la table (Tableau 7) et de le chiffrer par la clé du groupe (k_1).

Groupes d'élément	**Ensemble de règles**	**Clés de chiffrement**
Hôpital, DossiersPatient, Patient, nom, adresse, patientId, **Age**	R_1	k_1
actsMed, diagostic, AnalysesMed	R_2, R_1	k_2
Commentaire	R_3, R_2, R_1	k_3
Prescription	R_4, R_2, R_1	k_4

Tableau 7 : Table de clés

- Si l'ensemble de règles de l'élément/attribut n'existe pas dans la table de clés alors nous créons une nouvelle entrée dans la table et générons une nouvelle clé pour chiffrer l'élément/attribut. Ce cas peut survenir, par exemple, lorsque les règles du père sont no-propagate ce qui amène à avoir un ensemble de règle différent de celui du père.

Nous remarquons que dans certains cas l'ajout l'un élément nécessite la génération d'une nouvelle clé. Par contre, la suppression d'élément/attribut consiste, tout simplement, à le détruire et à le supprimer de la table des clés.

3.1.3 Mise à jour de la base de règles d'autorisation

La mise à jour de la base de règles d'autorisation consiste à ajouter ou à supprimer le 4-tupet < ***sujet, objet, signe, propagation*** >. Nous distinguons l'ajout d'une règle d'autorisation positive et l'ajout d'une règle négative. Idem, pour la suppression d'une règle d'autorisation. Pour cette raison, nous ne détaillons dans la suite que l'ajout d'une règle d'autorisation car le comportement de la suppression d'une règle est pratiquement similaire à celui de l'ajout. L'ajout d'une règle d'autorisation positive ou d'une règle d'autorisation négative influe

d'une façon considérable sur le document chiffré. Le problème est qu'une règle d'autorisation peut impacter un groupe d'éléments[34] qui a été chiffré par une seule clé. Ce groupe d'éléments initialement formé doit être divisé en deux sous groupes d'éléments et, par conséquent il faut les re-chiffrer avec des clés différentes. A titre d'exemple, si on se base sur l'exemple défini dans la Figure 45, l'ajout d'une règle d'autorisation R_5 sur l'élément *AnalysesMed* conduit à créer un nouveau groupe d'éléments (voir la Figure 57). L'élément *AnalysesMed* sera marqué par un nouvel ensemble de règles d'autorisation différent de celui de l'élément *actsMed* et *diagnostic*. Cela nécessite, donc, de générer une nouvelle clé et de re-chiffrer le document (voir Tableau 8).

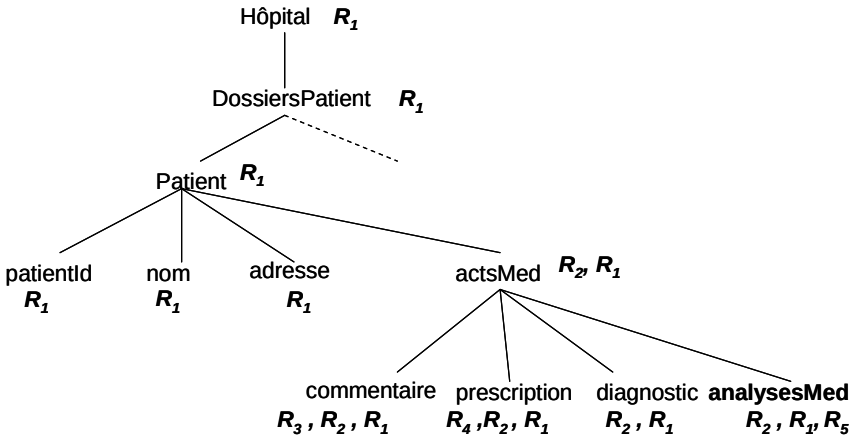


Figure 57 : Nouveau marquage du document XML

Groupes d'élément	**Ensemble de règles**	**Clés de chiffrement**
Hôpital, DossiersPatient, Patient, nom, adresse, patientId, **Age**	R_1	k_1
actsMed, diagostic	R_2, R_1	k_2
Commentaire	R_3, R_2, R_1	k_3
Prescription	R_4, R_2, R_1	k_4
AnalysesMed	R_2, R_1 R_5	k_5

Tableau 8 : Nouvelle table de clé

La mise à jour de la base de règle d'autorisation consiste, donc, à relancer le processus de marquage et à re-chiffrer le document car à chaque fois les priorités entre les règles

[34] Peut impacter, également, plusieurs groupes si la règle d'autorisation est récursive

existantes et la nouvelle règle ajoutée doivent être recalculées. Dans certains cas, l'ajout d'une règle d'autorisation peut de ne pas impacter le chiffrement initial du document. Par exemple, si on ajoute une règle positive donnant accès à tout le document[35]. Dans ce cas, le marquage est changé d'une façon uniforme et il suffit de diffuser les clés aux utilisateurs concernés par cette règle. Par contre, l'ajout d'une règle négative enlevant l'accès à tout le document[36] pose de grands problèmes. Bien que, le marquage ait été changé d'une manière uniforme, le document doit être re-chiffrer.

D'autres situations intermédiaires peuvent survenir et conduirent à re-chiffrer certaines portions du document et, par conséquent, à re-diffuser les clés aux utilisateurs concernés. Ces situations apparaissent lorsque la règle d'autorisation cible une partie du document.

3.2 Modèle de contrôle d'accès basé sur l'approche de Ray

Dans la théorie des clés compatibles, proposée par Ray, chaque utilisateur garde une et une seule clé lui permettant de déchiffrer les parties qui lui sont autorisées. La différence fondamentale entre l'approche précédente et celle de Ray se situe, plus particulièrement, au niveau de la *re-diffusion* des clés. Tout changement au niveau du chiffrement du document reste invisible pour les utilisateurs déjà existants. À titre d'exemple, si on se base sur l'exemple défini dans la Figure 45, l'ajout d'une règle d'autorisation R_5 donne l'accès au statisticien et conduit à créer un nouveau groupe d'éléments (voir la Figure 58), car l'élément *AnalysesMed* est marqué par un nouvel ensemble d'utilisateurs (voir le Tableau 9). Par contre, cette mise à jour reste invisible pour le médecin et le LabAnalyses qui avait déjà le droit d'accéder à cet élément.

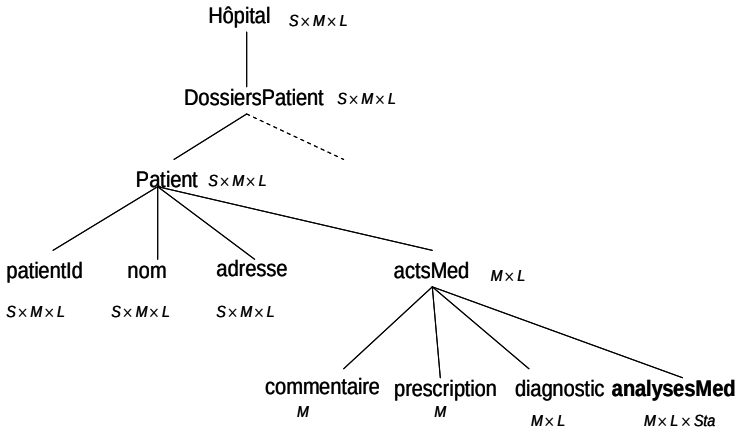


Figure 58 : Nouveau marquage du document

[35] À condition qu'il n'existe pas, auparavant, une autre règle négative plus prioritaire pour le même sujet.

[36] À condition qu'il n'existe pas avant une autre règle positive plus prioritaire pour le même sujet.

Groupes d'éléments	**Ensemble d'utilisateurs**	**Clés de chiffrement**
Hôpital, DossiersPatient, Patient, PatientId, nom, adresse	S, M, L	$K_S \times K_M \times K_L$
actsMed**,** diagnostic,	M, L	$K_M \times K_L$
Commentaire, Prescription	M	K_M
analysesMed	M, L, Sta	$K_M \times K_L \times K_{Sta}$

Tableau 9 : Table de clés

3.3 Le modèle de Miklau

Dans ce modèle, nous n'avons pas explicitement de mises à jour des sujets, car les accès aux données ne sont pas déterminés en fonction des sujets mais, plutôt, en fonction de leurs connaissances. En d'autres termes, l'accès aux données est basé sur un ensemble de règles d'accès conditionnelles spécifiant les valeurs qui doivent être fournies pour accéder à d'autres parties du document. Par conséquent, l'accès au document par le nouvel utilisateur dépend fortement de sa connaissance de quelques données dans le document.

La suppression d'un utilisateur n'est pas aussi facile, car un utilisateur n'est pas lié d'une façon explicite à une règle d'accès. Nous ne pouvons jamais lui effacer ***ses connaissances*** !. Dans ce cas, il faut, si c'est possible, définir d'autres règles d'accès conditionnelles et re-chiffrer le document et tous les autres utilisateurs vont être perturbés. Mais le risque existe toujours.

La mise à jour du document nécessite de chiffrer l'élément ajouté. Enfin, les mises à jour des règles d'autorisation ou, comme elles sont appelées dans ce modèle, les règles d'accès conditionnelles nécessitent, également, le re-chiffrement du document.

3.4 Piste de réflexion

Notre étude a montré que la traduction d'une base de règles en chiffrement n'est pas très compatible avec un degré élevé de mises à jour. Puisque le document chiffré est généré à partir de la base de règles, tout changement conduit à déstabiliser, bien évidement, le document chiffré et coûte très cher. Pour cette raison, il faut imaginer une autre façon pour protéger le document. Dans cette section, nous présentons une réflexion préliminaire, qui pourrait constituer un point de départ pour de futures recherches : "*le chiffrement et le modèle de contrôle d'accès pour XML dans un environnement dynamique*".

L'idée consiste à définir une méthodologie permettant de protéger un document XML indépendamment des règles d'autorisation des utilisateurs. En d'autres termes, l'identification des groupes d'éléments à chiffrer avec une même clé doit se faire indépendamment de la base de règles. Nous proposons donc de découper le document en un ensemble ***d'unités de protection*.** Une unité de protection peut contenir un seul élément, un attribut ou un sous-arbre (voir la Figure 59). Ce découpage repose, par exemple, sur une *pré-connaissance* d'éléments sensibles dans le document formant ainsi une unité de protection. Contrairement aux approches présentées précédemment, où la génération du document chiffré repose sur la base de règle, notre idée inverse la méthodologie. C'est la base de règle d'autorisation qui est générée à partir du document chiffré (ou document découpé). Une façon possible[37] pour construire les unités de protection peut être comme suit :

1. ***Première étape*** : consiste à identifier les nœuds (éléments/attributs) qui vont être protégés d'une façon indépendante. En se basant sur ces nœuds pré-identifiés, appelés nœuds sensibles, nous pouvons découper le document en unités de protection. La découpage suit la sémantique suivante : ***chaque sous arbre*** *porté par un* ***nœud sensible*** *forme une* ***unité de protection***. Plus précisément, l'unité de protection regroupe tous les descendants du nœud sensible qui ne sont pas identifiés comme des nœuds sensibles. Par exemple, nous avons l'ensemble des requêtes XPath {//Patient, //DP, //age, //commentaire, //actsMed, //AnalysesMed}. Ces requêtes identifient un ensemble de nœuds sensibles *NS* ={Patient, DP, age, actsMed, commentaire, analysesMed}. Le découpage se fait à partir de la racine. Pour chaque nœud appartenant à l'ensemble des nœuds identifiés, nous créons une unité de protection regroupant tous les descendants de ce nœud n'appartenant pas à l'ensemble des nœuds identifiés. À titre d'exemple, le nœud *patient* appartient à l'ensemble de nœuds *NS*, donc il forme une unité de protection et comprend uniquement le nœud *patient* car ses deux descendants (*DP* et *ActsMed*) appartiennent à *NS* et forment à leur tour des unités de protections différentes. Le nœud *DP* forme une unité de protection qui comprend ses descendants *nom* et *adresse* (ces deux éléments n'appartiennent pas à l'ensemble *NS*), et ainsi de suite…

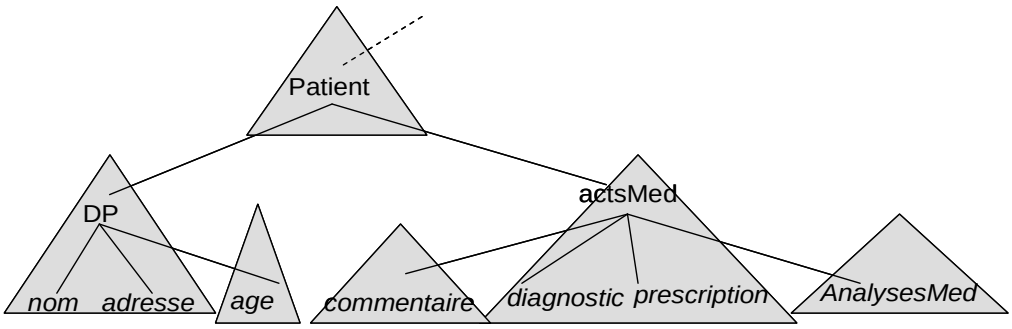


Figure 59 : Découpage du document en unités de protection

[37] Elle reste toujours une intuition.

2. ***Deuxième étape*** : consiste à générer les clés de chiffrement. Chaque unité de protection est chiffrée par une clé différente.

Unité de protection	**Groupes d'éléments**	**Clés de chiffrement**
UP_1	Patient	K_1
UP_2	DP, nom, adresse	K_2
UP_3	*Commentaire*	K_3
UP_4	actsMed, diagnostisc prescription	K_4
UP_5	AnalysesMed	K_5
UP_6	*age*	K_6

Tableau 10 : Table de clés

3. ***Troisième étape*** : consiste à attribuer les unités de protection et les clés correspondant aux utilisateurs. Chaque utilisateur dispose d'un accès à une ou plusieurs unités de protection, ces unités représentent l'ensemble des nœuds auquel il a le droit. L'administrateur peut définir une base de règle d'autorisation où chaque règle est de la forme <Sujet, $\sum$ (signe)UP_i>, le sujet représente un *utilisateur* ou un *groupe* d'utilisateurs et $\sum$ (signe)UP_i représente l'ensemble des unités de protection (accessible/inaccessible) appliqué sur le sujet. Le signe prend soit la valeur "+" qui signifie que l'unité de protection est autorisée. Soit il prend la valeur "-" qui signifie que l'unité de protection est interdite. Cela nous permet de gérer des exceptions à l'intérieur du même groupe d'utilisateurs. Par exemple, tout le groupe Médecin a le droit d'accéder à toutes les unités de protection du document à l'exception du médecin *Dupont* qui n'a pas le droit d'accéder à l'unité de protection contenant l' élément *AnalysesMed*. Pour cela, il suffit de définir deux règles d'autorisation. La première règle donne l'accès aux unités de protection (UP_1, UP_2, UP_3, UP_4, UP_5, UP_6) pour le groupe médecin (R_1). Par contre, la deuxième retire le droit au médecin *Dupont* d'accéder à l'unité de protection $UP_5(R_3)$. La combinaison des règles positives et des règles négatives amène à des situations conflictuelles. La politique de résolution de conflit que nous adoptons est "*le sujet le plus spécifique est le plus prioritaire*" et la "*règle d'autorisation négative est plus prioritaire*" dans le cas où le conflit persiste. La gestion des conflits se fait au moment de la distribution des clés. Un exemple de base de règles est présenté dans la Figure 60.

<- - - --------- - -Base de règles- - - ------- - - ->
R_1= (Médecin, $+UP_1$, $+UP_2$, $+UP_3$, $+UP_4$, $+UP_5$, $+UP_6$)
R_2= (Secrétaire, $+UP_1$, $+UP_2$, $+UP_6$)
R_3= (/Médecin/Dupont, - T_5)
<- - -------- - - - Base de règles - ------- - - - - >

Figure 60 : Exemple de base de règles

Nous remarquons clairement que le découpage en unités de protection crée une stabilité dans le document chiffré, puisque les droits des utilisateurs dépendent de ce découpage. Par contre, la difficulté de cette idée se situe au niveau de l'identification des unités de protection. En d'autres termes, la question qui reste ouverte est "***comment obtenir ce découpage ?***".

Par analogie, nous rapprochons cette idée de celle proposée dans les modèles de contrôle d'accès à base de rôle (RBAC) présentée dans le chapitre II. Les rôles, dans une entreprise, sont définis indépendamment des utilisateurs. Ensuite, chaque nouvel utilisateur est affecté à un rôle prédéfini. De la même façon, les unités de protection sont définies indépendamment des utilisateurs. Ensuite, nous affectons à chaque utilisateur les unités de protection auxquelles il a droit.

Nous concluons cette section par deux tableaux de synthèse sur les mises à jour des sujets et de la base de règles d'autorisation. Cette synthèse regroupe uniquement les modèles à base de chiffrement comparables entre eux. Pour cela, nos tableaux ne tiennent pas compte du modèle de G.Miklau car nous estimons qu'il n'est pas au même niveau de comparaison avec les autres modèles.

Modèles	***Ajout d'un utilisateur à groupe existant***	***Ajout d'un utilisateur sans groupe***	***Suppression d'un utilisateur du groupe existant/sans groupe***
Approche à base de profondeur et ***notre idée à base d'unités de protection***	- 0 clé générée - ***Pas de re-chiffrement***	- 0 clé générée - ***Pas de re-chiffrement***	- Invalidation des clés - 1 à n[38] clés générées - Re-chiffrement des parties concernées - ***Redistribution des clés***
Ray	- 1 clé générée -***Re-chiffrement des parties concernées***	- 1 clé générée - ***re-chiffrement possible***	- Invalidation seulement de la clé de l'utilisateur -0 clé générée -Re-chiffrement des parties concernées - ***Pas de re-distribution de clé***

Tableau 11 : Synthèse et comparaison des mises à jour des sujets

[38] Nombre de clés total utilisées pour chiffrer le document XML.

Modèles	Ajout d'une règle positive	Ajout d'une règle négative	Identification des parties à re-chiffrer[39]
Approche à base de profondeur	- Re-chiffrement - Redistribution des clés	- Re-chiffrement - Redistribution des clés	- Pas immédiate
Ray	- Re-chiffrement - Pas de re-distribution des clés	- Re-chiffrement - Pas de re-distribution des clés	- Pas immédiate
Notre idée à base d'unités de protection	- Pas de re-chiffrement - Pas de re-distribution des clés	- Re-chiffrement - Redistribution des clés	- Immédiate

Tableau 12 : Synthèse et comparaison des mises à jour de la base de règle

4 Conclusion

Dans ce chapitre, nous avons étudié les modèles de contrôle d'accès à base de chiffrement pour XML et adapté l'approche de Ray aux documents XML. Nous avons décrit leurs caractéristiques et montré leurs avantages et inconvénients. Dans le cadre du modèle de Bertino et des extensions proposées par Carminati, nous avons exhibé des contre-exemples montrant les limites de l'approche. Afin de corriger ces problèmes, nous avons proposé un nouvel algorithme de contrôle d'accès pour chiffrer le document XML.

Comme nous avons vu au chapitre II les problèmes d'administration d'une politique de contrôle d'accès sont importants. Il est donc primordial de définir des techniques s'adaptant bien à un contexte plus dynamique. C'est pourquoi nous avons étudié dans un premier temps l'impact des mises à jour dans les différents modèles. Cela nous a permis d'identifier clairement les difficultés des modèles pour supporter les différents types de mises à jour, qui nécessitent dans la majorité des cas le re-chiffrement du document et, par conséquent, une redistribution des clés. Dans un second temps, nous avons proposé une base de réflexion pour résoudre les problèmes rencontrés. Notre idée de découpage du document en unités de protection représente un point de départ pour de futures recherches "*le chiffrement et le modèle de contrôle d'accès pour XML dans un environnement dynamique*".

[39] Cela est également valable dans le cas de la suppression des utilisateurs.

Chapitre VII – Conclusion et Perspectives de recherche

Le travail de cette thèse a porté sur la protection de la confidentialité des données et plus spécifiquement sur les modèles de contrôle d'accès pour XML. Dans ce qui suit nous résumons nos contributions et nos perspectives de recherche.

1 Résumé des contributions

La première étude présentée dans ce document a caractérisé les carences des modèles de contrôle d'accès XML existants. Tout d'abord, nous avons montré qu'il n'existe pas une sémantique unique et non ambiguë des modèles de contrôle d'accès pour XML. En effet, pour une même politique de contrôle d'accès, plusieurs interprétations sont données, certaines ne respectant pas les règles d'autorisation prédéfinies. Cette différence d'interprétation est liée à la difficulté de définir la vue exacte d'un chemin conduisant à un nœud autorisé dans la hiérarchie XML. D'autre part, dans les modèles existants, les hypothèses de travail mentionnées ne permettent pas de prendre en compte les deux principes fondateurs édictés dans les législations pour la protection des données personnelles qui sont les principes de ***need-to-know*** et de ***consentement.*** Cette confrontation avec la législation nous a conduit à identifier deux problèmes importants des modèles existants : **la divulgation de la classification** et **la filiation uniforme.**

C'est au cours de cette étude que nous avons défini notre exemple de référence qui a motivé nos travaux de recherche sur la mise en œuvre d'un nouveau modèle de contrôle d'accès pour XML. Le choix de notre exemple de référence, le dossier médical personnel (DMP), a été fait pour différentes raisons fondamentales : i) les données médicales sont des données hautement confidentielles et leur révélation peut avoir de lourdes conséquences pour la personne concernée ; ii) les données médicales sont de plus en plus exprimées en langage XML et iii) compte tenu de leur sensibilité, le traitement des données médicales est encadré par une législation spécifique. Cet exemple applicatif correspond donc exactement au cadre recherché pour mon étude.

Dans la deuxième étude, nous avons proposé un nouveau modèle de contrôle d'accès pour les documents XML qui contribue à la définition d'une réponse crédible aux problèmes mentionnés ci-dessus. Ce modèle intègre les concepts de need-to-know et de consentement. Ce modèle repose sur des règles permettant de protéger un nœud vis-à-vis de ses ancêtres et

de certains de ses frères. Nous avons caractérisé deux classes d'autorisation sur les associations "**dépersonnalisation des ancêtres**" et "**réduction de chemin**" et pris en compte la dimension des associations de fraternité. Ce modèle a un fort pouvoir d'expression et garde un fort degré de concision. L'ensemble des règles d'autorisations est sûr, c'est-à-dire qu'il existe un algorithme déterministe et compréhensible par un humain qui permet de calculer la vue autorisée à partir du document initial. Pour finir, il est facile de faire évoluer les droits d'accès. L'intérêt de l'approche est qu'elle est compatible avec les modèles existants qui se focalisent uniquement sur la protection des nœuds. En effet, nous avons étendu les modèles existants afin d'intégrer la gestion des droits sur les associations entre les nœuds. Deux mécanismes indispensables le "**clonage**" et le "**shuffling**" ont été introduits dans notre modèle pour traduire exactement ces associations en une vue autorisée.

Cette approche ascendante a été validée par un prototype écrit en java. Il a été construit à partir du seul prototype existant dans le domaine public [40] qui a été étendu par la gestion des règles d'autorisation sur les associations. Nous avons réalisé des tests de performance qui montrent que l'intégration des règles d'autorisation sur les associations est tout à fait comparable en terme de coût à l'intégration de règles d'autorisation sur les nœuds.

Dans la troisième étude présentée dans ce document, nous sommes intéressés aux modèles de contrôle d'accès à base de chiffrement. Ces modèles sont à plusieurs titres intéressants soit dans le cas de serveur de non confiance pour le DMP, soit pour la dissémination sélective des données. Nous avons étudié les différentes approches de chiffrement d'un document qui permettent à partir d'un ensemble de règles d'autorisation sur les nœuds de définir un partage sécurisé d'un document chiffré. L'utilisateur peut déchiffrer le document s'il a en sa possession les clés associées à ses droits d'accès. Ces approches diffèrent dans la façon de générer un document chiffré. L'étude approfondie de ces approches a permis d'exhiber quelques contre-exemples qui montrent que ces méthodes ne fonctionnent pas toujours en pratique. Nous avons proposé une autre approche de chiffrement permettant de résoudre ces problèmes. D'autre part, l'inconvénient de ces approches est qu'elles sont statiques et ne s'adaptent pas très facilement à un contexte dynamique où les droits et les utilisateurs changent très souvent. Nous avons commencé à donner une ébauche de solution à ce problème. Notre intuition consiste à générer un document chiffré indépendamment des utilisateurs. La génération du document chiffré repose sur un découpage prédéfini ; les droits des utilisateurs sont ensuite déterminés en fonction de ce découpage.

2 Perspectives de recherche

- **Validation dans un cadre réel** : Parallèlement aux travaux menés dans cette thèse, il serait intéressant d'expérimenter notre modèle formel dans un cadre réel. L'objectif est de valider si le modèle proposé répond bien à une attente applicative. Le projet de recherche SMIS dans lequel se sont déroulés ces travaux cherche à développer une plate-

forme expérimentale de dossier médical personnel dans un contexte de réseau de soins de portée départementale. Si ce projet de plate-forme aboutit, il pourra servir de cadre à une telle expérimentation.

- **Techniques d'évaluation pour le calcul de la vue autorisée** : La plupart des travaux font aujourd'hui l'hypothèse que le document XML tient en mémoire pour évaluer la politique de contrôle d'accès. C'est l'hypothèse que nous avons retenue pour notre travail préliminaire de validation. Cependant, nous sommes conscients qu'il faut se tourner vers d'autres techniques d'évaluation pour pouvoir gérer des documents de grande taille. L'idée serait d'évaluer en parallèle et à la volée l'ensemble des requêtes XPATH définies dans les règles d'autorisations sur les nœuds et les associations afin de rendre en flux la vue autorisée du document. L'idée d'évaluer en flux un modèle de contrôle d'accès n'est pas nouvelle. Elle a été proposée dans le cadre du projet C-SXA [18] dont l'objectif est d'embarquer la gestion des droits d'accès XML dans une carte à puce. Cependant le modèle proposé dans [19] s'appuie sur un modèle de contrôle d'accès classique à base de nœuds et le type de règles d'autorisations qu'il est possible d'embarquer dans la carte est restreint en raison des contraintes matérielles de cette dernière. Une étude préliminaire a été menée afin de voir comment notre modèle de contrôle d'accès étendu avec des règles d'autorisation sur les associations entre les nœuds peut faire l'objet d'un traitement en flux dans un cadre différent de la carte à puce. Les challenges sont liés au problème de la restructuration parfois complexe liée à notre modèle (e.g. le clonage et le shuffling). Un premier prototype est aujourd'hui opérationnel, mais il reste encore beaucoup d'améliorations et d'extensions à apporter. La difficulté repose sur le fait de devoir gérer en même temps les règles sur les nœuds et les associations, de savoir quand délivrer l'information ou la mémoriser.

- **Contrôle d'accès et techniques de chiffrement en environnement dynamique** : un travail préliminaire a été fait dans cette thèse pour la gestion des droits d'accès en utilisant des techniques de chiffrement. Dans les modèles de contrôle d'accès existants la génération d'un document chiffré se fait à partir d'une base de règle d'autorisation prédéfinie. L'inconvénient de ces modèles est qu'ils sont statiques et ne s'adaptent pas très facilement à un contexte dynamique où les droits et les utilisateurs changent très souvent, d'où la nécessité de définir une façon de générer un document chiffré qui s'adapte au contexte dynamique. Une piste de réflexion a été décrite dans le chapitre IV qui consiste à générer un document chiffré indépendamment de la base de règles d'autorisation. L'intuition est de découper le document en un ensemble d'unités de protection chiffrées de façon indépendante. Ensuite, les droits d'accès des utilisateurs sont déterminés en fonction de ce découpage. Cela crée une certaine stabilité dans le document chiffré. La difficulté de la mise en place d'une telle approche se situe au niveau de l'identification de ces unités de protection. Il est nécessaire d'avoir un outil qui nous permette d'obtenir un découpage cohérent.

- **Modèle de privacité** : Au cours de cette thèse nous nous sommes intéressés à l'étude des

différentes législations relatives à la protection des données à caractère personnel. Un long chemin reste à parcourir avant de satisfaire tous les principes imposés par les législations. Nous avons traité dans le cadre de notre étude deux principes fondateurs des législations. Nous pouvons considérer cela comme un point de départ vers la protection des données à caractère personnel. De nouveaux modèles, appelés dans la littérature "***modèle de privacité"*** [74, 22, 6], cherchent à répondre à une question fondamentale dans tout traitement des données personnelles: « quel est l'*objectif* d'un tel traitement ». Par exemple, dans le domaine médical, les législations déclarent que "*le médecin peut utiliser les données médicales d'un patient pour un* ***objectif*** *de traitement et de diagnostic*". Certains traitements autorisés par les législations sont soumis à des *obligations*. Une des obligations, par exemple pour des raisons statistiques, est d'anonymiser les données avant de les divulguer à un tiers. La traçabilité (historique) des accès représente également un principe fondateur de la protection des données personnelles qui consiste à garder trace de l'heure d'accès, des informations divulguées, du nom et de l'adresse du tiers, de l'objectif… Par exemple, dans le domaine médical, l'accès au dossier médical peut être forcé en cas d'urgence pour sauver la vie de patient. Par contre, les circonstances des accès doivent être sauvegardées pour justifier par la suite les raisons de ces accès. Il reste beaucoup de travail pour intégrer ces différents principes dans un modèle de privacité pour XML. Avec l'informatisation du dossier médical personnel, la définition d'un tel modèle devient pourtant une priorité. Une de nos perspectives vise à poursuivre le travail effectué dans cette thèse dans cette direction.

- **Intégration de connaissances sémantiques** : les modèles actuels de contrôle d'accès définissent des règles d'autorisation en sélectionnant des fragments d'un document par des requêtes XPATH. Ces règles sont très dépendantes de la structuration des documents à protéger et n'intègrent pas de sémantique. Les règles sont donc difficiles à définir et ne s'appliquent pas si le concept interrogé ne correspond pas exactement au concept défini dans la règle. La structuration hiérarchique des informations révèle également certaines informations sensibles comme les relations qui existent entre les objets. Il est donc impératif de fournir des modèles et outils permettant de définir avec plus de précision « qui est autorisé à faire quelles actions sur quelles données et dans quel objectif » tout en offrant une bonne flexibilité en cas de mises à jour des règles ou des sources de données interrogées. L'idée est de définir des politiques de contrôles d'accès à un haut niveau sans se préoccuper du format de stockage ou d'échanges des données. Des travaux récents [7, 60] autour du Web sémantique et les propositions du W3C vont dans ce sens.

Références

1. Abou El Kalam. A., El Baida, R., Balbiani. P., Benferhat. S., Cuppens. F., Deswarte. Y., Miège. A., Saurel. C., and Trouessin, G., "*Or-BAC : un modèle de controle d'accès basé sur les organisations*", *IEEE 4th International Workshop on Policies for Distributed Systems and Networks* (Policy 2003), 2003
2. Abou El Kalam. A., Deswarte. Y. "*Sécurité des systèmes d'information et de communication dans le domaine de la santé*". Sécurité et Architecture Réseaux (SAR'03), France, 2003
3. Abou El Kalam. A., El Baida, R., Balbiani. P., Benferhat. S., Cuppens. F., Deswarte. Y., Miège. A., Saurel. C., and Trouessin, G., "*Modèles et Politiques de Sécurité des Systèmes d'Information et de Communication en Santé et Social*", 1ère Conférence Francophone en Gestion et Ingénierie des Systèmes Hospitaliers (GISEH'04), 2003.
4. Abou El Kalam. A., Deswarte. Y. "*Modèle de sécurité pour le secteur de la santé*", Techniques et Sciences Informatiques (TSI), numéro spécial Sécurité informatique, édition Hermès, vol. 23, n° 3, 2004.
5. Alotaiby. F., Chen. J., "*A Model for Team-Based Access Control (TMAC 2004)*", ITCC'04, 2004.
6. Ashley. P, Powers. C , and Schunter. M, "*privacy promises, access control, and privacy management*". In the third international sympsium on Electronic Commerce, 2002.
7. Benzine M, Beatrice Finance, "*Modèle de contrôle d'accès pour XML : intégration d'informations sémantiques*", stage de DEA, INRIA 2005.
8. Bertino. E., Castano. S., Ferrari. E. and Mesiti. M. "*Specifying and Enforcing Access Control Policies for XML Document Sources*" World Wide Web Tournal 3(3), 2000.
9. Bertino. E., Castano. S., Ferrari. E. and Mesiti. M. : "*Author-X : A java-Based System for XML Data Protection*". In Proc. Of the 14th Annual IFIP WG 11.3 Working Conference on Database Security 2000.
10. Bertino. E., Castano. S., Ferrari. E. and Mesiti. M. *"Controlled Acess and dissemination for XML*" Workshop on Web Information and Data Management 1999.
11. Bertino. E., and Ferrari, E. "*Secure and Selective Dissemination of XML Documents*",

ACM Transactions on Information and System Security, Vol, 5, No.3 August 2002, Pages 290-331.

12. Bertino. E., and Ferrari, E. "*Securing XML Documents with Author-X*", IEEE Internet Computing (2001).

13. Bertino, E., Castano, S. Ferrari, E. " *On specifiying Security Policies for Web Documents with an XML-Based Language*", SACMAT'01, 2001, Virginia, USA.

14. Bell.E, Lapadula, "*secure computer systems : Unified Exposition and Multics interpretation*", the MITRE Corporation, Technical report, 1976.

15. Benzaken V. Burelle M. Castagna G. "*Information flow security for XML transformation*". ASIAN'03, 2003.

16. Biba. K.J, "*Integrity Consideration for Secure Computer Systems*", The MITRE Corporation, Technical Report ESD-TR-76-372 & MTR-3153, 1977.

17. Birget, J., Zou, X., Noubir, G., Ramamurthy, B. "*Hierarchy-Based Access Control in Distributed Environments*", IEEE ICC, 2001.

18. Bouganim, L., Dang-Ngoc, F., Pucheral, P, Projet SMIS, http://www-smis.inria.fr/Eprototype_C-SXA.html.

19. Bouganim, L., Dang-Ngoc, F., Pucheral, P. "*Client-Based Access Control Management for XML Documents*", VLDB, 2004.

20. Bruno, E. , Le Maitre, J. , Murisasco, E. "*Extending XQuery with transformation operators*". ACM Symposium on Document Engineering 2003: 1-8.

21. Bulletin de l'ordre des médecins : http://bulletin.conseil-national.medecin.fr/CNOM/bulletin.nsf/(html)/503BOMN503P01?OpenDocument#enjeux.

22. Byun. J, Bertino. E, and Li. N "*Purpose based access control of complex data for privacy protection*". In SACMAT, 2005.

23. Carminati, B., Ferrari, E., "*AC-XML Documents* : *Improving the Performance of a Web Access Contrôle Module*", SACMAT 2005.

24. Carminati.B and Ferrari.E "*Management of Access Control Policies for XML Document Sources*". Rapport Technique, universite de Milan, Italy, 2002.

25. Computer Security Institute, "*CSI/FBI Computer Crime and Security Survey*", www.gocsi.com/forms/fbi/pdf.html.

26. Convention européenne de sauvegarde des droits de l'Homme et de libertés

fondamentales en 1950. http://sos-net.eu.org/etrangers/ddhc/cedh.htm.

27. Convention sur les droits de l'homme et la biomedicine. http://infodoc.inserm.fr/ethique/Ethique.nsf/Ethique.nsf/0/b7b2998b1bee8591c125665d004be7f9?OpenDocument.

28. Critères d'évaluation de la sécurité des systèmes informatiques (ITSEC) http://www.ssi.gouv.fr/site_documents/ITSEC/ITSEC-fr.pdf.

29. Cuppens, F., Miège, A., "*Or-BAC : Organization Based Access Control*", DRUIDE 2004.

30. Cuppens, F., Miège, A. "*Modeling Contextes in the Or-BAC Model*", in 19th Annual Computer Security Applications Conferen ce, 2003.

31. Cuppens, F., Miège, A. "*Conflict Management in the or-bac model*". Technical report, ENST Bretagne, 2003.

32. Cuppens, F., Nora Cuppens-Boulahia and Miège, A., "*Inheritance hierarchies in the Or-BAC model and application in a network environment*", (FCS'04).2004.

33. Cuppens, F., Pucheral, P., "*Encyclopedie Informatique*", Edition Viubert, à apparaître.

34. Clark, D, D., Wilson, D,R,. "*A comparaison of commercial and military comptuer securité policies*", 1987.

35. Cho, S., Amer-Yahia, S. , Lakshmanan, L., and Srivastava, D. "*Optimizing the secure evaluation of twig queries*", VLDB, 2002.

36. Cynthia CHASSIGNEUX "*La protection des données personnelles en France* " en 2001. http://www.lex-electronica.org/articles/v6-2/chassigneux.htm.

37. Damiani, E., De Capitani di Vimercati, S., Paraboschi, S. , Samarati, P. "*A Fine-Grained Access Control System for XML Documents*", ACM TISSEC 5(2), 2002.

38. E.Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati, " *Securing XML Documents*". In proc. of the 2000 International conference on Extending Data Technology (EDBT 2000), Konstanz, Germany, March 27-31, 2000.

39. Damiani, E,. Vimercati, S,C., Paraboschi, S., Samarati, P,. "*Design and Implementation of an Access Control Processor for XML Documents*".In computer Networks, vol.33, n, 1-6, 2000, PP. 59-75, Elsevier; and proc.of the ninth international world wide web conference(www9) Amsterdam, May 15-19, 2000.

40. Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., Samarati, P. "*Fine-Grained XML Access Control Prototype*", *http://seclab.dti.unimi.it/~xml-sec/*.

41. Dastjerdi, A., Pieprzyk, j., "*Security In Databases : A survey Study*",1996. http://citeseer.ist.psu.edu/baraani-dastjerdi96security.html.

42. David F.C. Brewer and Michael J. Nash,. "*The chinesse wall security policy*", IEEE Symposium on research in security and privacy, 1989.

43. Déclaration d'Helsinki de l'association médicale en 1964 mondiale. http://www.genethique.org/carrefour_infos/textes_officiels/titres_textes/declaration_helsinki_2000.htm.

44. Directive 95/46/CE du Parlement européen et du conseil, de 24 octobre 1995, relative à la protection des personnes physiques à l'égard du traitement des données à caractère personnel et à la libre circulation de ces données. *Journal officiel n° L 281 du 23/11/1995 p. 0031 - 0050* http://www.privacy.fgov.be/nieuw%2029-8-2002/directive_95_46_fr.pdf.

45. E-juriste. Dossier médical informatisé. http://60gp.ovh.net/~ejuriste/article.php3?id_article=620.

46. Fan, W. , Chan, C.Y. , Garofalakis, M. "*Secure XML Querying with Security Views*", SIGMOD, 2004.

47. Fernandez.E., Gudes, E, and Song. H, "*A model for evaluation and administration of security in object-oriented databases*", TKDE 1994.

48. Ferraiolo.D., Sandhu.R., and Gavrila. S,. "*Proposed NIST Standard for role-Based Access Control*", ACM Transactions on Information and System Security, Vol. 4, No.3, 2001.

49. Ferraiolo.D., Cugini.J., and Kuhn. R., "*Role-Based Access Control (RBAC) : Features and Motivations*", 11th Annual Computer Security Applications Proceeding, 1995.

50. Finance. B, Medjdoub. S, Pucheral. P, "*The Case for Access Control on XML Relationalships*", CIKM, 2005.

51. B. Finance, S. Medjdoub, P. Pucheral, "*The Case for Access Control on XML Relationalships*", BDA, 2005.

52. B. Finance, S. Medjdoub, P. Pucheral, "*privacy of Medical records: from law principles to practice*", CBMS, 2005.

53. Fundulaki, I., Marx, M,. "*Specifying Access Control Policies for XML Documents with Xpath*", SACMAT 2004.

54. Gabillon, A., Bruno, E. "*Regulating access to XML documents*". IFIP Conf. on Database and Application Security, 2001.

55. Gabillon, A. "*An Authorization Model for XML DataBases*". ACM Workshop on Secure Web Services. 2004.

56. Gabillon, A. "*A Formal Access Control Model for XML Databases*", 2nd vldb workshop on secure data management, 2005.

57. Gabillon, A. "*An Authorization Model for XML DataBases*". VLDB Workshop on Secure Web Services. 2005.

58. *Gary C. Kessler "Overview of cryptography "*, http://www.garykessler.net/library/crypto.html, 1998.

59. Georgiadis. C., Mavridis. I., Pangalos. G., and Thomas. R., "*Flexible Team-Based Access Control Using Contexts*", SACMAT'01, 2001.

60. Gowadia.V and Farkas. C, "*RDF Metadata for XML Access Control*", ACM Workshop on XML Security , 2003.

61. Griffiths.P, Wade.B "*An authorization mechanism for a relational database system*", ACM Transactions on Database Systems (TODS), 1976.

62. Gudes, E, Song. H, and Fernandez.E. "*Evaluation of Negative, Predicate, and Instance-based Authorization in Object-oriented Databases*", The IFIP WG 11.3 Workshop on Database Security, IV, 1990

63. Harrison, M, A., Ruzzo, Walter.L., Ullman, J., "*Protecting in operating system*", Communication ACM; 1976.

64. Haut-Commissariat aux droits de l'homme "*principes directeurs pour la réglementation des fichiers informatisés contenant des données à caractère personnel*" Adoptée le 14 décembre 1990 par l'Assemblée générale des Nations Unies. http://www.unhchr.ch/french/html/menu3/b/71_fr.htm.

65. He, H,. Wong, R,. "*A Role-Based Access Control Model For XML Repositories*", Proceedings of the First International Conference on Web Information Systems Engineering (WISE'00), 2000.

66. HEALTH INSURANCE PORTABILITY AND ACCOUNTABILITY ACT OF 1996. http://aspe.hhs.gov/admnsimp/pl104191.htm#261.

67. HIPAA Basics: Medical Privacy in the Electronic Age, 2003. http://www.privacyrights.org/fs/fs8a-hipaa.htm.

68. Hitchens, M., Varadharajan, V. *RBAC for XML Document Stores*, ICICS, 2001

69. HL7 : http://www.hl7.org/

70. http://www.medito.com/ci0403i.htm.

71. http://www.rsasecurity.com/rsalabs/node.asp?id=2152.

72. Jajodia, S., Samarati, P., Sapino, M., Subrahmanian, V., "*Flexible support for multiple access control policies*", ACM TODS, 26(2), 2001.

73. Jajodia, S., Samarati, P., Subrahmanian, V.S., and Bertino. E., "*A Unified Framework for Enforcing Multiple Access Control Policies*". In Proceedings of ACMSIGMOD, 1997.

74. Karjoth. G and Schunter. "*A Privacy Policy Model for Entreprises*", In the 15th IEEE Computer Security Foundations Workshop (CSFW'02), 2002.

75. Kudo, M., Hada, S. "*XML Document Security based on Provisional Authorization*", ACM CCS, 2000.

76. Kuper. G., Massacci, F., Rassadko.N., "*Generalized XML Security Views*", SACMAT 2005.

77. Lampson B, W., "*Protection*", In proc. 5th Princeton Conf.on information Sciences and systems, 1971

78. Lim,C., Park, S., Son, S, "*Access Control of XML Documents Considering Update Operations*", ACM Workshop on XML Security, 2003.

79. Loi n°78-17 du 6 Janvier 1978 relative à l'informatique, aux fichiers et aux libertés, journal officiel du 7 janvier 1978 et rectifiant au J.O. du 25 janvier 1978.

80. Loi du 6 janvier 1978 relative à l'informatique, aux fichiers et libertés (modifiée par la loi relative à la protection des personnes physiques à l'égard des traitements de données à caractère personnel du 6 août 2004). http://www.cnil.fr/fileadmin/documents/approfondir/textes/CNIL-78-17_definitive.pdf.

81. Loi n°2002-303 du 4 mars 2002 relative aux droits des malades et à la qualité du système de santé. http://www.legifrance.gouv.fr/WAspad/UnTexteDeJorf?numjo=MESX0100092L.

82. LOI n° 2004-810 du 13 août 2004 relative à l'assurance maladie (1): http://www.legifrance.gouv.fr/WAspad/UnTexteDeJorf?numjo=SANX0400122L.

83. Medjdoub. S, Finance. B, "*Publication sécurisée des documents XML*", http://www.prism.uvsq.fr/~sam/.

84. Menezes, P. Oorschot, S. Vanstone, 1996. "*Handbook of Applied Cryptography*". CRC Press, 1996.

85. Murata M., Tozawa A., Kudo M., "*XML Access Control Using Static Analysis*", ACM CCS, 2003.

86. Miklau, G., Suciu, D. "*Containment and equivalence for an XPath fragement*", ACM PODS, 2002.

87. Miklau.G, and Suciu.D " *Cryptographically Enforced Conditional Access for XML*", Fifth International Workshop on the Web and Databases (WebDB 2002).

88. National Computer Security Center, "*A Guide to Understanding discretionary Access Control in Trusted systems*", 1987.

89. Nyanchama. M., Osborn, S., "*Modeling Mandatory Access Control in Role-Based Security Systems*". Proceeding of the ninth annual IFIP TC11 WG11.3 working conference on Database security, 1996.

90. OASIS standard, "*eXtensible Access Control Markup Language*", http://www.oasis-open.org/committees/xacml, 2003.

91. OCDE : http://www.oecd.org/about/0,2337,en_2649_201185_1_1_1_1_1,00.html.

92. Parmar, V., Shi, H., "*XML Access Control for Semantically Related XML Documents*", HICSS, 2003.

93. Overview of the Privacy Act of 1974, May 2004 http://www.usdoj.gov/04foia/04_7_1.html.

94. Privacy Act of 1974. http://www.house.gov/matheson/the_privacy_act_of_1974.html.

95. Privacy Act of 2005, 109th Congress, session 1st, 24 janvier 2005 : http://www.droit-ntic.com/news/afficher.php?id=291.

96. Qi, N., Kudo, M,. "*Access-Condition-Table-driven Access control for XML Databases*", 2004.

97. Qi, N., Kudo, M,. "*Tree-based Access Control Mechnism for XML Databases*", Digital Engineering Workshop, 2005.

98. Rabitti, F., Bertino, E., Kim, W., and Woelk, D. "*A model of the authorization for next-generation databases systems*", ACM Trans Database Syst 16(1), 1991.

99. Ray.I and Ray.In "*Using Compatible Keys for Secure Multicasting in E-Commerce*". In 16th International Parallel and Distributed Processing Symposium (IPDPS) 2002.

100. Ray.I, Ray.In and Narasimhamurthi "*A cryptographic Solution to Implement Access Control in a hierarchy and More*". SACMAT 2002.

101. Recommandations pour la pratique clinque "*accès aux informations concernant la*

santé d'une personne – modalités pratiques et accompagnement", 2004. http://www.anaes.fr/anaes/Publications.nsf/nPDFFile/RE_LILF-5XDF7B/$File/Acces_Infos_sante%20-_Recos.pdf?OpenElement.

102. Recommandation n° R(97)5 du comité des ministres aux états membres relative à la protection des données médicales (adoptées par le Comité des ministres le 13 février 1997 lors de la 584ème réunion des délégués des ministres).

103. Recommandation R(81)1 à la réglementation applicable aux banques de données médicales automatisées : http://www.coe.int/T/F/Coh%E9sion_sociale/Sant%E9/Recommandations/Rec(1981)01.asp.

104. Samarati. P,. Jajodia. S,. "*data security*". Research and practice. I S 20(7): 537-556(95).

105. Samarati, P., Vimercati, S,C., "*Access Control : Policies, Models, and Mechanisms*", Foundations of Security Analysis and Design FOSAD, 2000.

106. Sandhu. R., Coyen. E., Feinstein, H., and Youman., C., "*Role-Based Access Control Models*", IEEE Computer, 1996.

107. Sandhu. R., and Munawer, "*How to do Discretionary Access Control Using Roles*", In proc of 3rd ACM Workshop on role-based Access Control, 1998.

108. Stoica.A and Farkas.C. "*Secure XML views*". In Proc. 16th IFIP WG11.3 Working Conference on Database and Application Security, 2002.

109. Tabatoni, P "*Stratégies de la privacy aux États-Unis, La dynamique des systèmes de protection"* http://www.asmp.fr/travaux/gpw/internetvieprivee/rapport3/chapitr14.pdf.

110. Thomas. R., "*Team-based Access Control (TMAC) : A primitives for Applying Role-based Access Controls in collaborative Environnements*", RBAC'97, 1997.

111. ToXgene – The ToX XML data generator, http://www.cs.toronto.edu/tox/toxgene/.

112. OASIS standard, eXtensible Access Control Markup Language, http://www.oasis-open.org/committees/xacml, 2003.

113. Osborn. S., "*Mandatory Access Control and Role-based Access Control Revisited*", RBAC'97, 1997.

114. XrML eXtendible rights Markup Language, http://www.xrml.org/.

115. XML : http://www.w3.org/XML/.

116. XPath : http://www.w3.org/TR/xpath.

117. Y. Wang, K.L. Tan, "*A Scalable XML Access Control System*", WWW Conference (poster), 2001

118. Wang, J., Osborn, S.L. "*A Role-Based Approach to Access Control for XML Databases*", ACM SACMAT, 2004.

119. Zhang, X., Park, J., Sandhu, R,. "*Schema Based XML Security : RBAC Approach* ", Seventeenth IFIP 11.3 Working Conference on Data and Application Security, 2003.

120. Wedde. H., Lischka. M., "*Modular Authorization*", SACMAT'01, 2001.