République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abderahmane Mira de Béjaïa

Faculté des Sciences Exactes

Département d'Informatique

École Doctorale Réseaux et Systèmes Distribués

Mémoire de Magistère

En Informatique

Option

Réseaux et Systèmes Distribués

Thème

Optimisation de gestion de politiques de sécurité dans des environnements multi-domaine

Présenté par

Hanane BOUCHENAK

Devant le jury composé de :

Président	KERKAR Moussa	Professeur	Université de Bejaia, Algérie.
Rapporteur	AMIRAT Yacine	Professeur	Université de Paris XII, France.
Examinateur	MELIT Ali	Maître de conférence	Université de Jijel, Algérie.
Examinateur	BOUALLOUCHE Louiza	Maître de conférence	Université de Bejaia, Algérie.
Invité	TARI Abdelkamel	Docteur	Université de Bejaia, Algérie. France

Promotion 2006-2007

Table des matières

Liste des figures

Liste des tableaux

Résumé

Les environnements multi-domaine où des organisations multiples distribuées collaborent entre elles, deviennent comme un témoin pour les applications naissantes d'entreprises basées sur Internet. La composition d'une politique de sécurité globale et cohérente qui régit les accès aux informations et ressources dans tels environnements est un problème challenge.
Dans ce projet, nous nous sommes intéressés à la composition et l'intégration des politiques de contrôle d'accès basées sur les rôles (RBAC). Établir une politique RBAC multi-domaines à partir de différentes politiques de contrôle d'accès peut générer différents types de conflits. Ces conflits, s'ils restent non détectés et non résolus, amènent les systèmes de collaboration à de nombreuses vulnérabilités et risques concernant la sécurité et à la confidentialité de leurs données et ressources. Nous nous sommes basés sur la méthode de composition et d'intégration des politiques RBAC proposée par BASIT. Cependant, cette méthode peut ne pas détecter et résoudre tous les conflits apparaissant dans une politique RBAC multi-domaines, en plus la représentation de la politique d'interopération n'est pas nécessairement minimale.
Dans ce travail, nous avons amélioré la méthode de détection et de résolution des conflits de BASIT afin de générer une politique RBAC multi-domaines cohérente et sûre. Nous avons étendu également le processus de composition de BASIT avec une étape afin d'obtenir une représentation minimale de l'interopérabilité sûre des politiques RBAC des domaines collaborants.

Mots clés : politique de sécurité multi-domaines, contrôle d'accès basé sur rôle, composition de politiques, optimisation.

Abstract

Multidomain application environments where distributed multiple organizations interoperate with each other are becoming a reality as witnessed by emerging Internet-based enterprise applications. Composition of a global coherent security policy that governs information and resource accesses in such environments is a challenging problem.
In this project, we have interested the composition and integration of the Role-Based Access Control (RBAC) policies. Establish a multi-domain RBAC policy from different access control policies can generate different types of conflicts. These conflicts, if remain undetected and unresolved, expose the collaborating systems to numerous vulnerabilities and risks pertaining to the security and privacy of their data and resources. We have based on the method of composition and integration of the policies RBAC proposed by BASIT. However, this method may not detect and resolve all conflicts occurring in multi domain RBAC policy ; in addition, the representation of policy of interoperation is not necessarily minimal.
In this work, we improved the method of detection and Resolution of BASIT in order to generate a multi domain consistent and safe RBAC policy. We have also extended BASIT's composition process with a step to obtain a minimal representation of interoperability Secure of RBAC policies of collaborating domains.

Keywords : multidomain security policy, role-based access control (RBAC), policies composition, optimization.

Dédicaces

A ma mère et mon père,
A mes soeurs et mes frères,
A notre petite princesse Houriya-Malak,
A toute la famille,
A tous mes amies et mes collègues.

Remerciements

JE remercie avant tout Dieu Tout puissant qui m'a donné la santé, le courage et la volonté pour réaliser ce travail.

L'élaboration de ce mémoire n'aurait pas été possible sans l'aide précieuse de nombreuses personnes. Je tiens donc à remercier tout particulièrement ceux qui ont su me conseiller dans mon travail.

Tout d'abord, je porte toute ma gratitude et ma reconnaissance à mon encadreur Mr AMIRAT Yacine, Professeur à l'université de Paris 12 pour l'aide et le temps qu'il a bien voulu me consacrer et sans qui, ce mémoire n'aurait jamais vu le jour.

Je veux exprimer ma reconnaissance à mon co-encadreur Mr TFAILI Walid, docteur à l'université de Paris 12 pour ses remarques et corrections qui ont permis l'élaboration de ce mémoire.

Je salue également Mr TARI Karim pour m'avoir encadré au début de ce projet, et pour ses précieux conseils et orientations.

Je remercie Mr KERKAR Moussa, professeur à l'université de Bejaia pour m'avoir fait l'honneur d'accepter de présider le jury de ma thèse.

Des remerciements tout particuliers à Mme BOUALLOUCHE Louiza, maître de conférence à l'université de Bejaia et Mr MELIT Ali, maître de conférence à l'université de Jijel pour avoir accepté d'être membres du jury de ma thèse et pour le temps qu'ils ont consacré pour examiner mon travail.

J'exprime ma gratitude à Mr TARI Abdelkamel, le chef de département d'informatique et responsable de l'école doctorale ReSyD, sans qui, il m'aurait sans doute été très difficile de soutenir ma thèse dans d'aussi bonnes conditions.

J'adresse mes plus sincères remerciements à mon amie DAOUDI Asma qui m'a toujours soutenue et encouragée au cours de la réalisation de ce mémoire.

Enfin, un grand merci à ma mère, mon père, mes frères : Yacine, Walid, Mohamed Amine, mes sœurs : Manel, Sabrina, ma petite nièce : Malak Houriya et toute ma famille pour sa présence, sa préoccupation et le souci qu'elle s'est fait pour moi, ses encouragements et son suivi, avec patience, du déroulement de mon projet.

Introduction générale

LES progrès récents dans le calcul à haute performance et les technologies des réseaux ont donné lieu à une croissance énorme des applications réparties à grande échelle, dans la médecine, l'enseignement, l'e-commerce, les bibliothèques numériques, et beaucoup d'autres. En même temps, il y a un besoin croissant du partage des informations et des ressources dans des environnements collaboratifs qui recouvrent des différentes entreprises. Diverses entreprises, gouvernements, et d'autres organismes ont compris que le partage des informations et les échanges de ressources est de plus en plus important à leur succès [3]. L'environnement multi-domaines est une collaboration des organisations, qui peuvent appartenir aux différents domaines (Militaire, Commercial, Recherche Scientifique, Médical, Industrie, etc), pour partager les données et les ressources entre elles. L'interopérabilité dans cet environnement est une tâche challenge, à cause de l'hétérogénéité dans différents niveaux ; la sémantique des données, la représentation des données, etc.

Avec l'augmentation d'accessibilité des informations et des données il y a un souci en croissance pour la sécurité et la vie privée de données. De nombreuses études ont montré que l'accès non autorisé, en particulier par des initiés, constitue un problème majeur de sécurité pour des environnements d'application d'entreprise [3]. Ce problème peut être fortement magnifié dans un environnement multi-domaines, où les organisations utilisent des politiques de contrôle d'accès différentes [10, 13].

Composer une politique globale de contrôle d'accès compatible avec les politiques des domaines participants est nécessaire pour établir un mécanisme efficace de contrôle d'accès dans cet environnement. Il est essentiel d'avoir une représentation uniforme des différentes politiques de contrôle d'accès dans un tel environnement, le modèle RBAC peut être efficacement utilisé pour la représentation uniforme des politiques de sécurité de domaines collaborants.

La politique multi-domaines composée peut ne pas être cohérente, et différents types de conflits peuvent apparaître. Pour détecter et résoudre ces conflits d'une manière optimale, il est possible de formuler le problème de composition sous la forme d'un problème d'optimisation avec l'objectif de maximiser les accès inter-domaines permis selon des critères d'optimalité prédéfinis.

Plusieurs métriques d'optimalité telles que le partage maximum, la représentation minimum et l'accès pondéré peuvent être utilisées pour résoudre ces conflits et les vulnérabilités des politiques de sécurité de manière optimale. C'est dans ce contexte que se situe le travail que nous allons présenter dans ce mémoire. Le problème auquel nous avons traité est l'optimisation des politiques de contrôle d'accès des environnements multi-domaines basées sur le modèle RBAC.

Ce mémoire comprend cinq chapitres :

Le premier chapitre présente des généralités sur le contrôle d'accès, les politiques de contrôle d'accès ainsi que les différents modèles de contrôle d'accès.

Le deuxième chapitre décrit les environnements multi-domaines, et la problématique de contrôle d'accès dans ces environnements.

Le troisième chapitre est dédié à citer les méthodes de composition et d'intégration de politiques de contrôles d'accès. Comme dans ce projet on s'interesse au modèle RBAC, nous avons détaillé les méthodes de composition des politiques RBAC.

Le quatrième chapitre concerne l'objectif de ce projet qui est l'optimisation de composition de politiques de contrôle d'accès. Il présente les différents types de conflits qui peuvent apparaître dans une politique multi-domaine RBAC et les méthodes de résolution de ces conflits. Puis, nous nous sommes intéressé à la formulation du problème de composition sous la forme d'un problème d'optimisation avec une représentation des critères d'optimisation et les méthodes d'optimisation existantes.

Le cinquième chapitre présente notre contribution dans la problématique d'optimisation de gestion de politiques de contrôle d'accès RBAC. Nous avons proposé une amélioration d'une méthode d'optimisation, avec une extension du processus de composition des politiques RBAC.

Enfin, notre mémoire s'achève par une conclusion générale résumant les grands points qui ont été abordés ainsi que des perspectives que nous souhaitons accomplir prochainement.

Chapitre 1

Les politiques de contrôle d'accès

1.1 Introduction

Les progrès dans les systèmes distribués et les technologies de gestion des réseaux ont rendu l'interopérabilité des systèmes des différents domaines non seulement faisable mais également de plus en plus populaire. En même temps avec ce grand succès, assurer la sécurité dans cet environnement est un problème très compliqué.

Pour garantir la sécurité, plusieurs techniques existent dont le chiffrement, l'authentification et le contrôle d'accès. Dans notre travail, on s'intéresse à la gestion de contrôle d'accès dans les environnements multi-domaine. Mais avant d'élaborer cet environnement et sa complexité, d'abord nous allons voir, dans ce chapitre, quelques notions de base sur le contrôle d'accès, la politique de contrôle d'accès, les principaux modèles et politiques de contrôle d'accès, et puis nous allons parler brièvement d'un langage de spécification des politiques de contrôle d'accès.

1.2 Les objectifs de sécurité

Selon [15, 11] la sécurité informatique recouvre trois types d'objectifs : la confidentialité, l'intégrité et la disponibilité.

- **La confidentialité :**
  C'est la prévention de la divulgation non autorisée de l'information. Ceci signifie que le système informatique doit :
  - Empêcher les utilisateurs de lire une information confidentielle (sauf s'ils y sont autorisés).
  - Empêcher les utilisateurs autorisés à lire une information et de la divulguer à d'autres

utilisateurs (sauf autorisation).

- **L'intégrité :**
  C'est la prévention de la modification non autorisée de l'information. Cela signifie que le système informatique doit :
  - Empêcher une modification indue de l'information, c'est-à-dire une modification par des utilisateurs non autorisés ou une modification incorrecte par des utilisateurs autorisés.
  - Faire en sorte qu'aucun utilisateur ne puisse empêcher la modification légitime de l'information. Par exemple, empêcher la mise à jour périodique d'un compteur de temps constituerait une atteinte à l'intégrité.
- **La disponibilité :**
  C'est la prévention d'un déni non autorisé d'accès à une information ou à une ressource. Cela signifie que le système informatique doit :
  - Fournir l'accès à l'information pour que les utilisateurs autorisés puissent la lire ou la modifier.
  - Faire en sorte qu'aucun utilisateur ne puisse empêcher les utilisateurs autorisés d'accéder à l'information.

1.3 Le contrôle d'accès

Le contrôle d'accès vise à intercepter toutes les tentatives d'accès aux informations critiques. Il se définit à différents niveaux de conception par les composants suivants :

- **Les politiques de sécurité** : définissent les règles de haut niveau qui régissent les accès.

- **Les modèles de sécurité** : dressent une représentation formelle des politiques de sécurité.

- **Les mécanismes de sécurité** : définissent les fonctions de bas niveau (logiciels et matérielles) permettant d'implanter le contrôle imposé par la politique.

1.4 Les concepts de base de contrôle d'accès

La mise en œuvre de contrôle d'accès fait intervenir plusieurs notions essentielles. Il s'agit des notions d'objet, du sujet, de permission, d'interdiction et d'obligation.

- **Les sujets :**
  Le sujet est l'entité active. Il désigne un utilisateur, le système lui même, un processus s'exécutant pour le compte d'un utilisateur ou un processus système.
- **Les objets :**
  L'objet est l'entité passive. Il désigne une information ou une ressource à laquelle un sujet peut accéder pour réaliser une action, (exemple : un fichier, une tuple, une table, périphériques matériels,etc).
- **Les actions(opérations) :**
  Les actions sont des moyens permettant aux sujets de manipuler les objets du système, (par exemple : lecture, écriture, exécution, modification, suppression, recherche,etc).
- **Les permissions(respectivement Les interdictions) :**
  Les permissions(respectivement Les interdictions) décrivent quels actions les sujets ont le droit(resp. sont interdits) de faire sur les objets.
- **Les obligations :** imposer une ou plusieurs actions ou bien renoncer à une obligation.

Contrôler les accès, c'est déterminer si un sujet peut effectuer une opération demandée sur une ressource.

1.5 La politique de contrôle d'accès

Une politique de contrôle des accès (access control policy) spécifie sous forme des règles, les droits des sujets sur les ressources d'un système dans le but de renforcer la politique de sécurité au sens large (intégrité, disponibilité, protection physique des biens et des personnes,etc) de l'organisation [29].

1.6 Les principes d'une politique de contrôle d'accès

Dans les politiques de contrôle d'accès, il y a deux principes qui doivent être vérifiés :

- **Le principe du moindre privilège**
  Ce principe consiste à affecter des préférences aux utilisateurs de façon à ce qu'ils n'aient

pas plus de permissions que nécessaire pour effectuer leurs tâches.

- **Le principe de séparation des tâches**
  Requiert afin de garantir l'intégrité des données sensibles, la conjonction des actions de plusieurs sujets avant de valider la permission d'accès.

1.7 Les modèles de contrôle d'accès

Généralement, il y a trois types principales de modèles de contrôle d'accès :

1.7.1 Le contrôle d'accès discrétionnaire (DAC)

Le contrôle d'accès discrétionnaire ou DAC(Discretionary Access Control)s'appuient sur les notions de propriété (tout sujet est propriétaire d'un ensemble d'objets) et de droit d'accès. C'était la première technique émergente dans le domaine de contrôle d'accès, elle était la plus naturelle ; le propriétaire de l'objet généralement son créateur, a le droit d'en faire ce qu'il veut et de spécifier aussi les permissions aux autres sujets. La gestion des droits d'accès est généralement donnée par la règle suivante : « Un sujet s peut donner un droit d'accès sur un objet o à un autre sujet s_0 si et seulement si s est le propriétaire de l'objet o ». Ainsi, elle permet à un sujet de céder la propriété d'un objet à un autre sujet.

Le contrôle d'accès est dit discrétionnaire lorsque la technique de restriction d'accès aux objets est basée sur l'identité des sujets et/ou des groupes auxquelles ils appartiennent. Le contrôle est discrétionnaire dans le sens où un sujet possédant un certain droit d'accès est capable de conférer ce droit à tout autre utilisateur [7].

La politique de contrôle d'accès discrétionnaire est individuelle, c'est-à-dire que chaque utilisateur construit sa propre politique de sécurité. DAC est une forme de contrôle d'accès largement utilisée dans les systèmes informatiques et réseaux. Par exemple, dans le système d'exploitation Windows, le propriétaire d'un fichier ou d'un dossier peut accorder ou refuser l'accès d'un utilisateur ou un groupe d'utilisateurs. DAC est basé sur la matrice de contrôle d'accès (Access Control Matrix : ACM). Le modèle de matrice a été développé par Lampson et étendu par Graham et Denning. Par la suite, Harrisson, Ruzzo et Ullman ont développé une version plus générale du modèle [26]. Cette matrice est définie comme suit :

- Les lignes correspondent aux identités des sujets.
- Les colonnes correspondent aux objets.

– Chaque entrée de cette matrice (intersection d'une ligne s et d'une colonne o) contient les privilèges du sujet s pour l'objet o.

	Fichier1	**Fichier2**	**Fichier3**	**Programme**
Ann	propriétaire, écrire, lire			exécuter
Bob		écrire, lire		
Carl			lire	exécuter, lire

TAB. 1.1 – Exemple de la matrice de contrôle d'accès.

1.7.1.1 L'implémentation de la matrice d'accès

Dans de grands systèmes, cette matrice devient énorme en taille, et plusieurs de ses entrées sont vides. Par conséquent, cette matrice est rarement implémentée comme une matrice mais plusieurs autres formes sont utilisées telles que ACL (Access Control List), CL (Capability List) et AR (Authorization Relations) [26].

a. Access Control List (ACL)

La liste de contrôle d'accès est l'une des formes utilisées pour implémenter la matrice d'accès. Elle est généralement utilisée dans les systèmes d'exploitation et les services de sécurité des réseaux [26]. Chaque objet est associé à une liste ACL, qui indique pour chaque sujet du système les accès autorisés à cet objet. La figure 1.1 montre un exemple d'ACL pour deux objets (Fichier1 et Fichier2) et trois utilisateurs X, Y et Z. Les modes d'accès R, W et Own correspondent respectivement à lire, écrire et propriétaire. Cette approche correspond à la sauvegarde de la matrice d'accès en colonnes. Avec ACL, il est facile de déterminer les privilèges d'accès associés à chaque sujet pour un objet donné. Cependant, pour déterminer tous les accès permis à un sujet dans le système, toutes les ACLs doivent être parcourues.

b. Capability List (CL)

La liste des capacités ou d'aptitudes (CL) est une approche duale à ACL. Chaque sujet est associé à une liste CL, qui indique les accès autorisés pour chaque objet du système. Cette approche correspond à la sauvegarde de la matrice d'accès par lignes. La figure 1.2 montre les CLs correspondantes aux ACLs de la figure 1.1.

Avec CL, il est facile de déterminer tous les privilèges d'accès associés à un sujet. Cependant, la détermination de tous les sujets dans le système qui peuvent accéder à un objet donné exige de parcourir toutes les CLs.

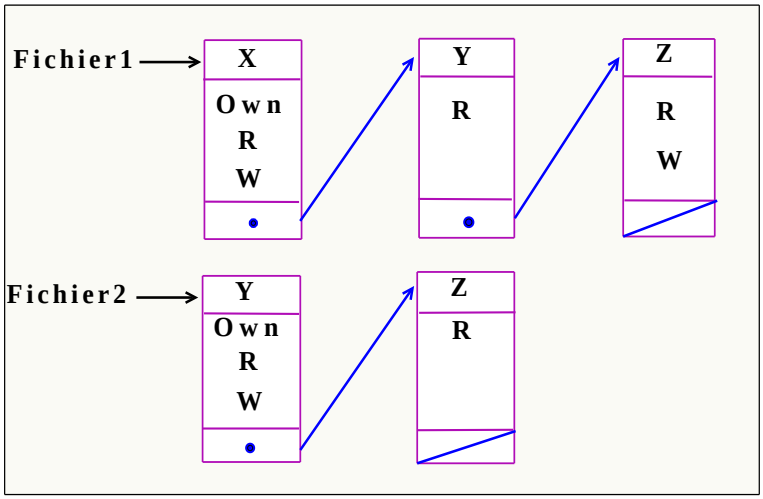


FIG. 1.1 – Exemple de liste de contrôle d'accès (ACL).

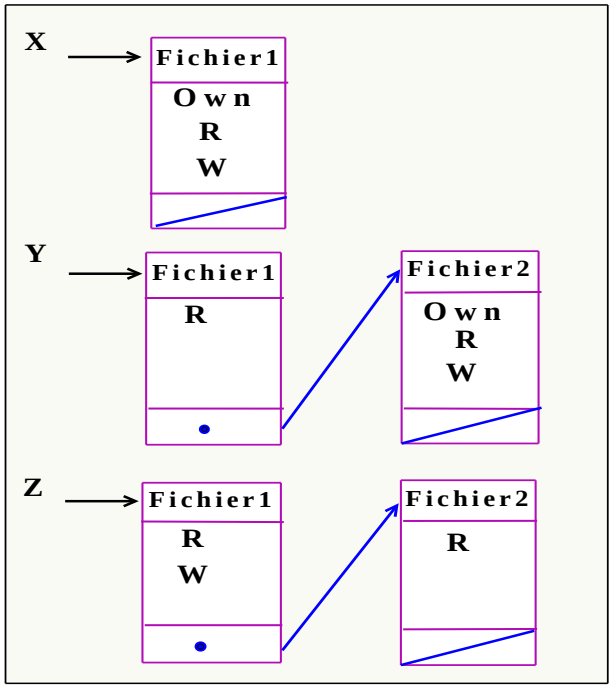


FIG. 1.2 – Exemple de liste de capacités (CL).

c. Authorization Relations (AR)

Il est mentionné précédemment que les listes ACL et CL ont des avantages et des inconvénients duaux par rapport à l'accès des listes. La représentation de la matrice d'accès par une table de relations d'autorisation ne favorise aucun type d'accès (par rapport aux sujets ou aux objets). Chaque ligne de cette table spécifie un seul mode d'accès d'un sujet à un objet. Par conséquent, AR a les effets de ACL et CL. Cette représentation est typiquement utilisée dans les systèmes de gestions des bases de données relationnelles. La table 1.2 montre la table

de relations d'autorisation correspondante à l'ACL et la CL de l'exemple précédent.

Sujet	**Mode d'accès**	**Objet**
X	Own	Fichier1
X	R	Fichier1
X	W	Fichier1
Y	R	Fichier1
Y	Own	Fichier2
Y	R	Fichier2
Y	W	Fichier2
Z	R	Fichier1
Z	W	Fichier1
Z	R	Fichier2

TAB. 1.2 – Exemple de table de relations d'autorisation (AR).

1.7.1.2 Les limitations

Ce modèle de politique de contrôle d'accès peut parfois amener le système dans un état d'insécurité (c'est-à-dire contraire aux objectifs de sécurité qui ont été choisis). Prenons un simple exemple, reposant sur les mécanismes de protection d'Unix. Dans un tel système, les droits d'accès à un fichier sont définis et modifiables librement par l'utilisateur propriétaire du fichier.

Supposons que le schéma d'autorisation se définisse de la manière suivante :
- Un utilisateur peut créer des fichiers dont il devient alors propriétaire ;
- Le propriétaire d'un fichier peut décider quels utilisateurs sont autorisés à lire ses fichiers.

D'autre part, supposons que la politique exige de respecter l'objectif suivant :
- Les utilisateurs qui n'ont pas le droit de lire un fichier ne doivent pas pouvoir en connaître le contenu.

Une telle politique n'est pas réalisable par des mécanismes d'autorisation discrétionnaire, parce que :
- Un utilisateur u_1 propriétaire d'un fichier f_1, peut donner à l'utilisateur u_2 le droit de lecture sur f_1.
- L'utilisateur u_2 peut créer un fichier f_2 sur lequel il peut donner le droit de lecture à u_3.
- L'utilisateur u_2 peut alors recopier f_1 dans f_2 pour transmettre les informations de f_1 à u_3 à l'insu du propriétaire u_1.

Une politique discrétionnaire n'est donc applicable que dans la mesure où il est possible de faire totalement confiance aux utilisateurs et aux sujets qui s'exécutent pour leur compte. Une telle politique est par là même vulnérable aux abus de pouvoir provoqués par malveillance. Ainsi, s'il est possible à un utilisateur d'accéder à certains objets ou d'en modifier les droits d'accès, il est possible qu'un cheval de Troie s'exécute pour le compte de cet utilisateur (à son insu) en fasse de même. De plus, si un utilisateur a le droit de lire une information il a le droit de la transmettre à n'importe qui.

1.7.2 Le contrôle d'accès mandataire (MAC)

Le contrôle d'accès obligatoire ou MAC (Mandatory Access Control) porte non plus simplement sur l'accès aux objets, mais également sur le flux de l'information contenue dans les objets. Les politiques obligatoires décrètent des règles incontournables destinées à forcer le respect des exigences de sécurité. Ainsi, les politiques de contrôle d'accès obligatoire affectent aux objets et aux sujets des niveaux non-modifiables par les usagers et donc qui limitent leur pouvoir de gérer les accès aux données qu'ils possèdent [15].

Le contrôle d'accès est dit obligatoire lorsque l'accès aux objets est basé sur le niveau de sensibilité de l'information contenue dans les objets. L'autorisation d'accéder à un objet est accordée à un sujet si le niveau d'autorisation de ce sujet est en accord avec le niveau de sensibilité de l'information [7]. La règle d'une politique de contrôle d'accès obligatoire peut ainsi s'exprimer de la manière suivante : Un sujet est autorisé à accéder à un objet si et seulement si le sujet est autorisé à accéder aux informations contenues dans l'objet [7].

MAC est utilisé quand la politique de sécurité dicte qu'un propriétaire d'une ressource ne doit pas prendre les décisions d'autorisation, cette situation est souvent retrouvée dans les systèmes gouvernementaux et les systèmes de défense.

1.7.2.1 Le modèle de Bell-LaPadula

Bell et LaPadula ont développé en 1975 pour le DoD (Department of Defense) des Etats-Unis [15] un modèle de contrôle d'accès mandataire appelé contrôle d'accès basé sur les treillis ou LBAC(Lattice-Based Access Control), il s'agit d'un modèle de sécurité multiniveaux, qui a été mis au point au moment où les efforts se concentraient pour concevoir un système d'exploitation multi-utilisateurs. Le modèle associé à la politique de Bell-LaPadula est fondé sur la notion de treillis. Il s'appuie sur l'association des niveaux de sécurité aux sujets et aux objets. Les niveaux de sécurité associés aux sujets sont appelés niveaux d'habilitation (noté

h) et les niveaux de sécurité associés aux objets sont appelé niveaux de classification (noté c) [26].

Chaque niveau $n = (cl,\ C)$ est caractérisé par ces deux attributs :

- C : définit la catégorie dans laquelle le sujet travaille ou spécifie la catégorie à laquelle l'information contenue dans l'objet est référée, par exemple NATO, nucléaire, et armée pour les systèmes militaires ; Financier, administration, et recherche pour les systèmes commerciaux.
- cl : une classification prise dans un ensemble totalement ordonné, par exemple : non classifié, confidentiel, secret, très secret tel que : très secret $>$ secret $>$ confidentiel $>$ non classifié.

Pour un objet o, la classification $c(o)$ est un moyen de représenter le danger que peut constituer la divulgation de l'information contenue dans cet objet. Pour un sujet s, c'est une habilitation $h(s)$ qui désigne la confiance qui lui est accordée. Les niveaux constituent un treillis partiellement ordonné par une relation de dominance notée "$\leq$ " et définie comme suit :
Soient deux niveaux $n = (cl,\ C)$ et $n' = (cl',\ C')$; n' domine n $(n \leq n')$ si et seulement si $cl \leq cl'$ et $C \subset C'$.

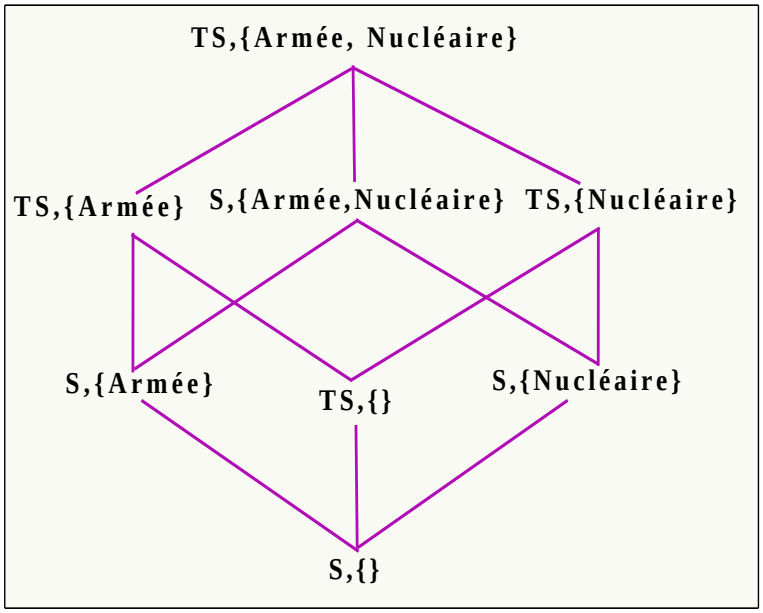


FIG. 1.3 – Un treillis partiellement ordonné.

Dans l'exemple de la figure 1.3 , il y a deux catégories {Armée, Nucléaire} et deux niveaux de classifications {Très secret (TS), Secret (S)}. Les objectifs de sécurité de cette politique sont les suivants :

- Interdire toute fuite d'information d'un objet possédant une certaine classification vers un objet possédant un niveau de classification inférieur ;

– Interdire à tout sujet possédant une certaine habilitation d'obtenir des informations d'un objet d'un niveau de classification supérieur à cette habilitation.

Parmi les différentes opérations qu'un sujet peut effectuer sur un objet, les opérations de lecture et d'écriture. Les règles suivantes sont incontournables même par les propriétaires des informations :

- **La sécurité simple :**
  le sujet s peut lire l'objet o seulement si $c(o) \leq h(s)$.
- **La propriété*libérale(large) :**
  le sujet s peut écrire l'objet o seulement si $h(s) \leq c(o)$. Cette propriété permet au sujet de bas niveau d'écrire un objet de haut niveau. Ainsi, les données de haut niveau peuvent être malicieusement détruites par un sujet de bas niveau.

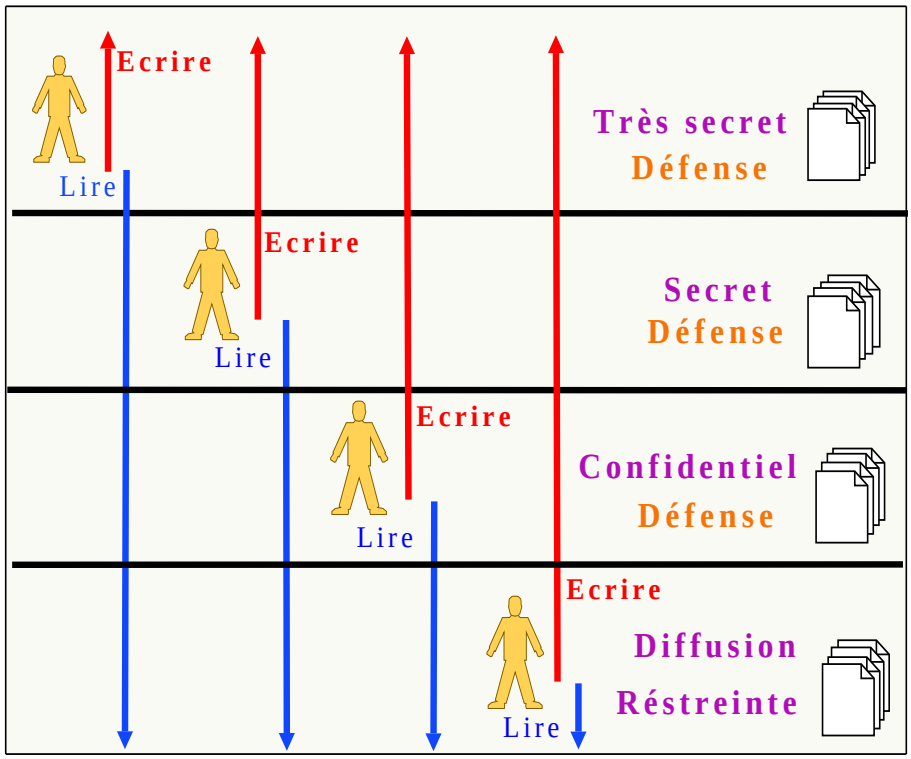


FIG. 1.4 – Modèle Bell-Padula : contrôle du flux d'informations pour la confidentialité

Pour éviter ce genre de situations, la propriété*stricte est définie.

- **La propriété*stricte :**
  le sujet s peut écrire l'objet o seulement si $h(s) = c(o)$.

  La propriété * large et la propriété * stricte sont aussi appelées respectivement Write-up et Write-equal.

  Généralement, LBAC est utilisé quand il y a un petit nombre de classifications (n) et de catégories (m) statiques tels que le nombre total de combinaisons de classifications

et de catégories est $n * m$.

1.7.2.2 Le modèle de Biba

Un modèle complémentaire à celui de Bell et La Padula appelé « Modèle de Biba », s'attache à protéger l'intégrité des informations. Pour cela il se base sur les deux principes suivants :

- **read up :** le sujet s peut lire l'objet o seulement si $h(s) \leq c(o)$.
- **write down(*-property) :** un sujet s doit être associé à un niveau de sécurité supérieur à celui de l'objet o qu'il souhaite écrire c-à-d $c(o) \leq h(s)$.

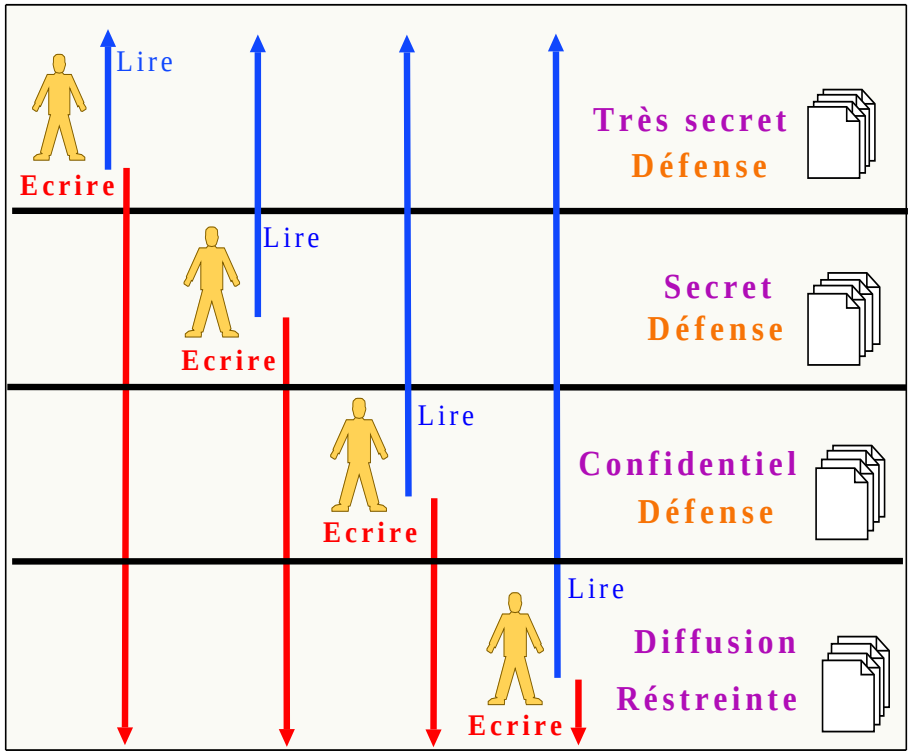


FIG. 1.5 – Modèle de Biba : contrôle du flux d'informations pour l'intégrité.

Ce modèle prévient les modifications inattendues, ou non autorisées des informations sensibles par des sujets non qualifiés, ou n'ayant pas un niveau de confiance suffisant. Inversement, ce modèle ne résout pas le problème du Cheval de Troie, c'est-à-dire la fuite d'informations vers le bas qui peut être résolu de la même manière que pour le modèle Bell-La Padula en modifiant « la propriété étoile » afin de n'autoriser l'écriture à un sujet que pour les objets de même niveau.

1.7.2.3 La combinaison des deux modèles

Il est parfois suggéré d'utiliser ces deux modèles conjointement afin de protéger d'une part, la confidentialité (Bell-La Padula) et d'autre part, l'intégrité (Biba) de l'information. Ces modèles reposent sur des principes contradictoires et non compatibles, ils doivent donc être modifiés afin de pouvoir se combiner :

- Les sujets et les objets sont tous deux associés à deux niveaux de sécurité : l'un concernant la confidentialité appelé « niveau de confidentialité » et l'autre concernant l'intégrité, « le niveau d'intégrité ».
- Un sujet ne peut lire un objet que si son niveau de confidentialité est supérieur ou égal au niveau de l'objet et que son niveau d'intégrité est inférieur ou égal à celui de l'objet.
- Un sujet ne peut écrire dans un objet seulement si son niveau de confidentialité est dominé ou égal à celui de l'objet et que son niveau d'intégrité domine (ou égale) celui de l'objet.

Dans le cas où l'on emploie un niveau de sécurité unique, cela revient à n'autoriser aux sujets la lecture et l'écriture que dans les objets ayant le même niveau.

1.7.2.4 Les limitations

Les modèles de sécurité multi-niveaux sont plus approprié aux environnements qui ont des contraintes de sécurité statiques, mais ils ne peuvent pas être employés pour satisfaire les exigences des contraintes dynamiques pour les applications et les systèmes d'information réparties à grande échelle [13, 14] . La séparation des tâches (SoD) et les contraintes de dépendance sont des exemples de telles contraintes dynamiques, et sont exigées dans la plupart des applications commerciales, comme le E-commerce,les systèmes de santé [3].

1.7.3 Le contrôle d'accès basé sur les rôles (RBAC)

Les politiques obligatoires imposent des contraintes fortes sur les organisations. Les relations d'ordre partiel qu'elles utilisent sont mal adaptées à la réalité des entreprises : peut-on dire qu'une information du service commercial est plus secrète ou plus critique que celle qui provient du bureau d'étude ? Ce n'est pas sur de tels critères, qu'il convient de contrôler les flux d'information. Les politiques discrétionnaires sont elles aussi mal adaptées : elles sont trop laxistes en général, et permettent difficilement de garantir des propriétés de sécurité. D'autre part, il faut constamment redéfinir les règles, à chaque fois qu'un nouvel utilisateur ou un nouvel objet est introduit dans le système. Les politiques discrétionnaires sont donc très difficiles à administrer.

À l'inverse, le contrôle d'accès basées sur les rôles (RBAC, pour Role-Based Access Control) visent à faciliter l'administration de la sécurité. Un rôle représente de façon abstraite une fonction identifiée dans l'organisation (par exemple, chef de service, ingénieur d'étude, docteur, infirmier, etc). À chaque rôle, on associe des permissions, privilèges, ou un ensemble de droits correspondant aux tâches qui peuvent être réalisées par chaque rôle. Enfin, et contrairement aux modèles qui ont précédé RBAC, les permissions ne sont plus associées d'une façon directe aux utilisateurs, mais à travers les rôles (les permissions sont associées aux rôles, et les utilisateurs sont faits des membres aux rôles appropriés).

Cela simplifie énormément la gestion des permissions. Les rôles sont créés pour les diverses fonctions dans une organisation et les utilisateurs se sont attribués les rôles selon leurs responsabilités et qualifications. Les utilisateurs peuvent facilement être réaffectés d'un rôle à l'autre. Les rôles peuvent s'accorder de nouvelles permissions à mesure que de nouvelles applications et systèmes sont intégrés, et les permissions peuvent être retirés à des rôles en fonction des besoins. Le rôle est plus stable parce que les activités ou les fonctions d'une organisation changent habituellement moins fréquemment.

Les deux relations de la figure 1.6 : Détient (Rôle, Permission) et Joue (Utilisateur, Rôle) définissent précisément les permissions accordées à chaque utilisateur [8, 28].

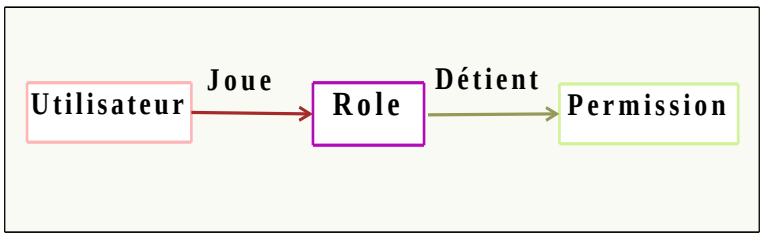


FIG. 1.6 – Le concept de base de RBAC

1.7.3.1 Les notions de base

Le modèle RBAC [27] est basé sur trois ensembles d'entités :

- **Les utilisateurs (U) :**
  Un utilisateur dans ce modèle est un être humain. Le concept d'un utilisateur peut être généralisé pour inclure des agents autonomes intelligents comme des robots, des ordinateurs immobiles,etc.
- **Les rôles (R) :**
  Un rôle est une fonction ou travail dans une entreprise avec quelques sémantiques concernant l'autorité et les responsabilités conférés à un membre du rôle.

- **Les permissions (P) :**
  Une permission est une approbation d'un mode particulier d'accès à un ou plusieurs objets dans le système. Les termes autorisation, droit d'accès et privilège sont également utilisés dans la littérature pour désigner une permission. Les permissions sont toujours positives et confèrent la capacité au détenteur de la permission d'exécuter une action (s) dans le système.

  On peut définir deux fonctions sur les ensembles utilisateurs, rôles et permissions [28, 27] :
- **L'assignation des utilisateurs (UA : User Assignment) :**
  Représentent l'assignation des utilisateurs aux rôles.
- **L'assignation des permissions (PA : Permission Assignment) :**
  Représentent l'assignation des permissions aux rôles.

La figure 1.10 montre les relations d'assignation des utilisateurs (UA) et d'assignation des permissions (PA) . Tous les deux sont des relations "plusieurs à plusieurs". Un rôle peut avoir plusieurs permissions et une permission peut être associée à plusieurs rôles. De même qu'un utilisateur peut être membre de plusieurs rôles et inversement, un rôle peut être exécuté par plusieurs utilisateurs. Ainsi, si le docteur Dupont est à la fois chirurgien et directeur de l'hôpital, en tant que chirurgien, il aura le droit d'accès aux dossiers médicaux, alors qu'en tant que directeur, il pourra accéder aux informations administratives.

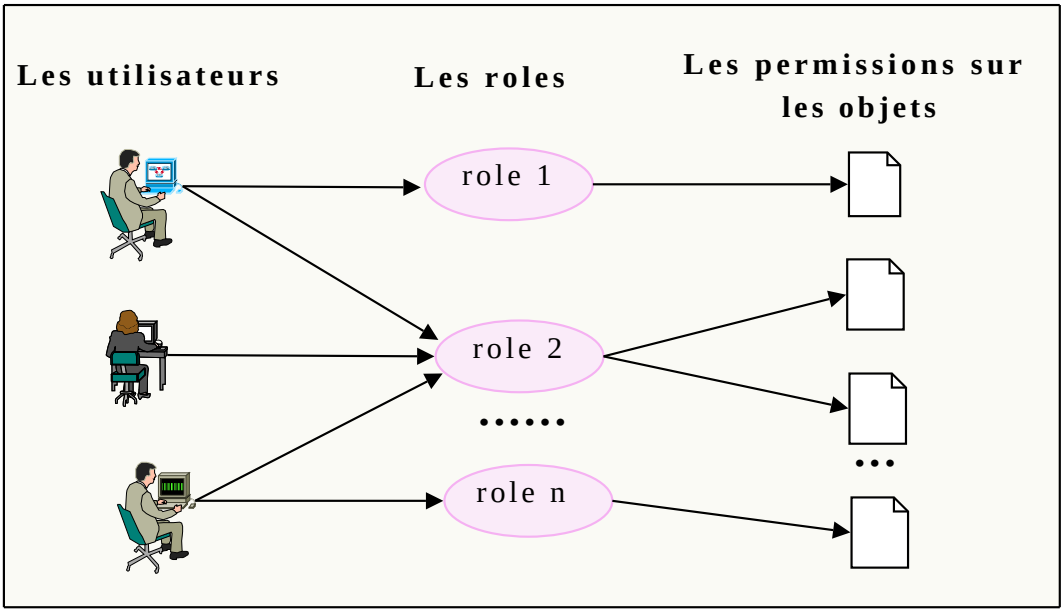


FIG. 1.7 – La politique de contrôle d'accès basée sur les rôles.

L'une des manières de définir les rôles au sein d'une organisation consiste à regrouper dans chaque rôle les tâches pouvant exécuter les mêmes opérations, ensuite il s'agit d'identifier les

objets que ces tâches utilisent, puis définir les droits d'accès sur ces objets, c'est-à-dire les couples (ensemble de droits, objets) et finalement, d'associer ces droits aux rôles. L'affectation des sujets aux rôles est une tâche à faire séparément, et probablement par d'autres administrateurs.

Le modèle RBAC introduit les notions de session, de hiérarchie de rôles et de contraintes sur les rôles [8, 28]

1.7.3.2 La session

Pour pouvoir réaliser une action sur un objet, un utilisateur doit d'abord créer une session et dans cette session, active le rôle qui a eu l'autorisation de réaliser cette action. L'utilisateur doit activer un ou plusieurs rôles parmi les rôles qui lui ont été attribués, comme il est montré dans la figure 1.10 par la double flèche dirigée de la session S aux rôles R, indique que des multiples rôles sont simultanément activés. Dans une même session, un utilisateur a la possibilité de ne pas activer tous ses rôles, mais uniquement le sous ensemble de ses rôles nécessaire à la réalisation de la tâche à accomplir. Une session est associée à un seul utilisateur, comme il est indiqué dans la figure 1.10 par la flèche simple dirigée de la session S à l'utilisateur U. Les permissions disponibles pour un utilisateur sont l'union des permissions de tous les rôles activés pendant sa session. Un utilisateur peut ouvrir en même temps plusieurs sessions.

1.7.3.3 La hiérarchie des rôles

La hiérarchie existe naturellement dans de nombreuses organisations, basées sur le principe de généralisation et la spécialisation [3]. La hiérarchie des rôles permet de mettre en place un mécanisme d'héritage des permissions entre les rôles, et simplifie d'autant l'administration de ce modèle. Par exemple, comme les chirurgiens et les gynécologues sont nécessairement des médecins, on assignera les permissions spécifiques des médecins au rôle médecin, et seulement les permissions supplémentaires qui caractérisent les chirurgiens au rôle chirurgien, d'une part et les permissions supplémentaires qui caractérisent les gynécologues au rôle gynécologue d'autre part.

Dans cet exemple, le rôle chirurgien et le rôle gynécologue sont implicitement associés aux permissions du rôle médecin sans que l'administrateur inscrivent explicitement les attributs du rôle médecin. On appelle les rôles chirurgien et gynécologue des rôles seniors, et le role médecin, le rôle junior, tel que le rôle senior hérite de toutes les permissions de son rôle junior. L'héritage des permissions est ainsi transitive , par exemple, dans la figure 1.8, le rôle chirurgien esthétique hérite les permissions des rôles chirurgien et médecin.

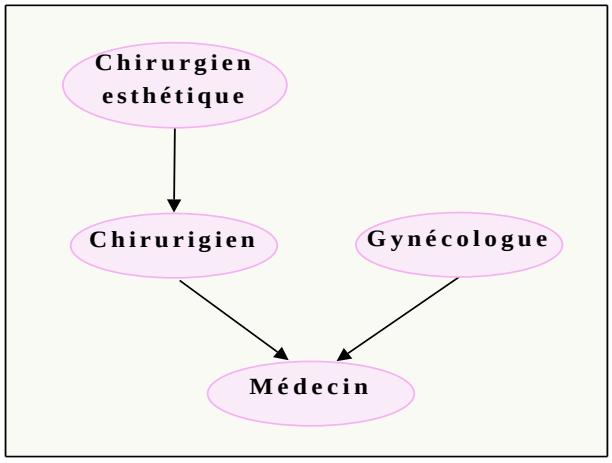


FIG. 1.8 – L'héritage des rôles.

Mathématiquement, la hiérarchies est une relation d'ordre partiel. Une relation d'ordre partiel est une relation reflexive, transitive et anti-symétrique. L'héritage est reflexive, car un rôle hérite ses autorisations, la transitivité est une exigence naturelle dans ce contexte, et l'anti-symétrie exclut les rôles d'hériter l'un de l'autre et seraient donc des rôles redondants.

Dans le modèle RBAC il n'est pas obligatoire de mettre les liens entre tous les rôles, la figure 1.8 montre que les rôles chirurgien et gynécologue ne sont pas hiérarchiquement liés.

Il y a deux types d'hiérarchie des rôles :

- Hiérarchie d'héritage.
- Hiérarchie d'activation.

1.7.3.3.1 La hiérarchie d'héritage

La hiérarchie d'héritage, dénotée par *I-hiérarchie*, permet à un utilisateur activant un rôle senior d'hériter toutes les permissions de ses rôles juniors sans les activer. Dans l'exemple montré à la figure 1.9, si les relations hiérarchiques entre les rôles sont les hiérarchies d'héritage, alors quand le rôle *CP* est activé, les permissions assignées à *CP*, *IP*, *IQ*, *I* et *ST* sont disponibles à l'utilisation.

1.7.3.3.2 La hiérarchie d'activation

Le modèle original de RBAC ne supporte que la hiérarchie d'héritage, ce qui permet aux utilisateurs d'un rôle senior d'hériter de toutes les permissions de ses rôles juniors. Afin de

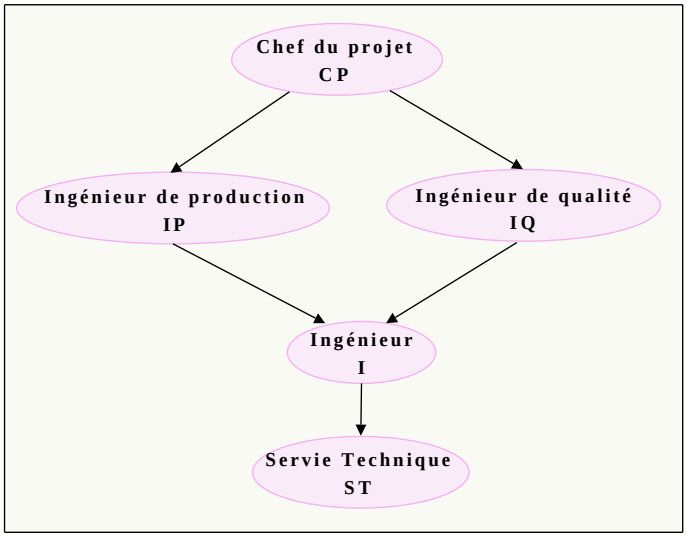


FIG. 1.9 – Exemple de l'hiérarchie d'héritage.

préserver le principe du moindre privilège, le modèle RBAC a été étendu pour inclure la hiérarchie d'activation qui permet à un utilisateur d'activer un ou plusieurs rôles juniors, sans activation de ses rôles seniors [28].

La hiérarchie d'activation, dénotée par *A-hiérarchie*, ne soutient pas la sémantique d'héritage de permissions. Dans l'*A-hiérarchie*, n'importe quel utilisateur assigne un rôle senior, a le droit d'activer tous ses rôles juniors. L'utilisateur est seulement autorisé à acquérir les permissions qui sont directement assignées au rôle activé. Dans l'exemple de la figure 1.9, si les relations hiérarchiques entre les rôles sont les hiérarchies d'activation, alors l'activation du rôle senior *CP* n'active pas les permissions de ses rôles juniors. Chaque rôle junior doit être explicitement activé pour que l'utilisateur puisse acquérir ses permissions.

1.7.3.4 La séparation des tâches(SoD, Separate of Duty)

La séparation des tâches (SoD) a été considérée comme une exigence importante de sécurité des organisations [3]. Les contraintes SoD sont appliquées principalement pour éviter les fraudes dans les organisations. Il s'agit d'interdire un même utilisateur d'effectuer deux tâches normalement séparées. RBAC peut être utilisé pour appliquer facilement ces exigences tant statiquement que dynamiquement.

a. La séparation statique des tâches

Un utilisateur est autorisé à posséder un rôle, si ce rôle n'est pas mutuellement exclusif avec

aucun des autres rôles que l'utilisateur a deja les assignés. Par exemple : les rôles médecin et infirmier ne peuvent pas être assignés au même utilisateur.

b. La séparation dynamique des tâches

Un utilisateur peut activer un nouveau rôle, si ce rôle n'est pas mutuellement exclusif avec aucun des rôles que l'utilisateur met actuellement en activité. Par exemple : les rôles médecin libéral et chirurgien ne peuvent être activés en même temps par un même utilisateur.

Dans [3] , on trouve une autre classification des contraintes SoD et un nouveau type de SoD :

a. Contraintes SoD spécifiques des rôles
Interdit la possession et-ou l'activation des rôles contradictoires au même utilisateur.

b. Contraintes SoD spécifiques des utilisateurs
Interdit à des utilisateurs contradictoires d'assumer le même rôle simultanément.

1.7.3.5 La cardinalité des rôles

Elle limite le nombre d'utilisateurs qui peuvent posséder ou activer un même rôle. Par exemple : il ne peut y avoir qu'un seul utilisateur qui peut posséder le rôle comptable, et cinq utilisateurs au maximum qui peuvent activer le rôle guichetier simultanément, etc.

Les concepts du modèle RBAC sont résumés dans la figure 1.10 et peuvent être formalisés comme suit [27] :

- U, R, P, S,représentent respectivement des ensembles des utilisateurs, des rôles, des permissions et des sessions ;
- PA $\subset$ R*P, une relation associant une permission à un rôle
- UA $\subset$ U*R, une relation associant un ou plusieurs rôles à un utilisateur ;
- RH $\subset$ R*R, une hiérarchie de rôles partiellement ordonnés ; r $\geq$ r' signifie que r' est un sous-rôle de r ;
- User : S $\rightarrow$ U, une fonction associant chaque session s_i à un seul utilisateur User(s_i), qui reste constant pour la durée de vie de la session ;
- Role : S $\rightarrow R^2$, une fonction associant chaque session s_i à un ensemble de rôles Role(s_i), avec : Role(s_i) $\subset$ {r $|(\exists r', r' \geq r)$ et (User(s_i), r') $\in$ UA}.
- une collection de contraintes qui détermine si certains éléments du modèle RBAC sont acceptables (seuls les éléments acceptables sont intégrés dans le modèle).

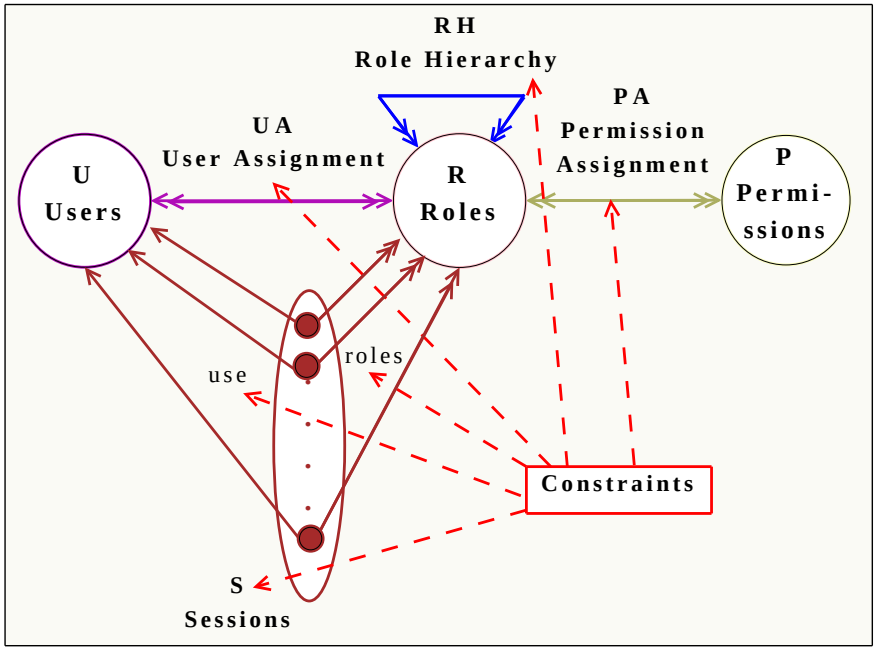


FIG. 1.10 – Le modèle RBAC [27]

1.7.3.6 La représentation de la politique RBAC par un graphe

On peut décrire la politiques de contrôle d'accès RBAC d'un domaine, par un graphe [17], tel que :

- Les utilisateurs, les rôles et les permissions sont représentés par des nœuds.
- Les arcs représentent les relations entre ces nœuds, comme le montre le schéma ci-dessous.

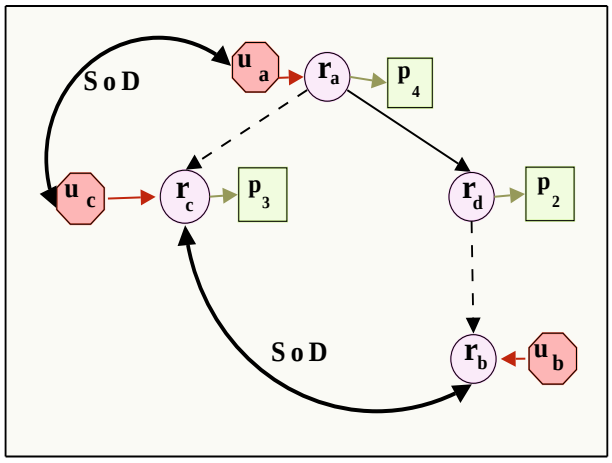


FIG. 1.11 – La représentation graphique du modèle RBAC.

– Un arc entre un nœud-utilisateur u_i et un nœud-rôle r_i indique que le rôle r_i est assigné à l'utilisateur u_i.
– La hiérarchie d'héritage des rôles (*I-hiérarchie*) est représentée par un arc continu.

- La hiérarchie d'activation des rôles (*A-hiérarchie*) est représentée par un arc discontinu.
- Un arc entre un nœud-permission p_i et un nœud-rôle r_i représente l'assignation de la permission p_i au rôle r_i.
- Le SOD spécifique des rôles est représenté par une double flèche entre les rôles correspondants.
- Le conflit entre deux utilisateurs u_i et u_j pour un rôle r_j, est représenté par une double flèche entre les nœuds-utilisateurs u_i et u_j, étiquetée par r_j.

1.7.3.7 Les Limitations

Les principaux inconvénients du modèle RBAC selon [16] sont les suivants :

1. Il n'est pas possible de définir que les permissions et non les interdictions, ou les obligations.
2. Il n'est pas possible dans le modèle RBAC d'exprimer des permissions qui dépendent du contexte. Plus précisément, si une certaine permission est accordée à un rôle, alors tous les utilisateurs qui jouent ce rôle héritent de cette permission. Par conséquent, il n'y a aucun moyen de spécifier qu'un médecin n'a la permission d'accéder au dossier médical d'un patient que si ce dernier est son patient.

1.7.4 Les autres techniques de contrôle d'accès

1.7.4.1 Le contrôle d'accès basé sur les tâches (TBAC)

Dans les modèles DAC, MAC et RBAC , les actions correspondent généralement à des commandes élémentaires, comme la lecture du contenu d'un objet ou l'écriture dans un objet. Dans les applications récentes, le besoin apparaît de contrôler la réalisation d'actions composites, appelées tâches ou activités. Par exemple, dans une agence de voyage, la tâche d'achat d'un billet d'avion se décompose en plusieurs actions plus élémentaires telles que la réservation du billet, le paiement du billet et l'édition d'une facture. Le besoin de contrôle sur des activités composites est en particulier présent dans les applications de type workflow. Le modèle T-BAC (Task Based Access Control) fut le premier modèle à introduire le concept de tâche. D'autres modèles ont ensuite été développés pour contrôler l'exécution des activités dans un workflow. En particulier, l'utilisateur ne doit obtenir une permission que lorsque c'est nécessaire pour poursuivre l'exécution de l'activité considérée ("just in time" permission). Ainsi, dans l'exemple d'achat d'un billet d'avion, la permission d'éditer une facture ne doit être activée qu'après la réservation et l'achat du billet. Il est parfaitement possible de combiner les concepts

de rôle et de tâche pour spécifier une politique d'autorisation et obtenir ainsi un modèle que l'on peut appeler TR-BAC (Task and Role Based Access Control). Dans ce cas, les permissions affectées aux rôles portent sur la réalisation des tâches. Diverses contraintes peuvent être spécifiées pour par exemple spécifier qu'un même sujet doit intervenir dans certaines actions nécessaires à la réalisation de la tâche (éventuellement avec des rôles différents) [1, 25, 4, 2, 4].

1.7.4.2 Le contrôle d'accès basé sur les organisations (Or-BAC)

L'objectif d'Or-BAC (Organization Based Access Control) est de permettre la modélisation d'une variété de politiques de sécurité, basées sur le contexte de l'organisation. L'organisation pouvant être divisée en sous-organisations, il peut s'avérer intéressant de spécifier plusieurs politiques de sécurité : pour les médecins d'un hôpital par exemple, les règles diffèrent selon la structure dans laquelle ils se trouvent : dans un service « classique », un médecin ne pourra voir que les dossiers de ses patients, mais aux urgences, tous les dossiers doivent être accessibles, etc.

DAC, MAC, RBAC et TBAC n'offrent pas la possibilité de nuancer les autorisations, de manière à spécifier des permissions, des obligations, des interdictions ou encore des recommandations. Ceci est pourtant très utile pour exprimer des politiques de sécurité dans le domaine médical [16].

Or-BAC s'attache donc à répondre à ces deux besoins, à savoir la spécification de plusieurs politiques de sécurité dans une organisation (prise en compte du contexte), et la définition plus fine des autorisations.

1.8 Le langage XACML

XACML (eXtensible Access Control Markup Language) est un langage standardisé par *OASIS*[1] basé sur XML et dédié pour spécifier et appliquer des politiques de contrôle d'accès. Dans ce langage, toute entité concernée par le contrôle d'accès (i.e. sujets, ressources, actions et environnement) est spécifiée par un ensemble d'attributs [18]. Le standard fournit également une architecture qui définit différentes entités impliquées dans le processus de prise de décision d'une autorisation d'accès [21].

Pour ce faire, ce langage inclut des types de données, des fonctions ainsi que de la logique.

[1] www.OASIS-open.org

Afin d'effectuer une action sur une ressource, un demandeur va envoyer une requête au près de l'entité protégeant la ressource appelée Policy Enforcement Point (PEP). Ce dernier va construire une requête en XACML en fonction des divers attributs du demandeur, de l'action demandée, de la ressource concernée et de toute autre information pertinente. Une fois ceci fait, il va l'envoyer à un tiers de decision appelé Policy Decision Point (PDP) qui va verifier s'il existe une règle accordant l'accès à cette ressource pour cette action à cet utilisateur et renvoyer une réponse également en XACML au PEP qui va accorder ou non en consequence l'accès au demandeur (voir la figure 1.12) [21, 6].

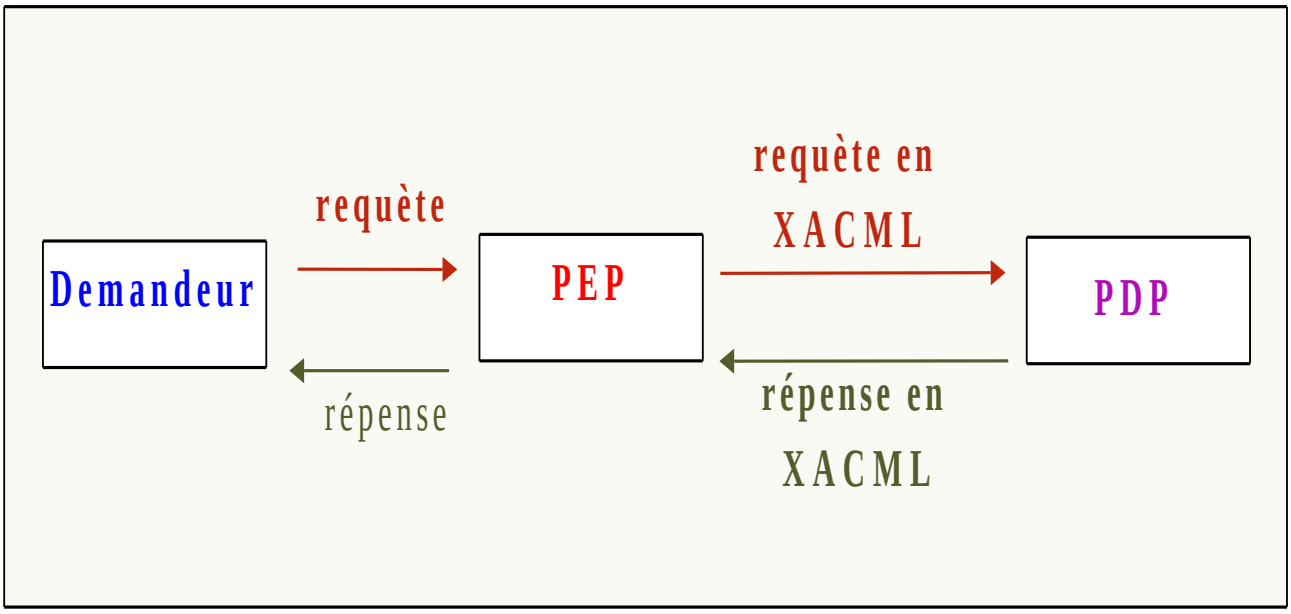


FIG. 1.12 – Le principe général de XACML

La naissance de XACML vise à assurer un accès sécurisé aux ressources de données hétérogènes et dynamiques. C'est donc un langage pour l'interopérabilité (syntaxique) mais il ne résout pas les problèmes sémantiques de la coopération.

1.9 Conclusion

L'analyse des politiques basées sur les rôles permet de conclure qu'elles sont relativement faciles à administrer et suffisamment souples pour s'adapter à chaque organisation . En effet, la définition des rôles peut refléter la structure de l'organisation. Les rôles peuvent être structurés de façon hiérarchique, pour simplifier encore l'affectation des permissions. Avec RBAC, il est facile d'ajouter un utilisateur : il suffit de lui assigner les rôles qu'il doit jouer dans l'organisation. De même, il est relativement facile de faire évoluer les tâches suite à la création ou la modification d'un objet : il suffit de mettre à jour les privilèges des rôles concernés.

Notre travail est basé sur ce modèle. Dans les chapitres suivants, Nous allons voir comment on peut établir le contrôle d'accès dans les environnements multi-domaines en basant sur le modèle RBAC, mais avant ça nous devons connaître c'est quoi un environnement multi-domaines, et c'est ça l'objectif de deuxième chapitre.

Chapitre 2

L'interopérabilité dans les environnements multi-domaines

2.1 Introduction

Ce chapitre est une introduction au problème de gestion de contrôle d'accès dans les environnements multi-domaines. D'abord, nous présentons c'est quoi un environnement multi-domaines, puis nous passerons aux difficultés qu'on doit traiter, pour qu'on puisse établir une politique RBAC globale et cohérente, qui est capable de gérer les accès aux ressources et données confidentielles de cet environnement d'une manière fiable.

2.2 La problématique du contrôle d'accès multi-domaines

Un environnement multi-domaines est un ensemble d'organisations autonomes, distribuées, hétérogènes, utilisant des politiques de sécurité différentes, et collaborant entre elles. Les différentes organisations formant des domaines indépendants visent non seulement à partager les ressources mais également un accès sûr aux ressources partagées. Ce qui exige une structure de sécurité qui doit former, administrer et appliquer une politique de sécurité globale, compatible avec toutes les politiques des domaines participants.

Cette politique peut être établie en intégrant les politiques de contrôle d'accès des domaines collaborants. La composition d'une politique interopérable, à partir des diverses politiques des différents domaines, comporte des défis variés qui ont trait à la réconciliation des différences sémantiques, à la sécurisation de l'interopérabilité, aux pertes d'autonomie, et au confinement d'une éventuelle propagation de risques.

L'interopérabilité des politiques de sécurité de domaines collaborants est sûre, si les deux principes suivants sont respectés [10] :

- **Principe d'autonomie :**
  N'importe quel accès autorisé dans un système individuel, doit être également autorisé sous l'interopérabilité sûre.

- **Principe de sécurité :**
  N'importe quel accès non autorisé dans un système individuel, doit également être refusé sous l'interopérabilité sûre.

2.3 Le modèle de contrôle d'accès

Il est nécessaire d'utiliser un modèle de contrôle d'accès, qui peut être employé pour représenter uniformément les politiques de contrôle d'accès des domaines collaborants. Un tel modèle doit permettre l'interopération et le partage d'informations entre les multiples domaines, et en même temps garantir que de tels accès inter-domaines ne violent pas les politiques fondamentales des domaines constitutifs.

Dans la littérature, le problème de l'interopérabilité sûre a été abordé dans le contexte du modèle de sécurité multi-niveaux [10]. Mais ce modèle présente des inconvénients pour modéliser les contraintes dynamiques des systèmes et des applications réparties à grande échelle (voir la section 1.7.2.4).

Le modèle RBAC en raison de sa richesse dans la modélisation hiérarchique, la séparation de tâches (SOD), les cardinalités et les contraintes de dépendance, apparaît comme un modèle prometteur pour satisfaire les exigences de contrôle d'accès dans la plupart des applications commerciales [27].Pour les mêmes raisons, RBAC a été choisi dans ce projet pour exprimer les politique de sécurité des domaines collaborants.

2.4 Exemple d'illustration

Dans cette section, nous présentons un exemple d'un environnement multi-domaines. Cet environnement est une collaboration entre divers départements d'un comté : le département des services financiers *CTO* (County Treasure Office), le département des services du personnel *CCO* (County Clerk Office), et le département des services judiciaires *CAO*(County Attorney

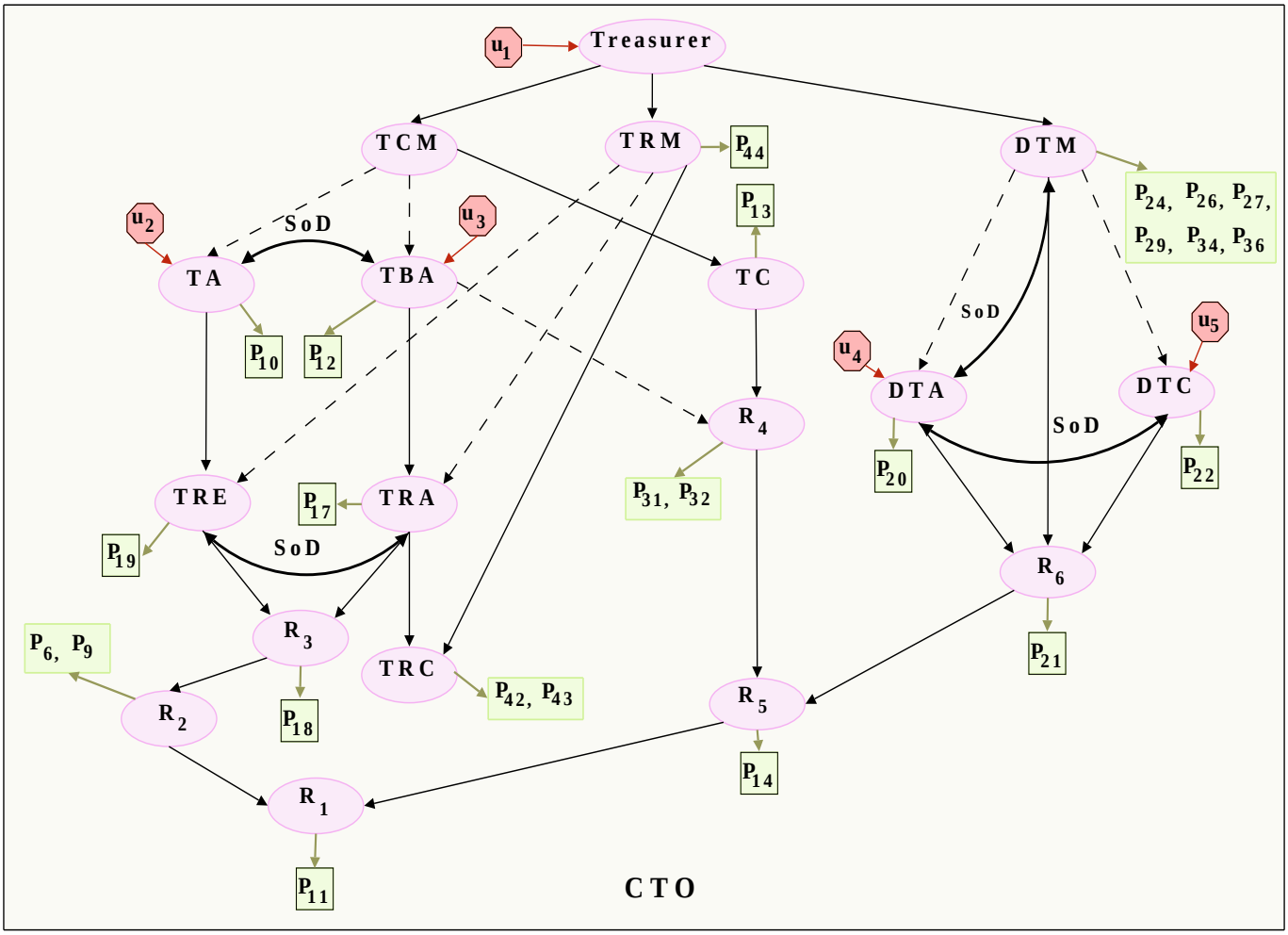
Office), pour la perception et la vente des impôts en états réel sur les propriétés situées dans la juridiction de ce comté [3].

Ces départements partagent des informations entre eux pour la planification de budget, la facturation des impôts, la perception des impôts, la vente des fraudes fiscales, et l'audit, etc.

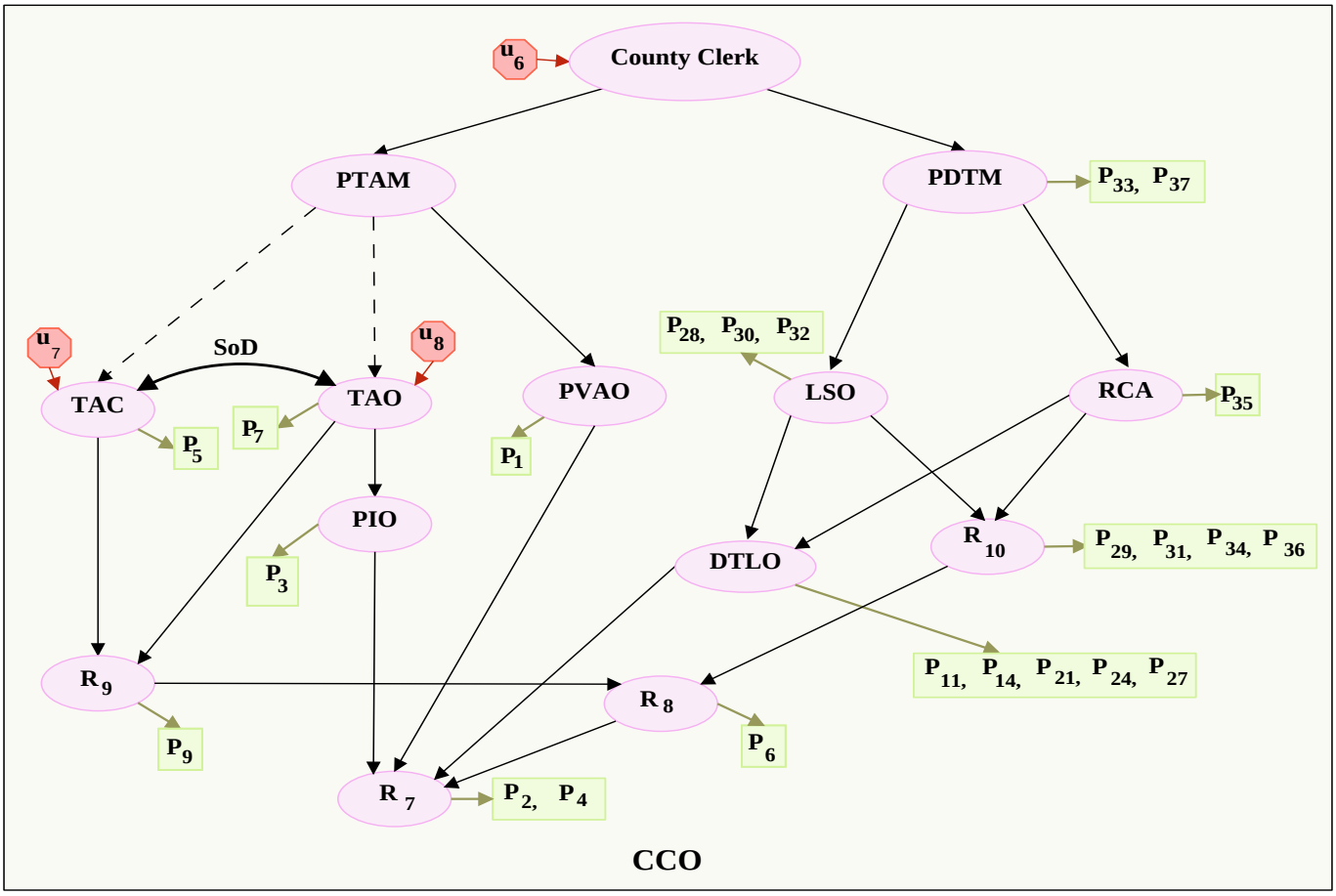
Chaque département du comté préserve les informations en sa possession dans ses bases de données locales. L'intégration de ces bases de données locales est nécessaire pour fournir la possibilité d'accès inter-domaines aux informations.

Une telle intégration accélère non seulement le processus de perception des impôts et de vente en fournissant l'accès immédiat aux informations opportunes, précises et complètes mais améliore aussi la productivité de personnel existant en réduisant les efforts de collection des données redondantes entre les départements du comté.

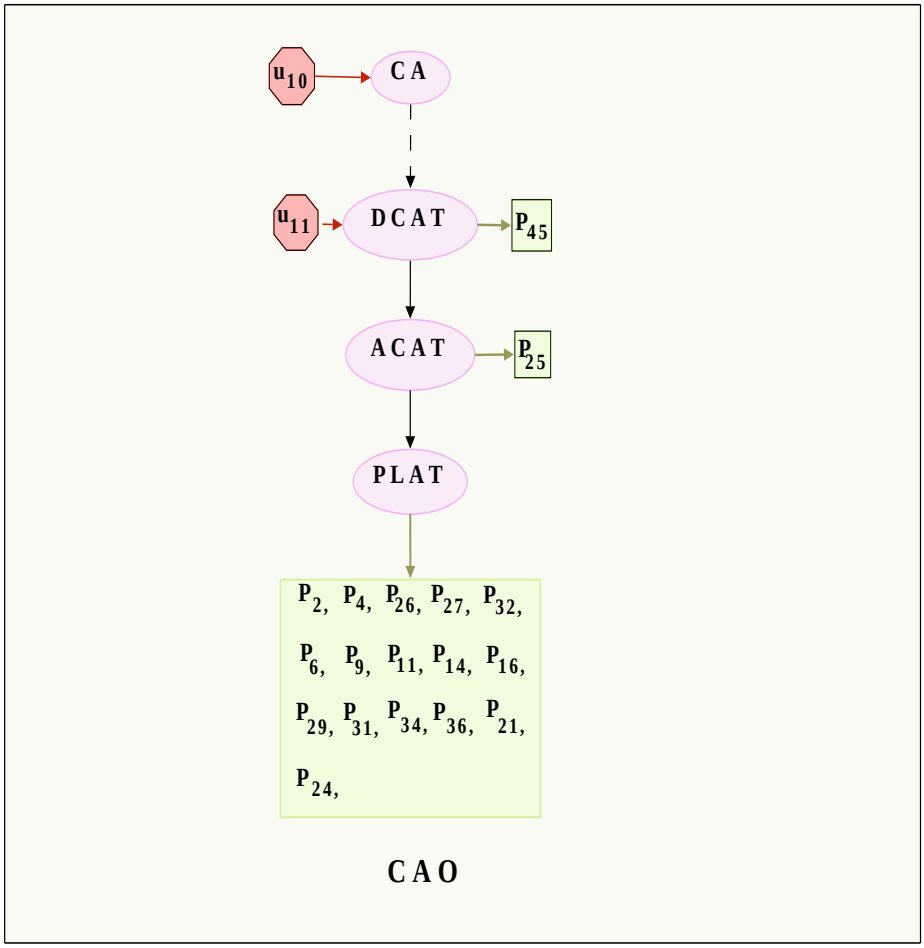
Les politiques de contrôle d'accès RBAC de chaque département *CTO*, *CCO*, et *CAO* sont notées dans la figure 2.1. Le tableau 2.2 décrit chaque rôle de ces politiques RBAC.


u1
Treasurer
TCM
TRM
P44
DTM
P24, P26, P27, P29, P34, P36
u2
SoD
u3
P13
TA
TBA
TC
SoD
u5
u4
P10
P12
DTA
DTC
R4
SoD
TRE
P17
TRA
P20
P22
P31, P32
P19
SoD
R6
R3
P6, P9
TRC
P21
R2
P18
P42, P43
R5
P14
R1
P11
CTO

(a)


u6
County Clerk
PTAM
PDTM
P33, P37
u7
SoD
u8
P28, P30, P32
TAC
TAO
PVAO
LSO
RCA
P35
P5
P7
P1
PIO
R10
P29, P31, P34, P36
P3
DTLO
R9
P11, P14, P21, P24, P27
R8
P9
P6
R7
P2, P4
CCO

(b)



(c)

FIG. 2.1 – (a)La politique RBAC du département des services financiers *CTO*, (b) du département des services du personnel*CCO*, et (c) du département des services judiciaires *CAO*

Rôle	**Abréviation sur le graphe RBAC**	**Domaine**	**Description du rôle**	**Permission associées**
Trésorier	Treasurer	*CTO*	Surveiller toutes les opérations du bureau de trésorier	Hérite de toutes les permissions de *TCM*, *TRM*, et de *DTM*.
Contrôleur des impôts	*TA*(Tax Assessor)	*CTO*	Évaluer et préparer les factures des impôts.	P_6, P_9, P_{10}, P_{11}.

Approbateur des factures des impôts	*TBA*(Tax Bill Approver)	*CTO*	Réévaluer et approuvez les factures d'impôts.	P_6, P_9, P_{10}, P_{11}, P_{12}
Percepteur	*TC*(Tax Collector)	*CTO*	Perception et vente des impôts, préservation des registres des soumissionnaires des impôts	P_{11}, P_{13}, P_{14}, P_{31}, P_{32}
Directeur de perception des impôts	*TCM*(Tax Collector Manager)	*CTO*	Surveiller *TA*, *TBA* et *TC*	Hérite de toutes les permissions autorisées des *TA*, *TB*, et *TC*.
Contrôleur de remboursement d'impôt	*TRA*(Tax Refund Assessor)	*CTO*	Évaluer, et préparer des commandes de remboursement d'impôt	P_6, P_9, P_{11}, P_{17}, P_{18}
Examinateur de remboursement d'impôt	*TRE*(Tax Refund Examiner)	*CTO*	Réévaluer/approuver les commandes de remboursement.	P_6, P_9, P_{11}, P_{18}, P_{19}
Employé du bureau de Remboursement d'impôt	*TRC*(Tax Refund Clerk)	*CTO*	Préparer des bons de remboursement	P_{42}, P_{43}
Directeur de Remboursement d'impôt	*TRM*(Tax Refund Manager)	*CTO*	Approuver des bons de remboursement	P_{42}, P_{43}, P_{44}
Employé du bureau des fraudes Fiscales	*DTC*(Delinquent Taxes Clerk)	*CTO*	Conserver les registres des fraudes fiscales	P_{11}, P_{14}, P_{20}, P_{21}
Contrôleur des fraudes fiscales	*DTA*(Delinquent Taxes Assessor)	*CTO*	Évaluez des registres des fraudes fiscales	P_{11}, P_{14}, P_{20}, P_{21}, P_{22}
Directeur des fraudes fiscales	*DTM*(Delinquent Taxes Manager)	*CTO*	Approuver des fraudes fiscales à la vente / revente (surveiller *DTC* et *DTA*).	Héritez les permissions de *DTC* et *DTA*, P_{24}, P_{26}, P_{27}, P_{29}, P_{31}, P_{32}, P_{34}, P_{36}
Sous-préfet	*County clerk*	*CCO*	Surveille toutes les opérations de bureau d'employé	Hérite de toutes les permissions de *PTAM* et de *PDTM*.

Officier d'Évaluation de Valeur de Propriétés	*PVAO*(Property Value Assessment Officier)	*CCO*	L'évaluation de la valeur de propriétés	P_1, P_2, P_4
Employé du bureau d'évaluation fiscale	*TAC*(Tax Assessment Clerk)	*CCO*	Déterminez les taux des impôts fonciers	P_2, P_4, P_5, P_6, P_9
Officier d'évaluation fiscale	*TAO*(Tax Assessment Officier)	*CCO*	Réévaluer/approuver des taux(tarifs) fiscaux	P_2, P_4, P_6, P_7, P_9
Directeur d'évaluation des impôts fonciers	*PTAM*(Property Tax Assessment Manager)	*CCO*	Surveiller TAC et TAO	Hérite les permissions de TAC et TAO.
Officier d'indexation de propriétés	*PIO*(Property Indexing officier)	*CCO*	Indexation de propriétés	P_2, P_3, P_4
Officier des privilèges et des évasions fiscaux	*DTLO*(Delinquent Taxes and lien officier)	*CCO*	Préservation des registres des évasions fiscaux et des autres privilèges fiscaux	P_2, P_4, P_{11}, P_{14}, P_{21}, P_{24}, P_{27}.
Officier de vente des privilèges	*LSO*(Lien Sale Officier)	*CCO*	Vente des fraudes fiscales, et préserver les registres des acheteurs des impôts	Hérite les permissions de *DTLO*, P_{28}, P_{29}, P_{30}, P_{31}, P_{32}, P_{34}, P_{36}.
Contrôleur des coûts de remboursement	*RCA*(Redemption Cost Assessor)	*CCO*	Préparer les estimations de remboursement pour les fraudes fiscales	Hérite les permissions de *DTLO*, P_{29}, P_{31}, P_{34}, P_{35}, P_{36}
Directeur de propriétés de fraudes fiscales	*PDTM*(Property Delinquent Tax Manager)	*CCO*	Réévaluer/approuver des estimations de remboursement fiscaux (surveille *LSO* et *RCA*)	Hérite les permissions de *RCA* et *LSO*, P_{33}, P_{37}
Mandataire du comté	*CA*(County Attorney)	*CAO*	Dirige le département des services judiciaires	les permissions de tous ses rôles juniors.
délégué de section des impôts des services judiciaires	*DCAT* (Deputy County Attorney Tax section)	*CAO*	Évaluer et approuver la réclamation de vente des impôts	hérite les permissions de *ACAT*, P_{45}.

Assistant de section des impôts des services judiciaires	*ACAT*(asst. County Attorney Tax section)	*CAO*	Préparer les réclamations de vente d'impôts pour des évasions fiscaux et d'autres privilèges dirige les ventes d'impôt	hérite les permissions de PLAT, P_{25}
Section para légal	*PLAT*(Para legal tax section)	*CAO*	Préserver les registres des informations obtenues de *CCO* et *CTO* pour les affaires liées aux impôts , aide des avocats dans la préparation de réclamations de vente des impôts	P_2, P_4, P_6, P_9, P_{11}, P_{14}, P_{16}, P_{21}, P_{24}, P_{26}, P_{27}, P_{29}, P_{31}, P_{32}, P_{34}, P_{36}

TAB. 2.2 – La description des rôles des politiques RBAC des domaines *CTO*, *CCO*, et *CAO*.

2.5 La politique de partage des informations

Le but de l'interopérabilité est de partager les données entre les domaines collaborants et l'échange de ressources. Un domaine peut ne pas partager totalement ses objets de données et de ressources, (exemples d'objets : fichier, une relation/vue de base de données, ou un périphérique d'entrée-sortie, etc).

Pour chaque objet partageable, le contrôleur/propriétaire du domaine de l'objet doit fournir les informations suivantes :

- Les domaines qui sont autorisés à accéder à l'objet.

- La précision des exigences de l'objet avant qu'il soit partagé avec d'autres domaines. Par exemple, l'objet peut être totalement partagé ou partiellement partagé ou l'objet peut ne pas être partagé lui même, mais seulement certaines propriétés dérivées de lui sont partagées.

- Les permissions d'accès (lire, écrire, exécuter, etc.) à l'objet qui sont disponibles aux utilisateurs des domaines étrangers.

- N'importe quelle condition spécifique pour le partage, par exemple, un objet peut être partagé totalement ou partiellement avec un utilisateur inter-domaine seulement si l'utilisateur inter-domaines a l'accès local à certains attributs de l'objet dans son domaine propre.

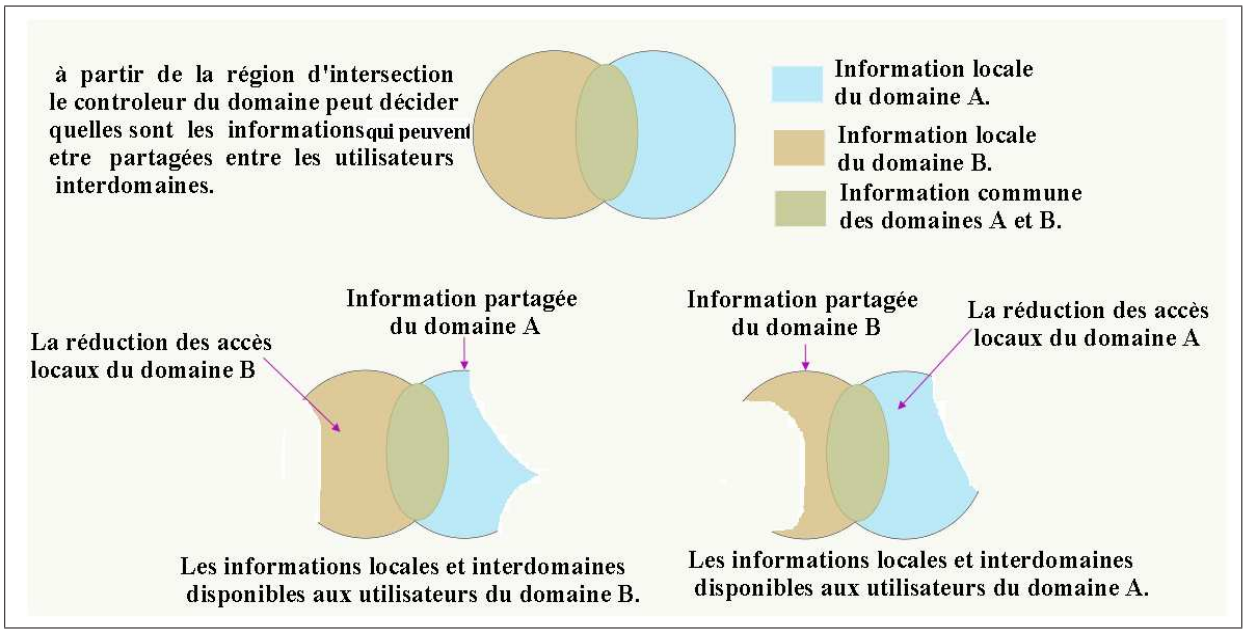


FIG. 2.2 – une vue abstraite du partage des informations inter-domaines [3].

La figure 2.2 décrit une vue abstraite du partage des informations entre les domaines. Cette figure illustre le partage partiel, qui est la forme la plus commune d'interopération et il est utilisé dans presque chaque environnement de collaboration [3]. Dans cette figure, on observe que les accès locaux aux ressources de l'information sont aussi réduits à cause du partage des ressources entre les domaines. Cette réduction des accès locaux provoque la diminution de l'autonomie des domaines collaborants.

2.6 L'hétérogénéité dans la composition des politiques

Établir l'interopérabilité dans les environnements multi-domaines n'est pas une tâche facile, différents types d'hétérogénéité peuvent existes. L'hétérogénéité est un des aspects principaux de la complexité de la composition des politiques de contrôle d'accès qui doivent être traitées comme une partie du mécanisme d'intégration de politiques.

L'hétérogénéité peut se poser en raison de conflits de nommage, et de la différence de la représentation des contraintes par les domaines collaborants.

2.6.1 Les conflits de nommage

Les conflits de nommage peuvent apparaître entre les rôles ou les objets des domaines collaborants [3], à cause de l'utilisation des homonymes, ou des noms identiques, pour représenter des entités conceptuellement différentes et des synonymes, ou des noms différents, pour représenter des entités conceptuellement équivalentes [31].

La résolution des conflits de nommage a été traitée dans la littérature dans le contexte de l'intégration des schémas dans le domaine des bases de données [31]. Ces techniques exigent l'utilisation d'un lexique global pour extraire la signification conceptuelle des attributs à partir de leurs noms. En plus, des comparaisons basées sur le domaine et l'ensemble de valeurs des attributs peuvent être effectuées pour raffiner les significations conceptuelles des attributs [19].

En plus des conflits de nommage, l'hétérogénéité peut apparaître dans la spécification de diverses contraintes de politique de contrôle d'accès : la hiérarchie, les contraintes SoD, la cardinalité et autres contraintes dynamiques. La réconciliation de la différence sémantique devient plus complexe en présence de l'hétérogénéité de contraintes.

2.6.2 Les conflits de la hiérarchie des rôles

Dans RBAC, l'hétérogénéité hiérarchique entre les politiques des domaines collaborants peut apparaître pour deux raisons :

1) L'utilisation de différents types de hiérarchies des rôles par les domaines collaborants, par exemple comme montre dans la figure 2.3 :
   A et B sont deux domaines, le domaine A contient deux rôles r_1 et r_2 tel que la relation hiérarchique entre r_1 et r_2 est la hiérarchie d'activation ($r_1 >_A r_2$), et le domaine B contient les rôles r_3 et r_4, et le rôle r_4 hérite le rôle r_3 ($r_4 >_I r_3$).
   Les rôles r_3 et r_4 sont assignés les mêmes permissions que les rôles r_1 et r_2, mais les deux rôles r_4 et r_1 ne sont pas équivalents parce que r_1 n'est pas autorisé pour la permission P_2 tant que r_4 peut directement accéder à P_2 sans activer aucun rôle junior. Les autorisations de deux rôles sont différentes parce que les deux domaines utilisent des types de hiérarchies différents.
2) Les domaines peuvent utiliser un ordre hiérarchique différent pour représenter les mêmes autorisations pour les rôles. Par exemple comme il est noté dans la figure 2.3 le rôle r_2 dans le domaine A a les mêmes autorisations que le rôle r_4 du domaine B, bien que l'ordre hiérarchique pour les deux rôles est différent. Pour le rôle r_2, les permissions P_1 et P_2 sont associées directement à lui tant que pour le rôle r_1, la permission P_1 est associée

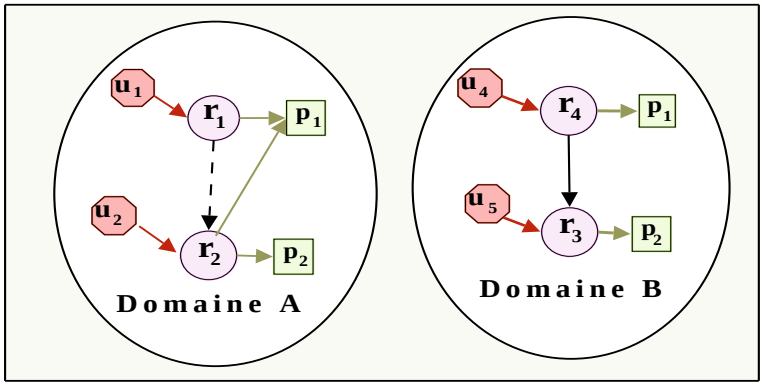


FIG. 2.3 – Les conflits de la hiérarchie des rôles

directement à lui, mais la permission P_2, il l'a obtenu par l'héritage.

2.7 Les conflits

Le deuxième principal aspect de la complexité de la composition des politiques de contrôle d'accès est l'apparition des conflits dans la politique d'interopérabilité.

Le but principal de la mise en place d'une interopération sûre est la composition d'une politique d'interopérabilité cohérente et sans conflits qui dirige toutes les informations et les ressources inter-domaines. Plusieurs efforts de recherche ont été consacrés au sujet de composition de politiques dans un environnement multi-domaines particulièrement, sur la détection et la résolution de conflits dans la politique d'interopération. Les conflits qui peuvent apparaître dans une politique d'interopérabilité peuvent être divisés en quatre types :

1. Les conflits de modalité.
2. Les conflits de gestion multiple.
3. Les conflits d'héritage cyclique.
4. La séparation des tâches (SOD).

2.7.1 Les conflits de modalité

Les conflits de modalité dans une politique apparaissent à cause de l'existence d'autorisations positives et négatives entre les sujets et les objets [20]. Les conflits de modalité sont résolus en se basant sur une relation de priorité de politique, qui implique que l'autorisation la plus stricte ignore la moins stricte. Dans le cas de relation de priorité, les conflits ne peuvent pas être établies entre les autorisations, l'autorisation négative dominant la positive.

Les conflits de modalité ne peuvent pas se produire dans une politique qui utilise le modèle RBAC étant que les autorisations négatives ne sont pas supportées dans ce modèle.

2.7.2 Les conflits de gestion multiples

Les conflits de gestion multiples se produisent lorsque plusieurs administrateurs ayant l'autorité sur un ensemble commun de sujets et d'objets, spécifient des autorisations contradictoires dans leurs politiques respectives. Dans ce sens, des conflits de gestion multiples sont semblables aux conflits de modalité [20].

2.7.3 Les conflits d'héritage cyclique

Les conflits d'héritage cyclique surviennent principalement dans l'interopération des systèmes utilisant les modèles de sécurité multi-niveaux tels que le contrôle d'accès basé sur un treillis (LBAC) et le contrôle d'accès basé sur les rôles (RBAC). Dans ces interopérations, la hiérarchie des relations entre domaines peut introduire un cycle dans le treillis d'interopérabilité permettant à un sujet plus bas dans la hiérarchie de contrôle d'accès à assumer les permissions d'un sujet situé plus haut dans la hiérarchie.

Une structure à base de médiateur a été proposée par Dawson [5] pour établir une interopération sûre entre des systèmes hétérogènes utilisant la politique LBAC. Dans cette structure, les conflits cycliques dans le treillis d'interopération sont résolus par l'intervention manuelle de l'éditeur de politique. L'éditeur de politique permet à l'administrateur de spécifier par incrémentation les relations inter-domaines de treillis. Après l'ajout de chaque relation, l'éditeur détermine la cohérence de la politique d'interopération résultante et identifie toutes les relations potentiellement incluses dans la violation de sécurité.

Les conflits cycliques sont aussi résolus en retirant toutes les relations inter-domaines potentiellement aboutissant à la violation de sécurité ou le retrait d'une ou plusieurs relations jusqu'à ce que la violation soit corrigée.

Dans le cas où il y a des multiples administrateurs de politique, un consensus sur la résolution doit être obtenu.

2.7.4 La séparation des tâches (SoD)

La séparation des tâches (SoD) empêche deux ou plusieurs utilisateurs d'accéder à un objet qui se trouve dans leur conflit d'intérêts ou bien un utilisateur d'accéder aux objets ou des permissions contradictoires, par exemple : on ne peut pas autoriser à un même utilisateur des paiements et de signer les chèques de paiement [3]. Les violations de contraintes SoD peuvent apparaître dans une politique d'interopération à cause de l'interaction des contraintes de diverses politiques à travers les domaines.

2.8 Conclusion

L'interopérabilité des politiques de contrôles d'accès des domaines différents n'est pas une tâche aisée. Différents types d'hétérogénéité peuvent exister, et différents types de conflits peuvent apparaître.

Un modèle de contrôle d'accès est nécessaire pour représenter uniformément les différentes politiques de contrôle. Notre projet est basé sur le modèle RBAC, qui est un modèle assez satisfait pour modéliser les exigences des systèmes actuels.

Dans le chapitre suivant, nous allons présenter la méthode de composition des politiques de contrôle d'accès basées sur le modèle RBAC.

Chapitre 3

La composition des politiques de contrôle d'accès RBAC

3.1 Introduction

Dans la littérature il n' y a pas beaucoup de travaux sur la composition des politiques de contrôle d'accès, les rares travaux existants se basent sur les modèles de contrôle d'accès multi-niveaux.
La seule structure de composition des politiques de contrôle d'accès, basant sur le modèle de contrôle d'accès RBAC est proposée par BASIT [3]. Dans ce chapitre, nous présentons le principe de cette méthode.

3.2 Les exigences d'intégration de politiques RBAC (PIR)

Les PIRs (Policy Integration Requirements) définissent les conditions pour que la politique RBAC d'interopération soit compatible avec les politiques RBAC des domaines collaborants. Les PIRs sont définis en se basant sur la représentation graphique du modèle RBAC [3, 24].

3.2.1 Préservation des éléments

Chaque élément (rôle, utilisateur ou permission) dans le graphe de la politique RBAC d'un domaine k doit avoir un élément correspondant dans le graphe de la politique RBAC multi-domaine.

3.2.2 Préservation des relations

Chaque relation dans le graphe de la politique RBAC d'un domaine k doit être préservée dans le graphe de la politique RBAC multi-domaines.

3.2.3 Préservation des permissions des utilisateurs

Pour chaque utilisateur u du domaine k, l'ensemble des permissions de u sur les objets du domaine k dans le graphe de la politique RBAC multi-domaines, ne doit pas être différent de l'ensemble des permissions spécifié dans la politique RBAC du domaine k.

3.2.4 Indépendance d'ordre

L'ordre dans lequel les politiques sont intégrées ne doit pas avoir l'influence sur la production de la politique d'interopération.

3.2.5 Satisfaction de contraintes

Le graphe RBAC multi-domaines doit satisfaire toutes les contraintes de politiques RBAC des domaines collaborants. En particulier, aucun accès qui aboutit à une violation des contraintes de sécurité des domaines collaborants, ne peut être permis dans le graphe RBAC multi-domaines. Les contraintes de sécurité incluent l'assignation des rôles, la hiérarchie des rôles et les contraintes SoD.

3.3 La méthode de composition proposée par BASIT [3]

Dans la méthode de composition proposée, pour produire une politique d'interopération sûre, on distingue deux étapes(3.1) :

1) La fusion des politiques.
2) La résolution des conflits.

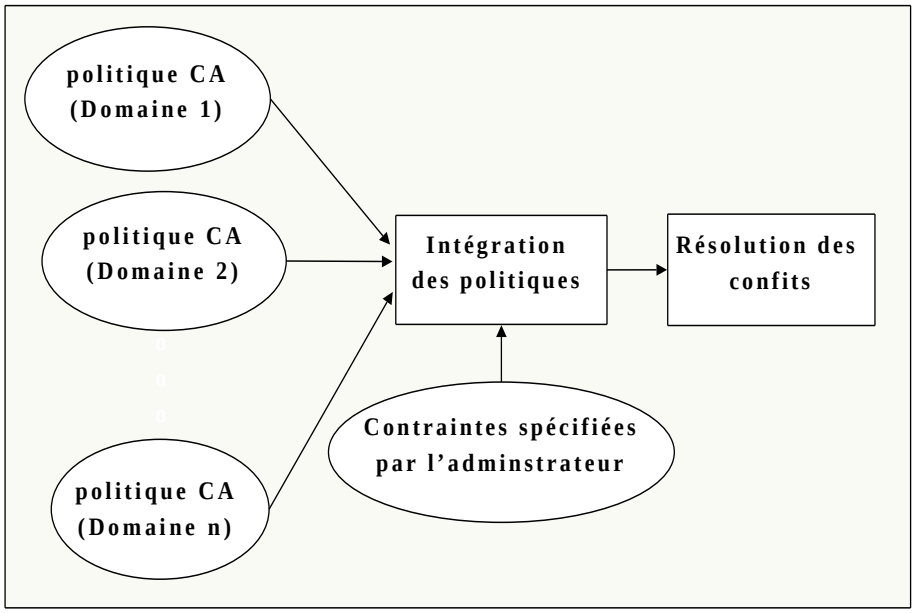


FIG. 3.1 – La structure d'intégration des politiques.

3.3.1 La fusion des politiques(Policies Merging)

La politique globale RBAC des environnements multi-domaines dirige tant des informations intra-domaines que des informations inter-domaines. Dans le contexte du modèle RBAC, l'intégration des politiques s'établie par la définition des mappings entre les rôles des domaines collaborants.

3.3.1.1 Définition des mappings des rôles inter-domaines

La fonction de mapping des rôles M_{AB} du domaine A vers le domaine B, est définie par $M_{AB} : r_A \rightarrow r_B$. Elle donne le correspondant du rôle r_A du domaine A dans le domaine B, le rôle r_B. La définition de ce mapping permet à chaque utilisateur u_i peut assumer le rôle r_A d'hériter de toutes les permissions du rôle r_B ($M_{AB}(r_A) = r_B$).

L'algorithme proposé *RBAC-Integrate*(voir la figure 3.2) de la fusion de politiques RBAC, intègre les politiques par l'application de la comparaison et de la fonction de mapping des rôles inter-domaines entre les rôles des domaines collaborants.

L'algorithme *RBAC-Integrate* définit un mapping des rôles inter-domaines en se basant sur les ensembles des permissions assignées et les ordres hiérarchiques des rôles correspondants. L'ensemble de permissions assignées d'un rôle contient les permissions associées directement

à ce rôle et les permissions héritées.

3.3.1.2 La comparaison entre rôles inter-domaines

Deux permissions inter-domaines $p_A : (O_A, a_A)$ et $p_B : (O_B, a_B)$ de domaines respectivement A et B, sont dites équivalentes si les objets inter-domaines O_A et O_B appartiennent à la même classe conceptuelle et que les permissions p_A et p_B sont déclarées partageables dans leur politiques respectives de domaines.

Quatre types de relations peuvent être définis entre deux rôles inter-domaines r_A et r_B :

1. L'equivalence (Equivalent) : Les rôles inter-domaines r_A et r_B sont équivalents($r_A \approx r_B$) si les conditions suivantes sont vérifiées :

a. L'ensemble des permissions Pset(r_A) et Pset(r_B) des rôles r_A, r_B sont équivalents, formellement :

$$\forall i, j : class(O_{A_i}) = class(O_{B_j}) \wedge [(O_{A_i}, a) \in Pset(r_A) \Leftrightarrow (O_{B_j}, a) \in Pset(r_B)]$$

b. Toutes les permissions des ensembles Pset(r_A) et Pset(r_B) sont partageables dans les domaines des rôles r_B et r_A respectivement, formellement :

$$\forall i, j : shareable(O_{A_i}, a, B) \wedge shareable(O_{B_j}, a, A)$$

c. Les rôles r_A et r_B ont la même structure hiérarchique.

$$\begin{cases} \text{Pour tout } r_{Aj} \text{ tel que } r_A \geq_I r_{Aj}, \text{ il existe } r_{Bj} \text{ pour que } r_B \geq_I r_{Bj} \text{ et } r_{Aj} \approx r_{Bj} \\ \text{Pour tout } r_{Aj} \text{ tel que } r_A \geq_A r_{Aj}, \text{ il existe } r_{Bj} \text{ pour que } r_B \geq_A r_{Bj} \text{ et } r_{Aj} \approx r_{Bj}. \end{cases}$$

2. L'inclusion (Contain) : r_A contient r_B ($r_A \supset r_B$) si les conditions suivantes sont vérifiées :

a. L'ensemble des permissions Pset(r_B) du rôle r_B est inclut dans l'ensemble des permissions Pset(r_A) du rôle r_A.

b. Toutes les permissions de l'ensemble Pset(r_B) sont partagées dans le domaine A.

c. Tous les rôles juniors du rôle r_B doivent être aussi des rôles juniors du rôle r_A de la même hiérarchie sémantique.

$$(r_B \geq_I r_C \wedge r_C \neq r_A) \Rightarrow r_A \geq_I^* r_C.$$

3. L'intersection (Overlap) : r_A recouvre r_B (r_A O r_B) si les ensembles Pset(r_A) et Pset(r_B) ont des permissions partageables en commun, mais ni le rôle r_A contient r_B ni r_B contient r_A, formellement :

$$\begin{cases} \exists i,j : class(O_{A_i}) = class(O_{B_j}) \wedge [(O_{A_i}, a) \in Pset(r_A) \wedge (O_{B_j}, a) \in Pset(r_B) \wedge \\ (shareable(O_{A_i}, a, B) \wedge shareable(O_{B_j}, a, A)) \wedge (\neg(r_A \ \ contient \ \ r_B) \wedge \neg(r_B \ \ contient \ \ r_A))] ; \\ \exists r_k, r_m / r_A \geq_I r_k \Rightarrow (r_B \geq_I r_m \wedge r_k \approx r_m). \end{cases}$$

4. La différence (Not related) : le rôle r_A n'est pas lié au rôle r_B ($r_A \neq r_B$) si les rôles r_A et r_B n'ont aucune permission partageable en commun, formellement :

$$\neg\exists i,j : class(O_{A_i}) = class(O_{B_j}) \wedge [(O_{A_i}, a) \in Pset(r_A) \wedge (O_{B_j}, a) \in Pset(r_B)]$$

3.3.1.3 L'algorithme de fusion des politiques

La figure 3.2 montre l'algorithme proposé de la fusion des politiques, *RBAC-Integrate* qui fusionne les politiques RBAC de n domaines pour composer la politique globale.

RBAC-integrate(G_1,G_2,...,G_n)
1. G = {V[G_1], E[G_1]}
2. **pour** i ← à n
3. r1 ← seniormost-role(G)
4. r2 ← seniormost-role(G_i)
5. G ← Role-integrate(r_1, r_2)
6. **pour** chaque r ∈ G
7. **si** (new-role(r) et redundant(r))
8. **alors** Remove-Role(G, r)
9. **retourne**

FIG. 3.2 – L'algorithme *RBAC-integrate*.

Role-integrate(r_1, r_2)
1. **pour** chaque $r_c \in$ Children(r_1)
2. **faire si** ((Pset(r_c)$\bigcap$ Pset(r_2) $\neq \phi$) et Not-compared-previously(r_c,r_2))
3. **alors** Role-integrate(r_c, r_2)
4. **pour** chaque $r_c \in$ Children (r_2)
5. **faire si** ((Pset(r_1)$\bigcap$ Pset(r_c)$\neq \phi$) et Not-compared-previously(r_1, r_c))
6. **alors** Role-integrate (r_1, r_c)
7. /* retourne sans rien faire si r_1 et r_2 sont déjà liés */
8. **si** Already-linked (r_1, r_2)
9. **alors retourne**
10. **si** Eq_role(r_1, r_2)
11. **alors si** la liaison de r_1 et r_2 ne viole pas les propriétés de consistance de RBAC
12. **alors** link (r_1, r_2)
13. **retourne**
14. **sinon si** Contained (r_2, r_1)
15. **alors** r_{1j} = split (r_1, Common-permission(r_1, r_2), Common-juniors-I(r_1, r_2))
16. **si** la liaison de r_{1j} et de r_2 ne viole pas les propriétés de consistance de RBAC
17. **alors** link (r_{1j}, r_2)
18. **retourne**
19. **sinon si** Contained (r_1, r_2)
20. **alors** r_{2j} = split (r_2, Common-permission(r_1, r_2), Common-juniors-I(r_1, r_2))
21. **si** la liaison de r_1 et r_{2j} ne viole pas les propriétés de consistance de RBAC
22. **alors** link (r_{2j}, r_1)
23. **retourne**
24. **sinon si** overlap (r_1, r_2)
25. **alors** r_{1j} = split(r_1, Common-permission(r_1,r_2), Common-juniors-I(r_1,r_2))
26. r_{1j} = split(r_1, Common-permission(r_1,r_2), Common-juniors-I(r_1,r_2))
27. **si** la liaison de r_{1j} et r_{2j} ne viole pas les propriétés de consistance de RBAC
28. **alors** link (r_{1j}, r_{2j})
29. **retourne**
30. **retourne**

FIG. 3.3 – L'algorithme *Role-integrate*.

split(r_1, Common-permissions(r_1, r_2), Common-juniors(r_1, r_2))
1. create-role(r_{1j})
2. ajouter(Childrenlist-I(r_1), r_{1j})
3. **pour** chaque p ∈ Common-permissions(r_1, r_2)
4. **faire** supprimer(Plist(r_1), p)
5. **pour** chaque r_c ∈ Common-juniors(r_1, r_2)
6. **faire** supprimer(Childrenlist-I(r_1), r_c)
7. ajouter(Childrenlist-I(r_{1j}), r_c)
8. **retourne** r_{1j}

link(r_1, r_2)
1. ajouter(Childrenlist-I(r_1), r_2)
2. ajouter(Childrenlist-I(r_2), r_1)
3. **pour** chaque r_i /* (r_i=$r_1 \vee r_i \geq_I{}^* r_1$) */
4. **faire pour** chaque r_j /* ($r_j \in$ Conf-rset(r_1))$\vee(r_j \geq_I^* r_c \wedge r_c \in$ Conf-rset(r_1))*/
5. **faire** Conf-rset(r_i) = Conf-rset(r_i) $\cup$ r_j
6. Conf-rset(r_j) = Conf-rset(r_j) $\bigcup$ r_i
7. **pour** chaque r_i /*($r_i = r_2 \vee r_i \geq_I{}^* r_2$) */
8. **pour** chaque r_j /* ($r_j \in$ Conf-rset(r_2))$\vee(r_j \geq_I^* r_c \wedge r_c \in$ Conf-rset(r_2)) */
9. **faire** Conf-rset(r_i) = Conf-rset(r_i)$\bigcup$ r_j
10. Conf-rset(r_j) = Conf-rset(r_j)$\bigcup$ r_i
11. **retourne**

Remove-Role(r_d)
1. $R_p \leftarrow R_p \cup$ {r}, pour tout r tel que r $\geq_I r_d$
2. $R_c \leftarrow R_c \cup$ {r}, pour tout r tel que $r_d \geq_I$ r
3. **pour** chaque $r_p \in R_p$
4. **pour** chaque $r_c \in R_c$
5. si (∃ r' : r' $\neq r_d \wedge r_p \geq_I$ r' $\wedge$ r' $\geq_I r_c$)
6. continue
7. ajouter (Childrenlist-I (r_p), r_c)
8. supprimer (Parentlist-I (r_c), r_d), ajouter (Parentlist-I (r_c), r_p)
9. **pour** tous r_e : $r_e \in$ equivalent(r_d)
10. supprimer(Parentlist-I (r_e), r_d), supprimer(Childrenlist-I (r_e), r_d)
11. **pour** tous r_s : $r_d \in$ Conf-rset(r_s)
12. supprimer(Conf-rset (r_s), r_d)
13. **pour** chaque $r_p \in R_p$
14. **pour** chaque p ∈ Pset(r_d)
15. ajouter (Pset(r_p), p)
16. désallouer (r_d)

FIG. 3.4 – Les procédures utilisées dans l'algorithme *Role-integrate*.

Prédicat	**Description**
Pset(r)	Retourne l'ensemble de toutes les permissions qui sont assignées directement au rôle r ou sont héritées par le rôle r.
Plist(r)	Retourne l'ensemble de toutes les permissions qui sont assignées directement au rôle r.
$Pset_{assign}$(r)	Retourne l'ensemble des permissions qui sont assignées directement au rôle r.
Class(O)	Retourne la classe conceptuelle de l'objet O.
Conf-rset(r)	Retourne l'ensemble de tous les rôles qui sont en conflits avec le rôle r.
Conf-user(r)	Retourne l'ensemble des utilisateurs qui ne peuvent pas acquérir le rôle r simultanément.
Shareable(O,a,X)	Retourne True si la permission (O, a) peut être partagé dans le domaine X.
Seniormost-role(G)	Retourne le plus senior rôle de graphe RBAC.
Children(r)	Retourne tous les rôles r' tel que r $\geq_I$ r' $\vee$ r $\geq_A$ r'.
Childrenlist-I(r)	Retourne tous les rôles r' tel que r $\geq_I$ r'.
Parentlist-I(r)	Retourne tous les rôles r' tel que r' $\geq_I$ r.
Common-permissions(r_1,r_2)	Retourne l'ensemble de toutes les permissions commun et assignées directement aux rôles r_1 et r_2.
Common-juniors-I(r_1,r_2)	Retourne l'ensemble des rôles R_j. $R_j = \{r : r_1 \geq_I r \; et \; \exists \; r' \; (Eq\_role \; (r, \; r') \; \wedge \; r_2 \; \geq_I \; r')\}$, r_1 et r_2 sont des rôles interdomaines.
Creater-role(r)	Créer un nouveau rôle r.
New-role(r)	Retourne True si r est un nouveau rôle crée pendant la fraction des rôles.
Redundant(r)	Retourne True si r est un rôle redondant.
Not-compared-previously(r_1,r_2)	Retourne True si les rôles interdomaines r_1 et r_2 ne sont pas comparés par l'algorithme Role-integrate.
Already-linked(r_1,r_2)	Retourne True si les rôles r_1 et r_2 sont des rôles interdomaines et $r_1 \geq_I r_2$ et $r_2 \geq_I r_1$.
Eq_ role (r_1,r_2)	Si les conditions suivantes sont vérifiés : $\begin{cases} Pset_{assign}(r_1) = Pset_{assign}(r_2) \wedge ; \\ \forall \; r_{1j} \text{ tel que } r_1 \geq_I r_{1j}, \exists \; r_{2j} \text{ pour que } r_2 \geq_I r_{2j} \text{ et Eq\_ role } (r_{1j}, r_{2j}) \wedge ; \\ \forall \; r_{1j} \text{ tel que } r_1 \geq_A r_{1j}, \exists \; r_{2j} \text{ pour que } r_2 \geq_A r_{2j} \text{ et Eq\_ role}(r_{1j}, r_{2j}). \end{cases}$
contained(r_1,r_2)	(p $\in Pset_{assign}$ (r_1) $\Rightarrow$ p $\in Pset_{assign}(r_2)) \wedge ((r_1 \geq_I r_k \wedge r_k \neq r_2) \Rightarrow r_2 \geq^*_I r_k)$.
Overlap(r_1,r_2)	$(\exists p \in Pset_{assign}(r_1) \Rightarrow p \in Pset_{assign}(r_2)) \vee \; (\exists r_k, r_m / r_1 \; \geq_I \; r_k \; \Rightarrow \; (r_2 \; \geq_I \; r_m \wedge Eq\_role(r_k, r_m)))$.
u-assign(u, r)	Retourne True si l'utilisateur u est assigné au rôle r.
Conf-role(r_1,r_2)	Retourne True si r1 et r2 sont des rôles en conflits

TAB. 3.1 – Les prédicats utilisés pour la composition des politiques.

Le paramètre d'entrée G_i représente le graphe de la politique RBAC du domaine *i*. Cet algorithme combine itérativement les politiques de contrôle d'accès de domaines collaborants en paires. Dans la première itération, la politique d'interopération RBAC est composée de deux domaines en appelant la procédure *role-integrate*(voir la figure 3.3). Dans les itérations suivantes, la politique RBAC d'un autre domaine est combinée avec la politique RBAC obtenue dans l'itération précédente. Après *n-1* itérations, les politiques RBAC de tous les *n* domaines sont fusionnées pour produire une politique RBAC multi-domaines.

L'algorithme *role-integrate*, représente dans la figure 3.3 intègre les rôles inter-domaines en se basant sur leurs ensembles de permissions assignées et leurs ordres hiérarchiques. *role-integrate* est un algorithme récursif qui utilise la stratégie bottom-up (de bas en haut) pour déterminer les rôles équivalents entre deux domaines. L'algorithme attribue à tous les rôles inter-domaines une des quatre relations définies dans le paragraphe (cf.§ 3.3.1.2) :

- Si les rôles ne partagent aucune permission, alors l'algorithme termine l'exécution sans rien faire.

- Si les rôles inter-domaines par exemple r_1 et r_2, sont équivalents selon leurs permissions assignées et leurs ordres hiérarchiques, alors ils sont liés ensemble en définissant un mapping bidirectionnel des rôles inter-domaines entre r_1 et r_2, c-à-d $r_1 \geq_I r_2$ et $r_2 \geq_I r_1$. Le mapping des rôles $r_1 \geq_I r_2$ est représenté dans le graphe RBAC par un arc de «I-hiérarchie» de r_1 à r_2. La liaison de deux rôles inter-domaines r_1 et r_2 par un mapping bidirectionnel, implique qu'un utilisateur u_i assume le rôle r_1, hérite toutes les permissions du rôle r_2. De même, un utilisateur u_j assume le rôle r_2, hérite toutes les permissions du rôle r_1.

  La procedure *Role-integrate* appelle la fonction *link* (montrée dans la figure 3.4) pour établir les mappings bidirectionnels des rôles inter-domaines équivalents r_1 et r_2.
  De plus, les ensembles des rôles en conflit de r_1 et de r_2 et de tous leurs rôles seniors qui ont un chemin d'héritage à r_1 et r_2 et de tous les rôles qui sont en conflit avec r_1 et r_2 et de leurs rôles seniors sont mis à jour dans la fonction *link*.
  Cette mise à jour dans les ensembles des rôles en conflit est essentielle pour préserver la cohérence de la propriété hiérarchique du modèle RBAC qui exige que l'ensemble des rôles en conflit d'un rôle junior doit être contenu dans les ensembles des rôles en conflit de ses rôles seniors.
  En raison de cette mise à jour dans les ensembles des rôles en conflit, des nouvelles contraintes SoD sont ajoutées entre des rôles qui ne sont pas en conflit dans leur politique RBAC du domaine original. Ce type de contraintes SoD est appelé contrainte SoD induite. Puisque il y a plusieurs types d'hiérarchie, l'ajout des rôles dans les ensembles des rôles en conflit peut mener à une situation dans laquelle deux rôles en conflit ont le

même rôle ancêtre qui hérite ces deux rôles. Cette situation peut être évitée par l'ajout de la condition suivante : deux rôles r_2 et r_3 peuvent être mis en conflit si et seulement s'ils n'ont pas un rôle ancêtre commun qui hérite les deux.

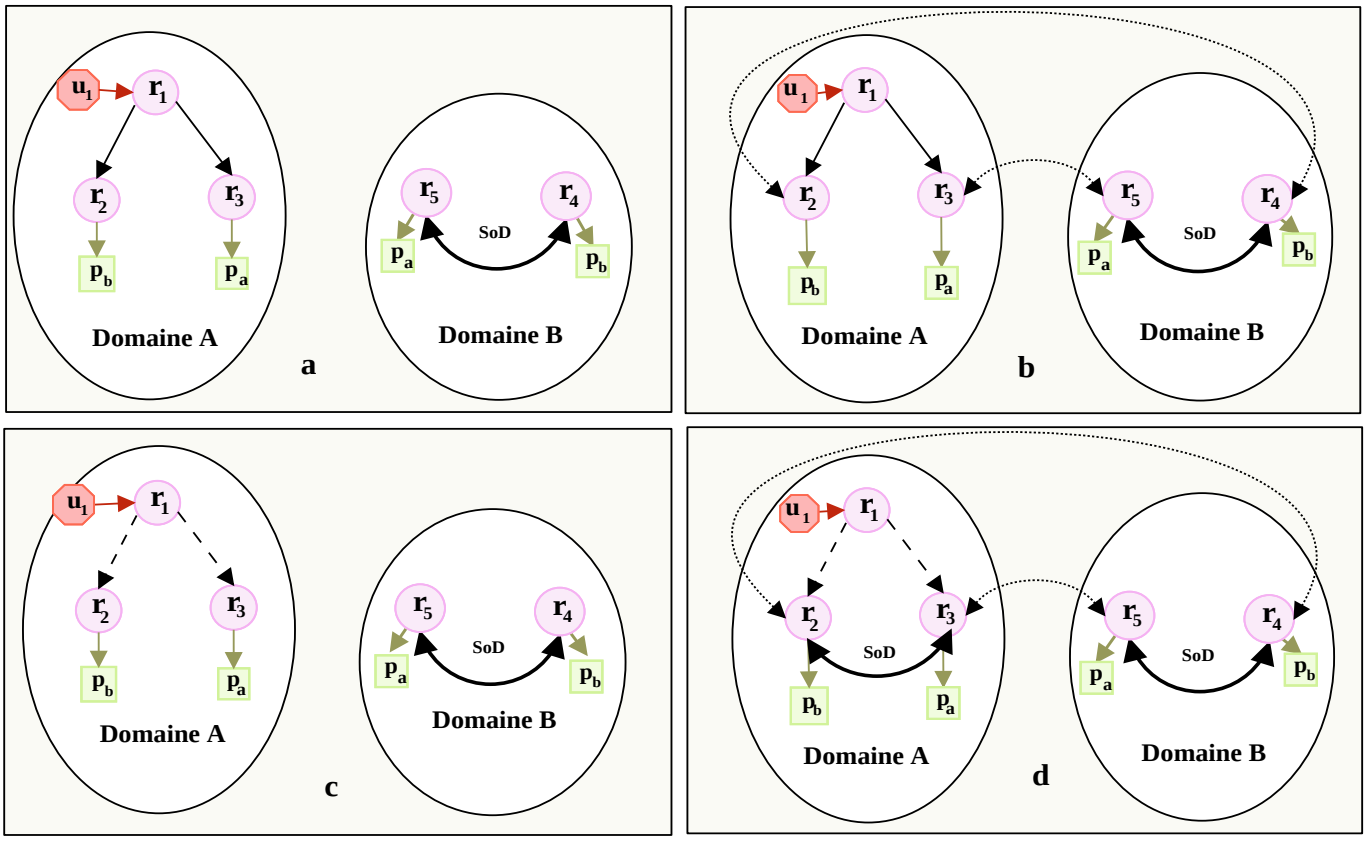


FIG. 3.5 – Exemple de SoD induite .

Par exemple, dans la figure 3.5(d), qui représente la politique d'interopérabilité de politiques RBAC des domaines A et B notées dans la figure 3.5(c), une contrainte SOD induite est définie entre les rôles r_2 et r_3 quand ces rôles sont liés, respectivement aux deux rôles inter-domaines en conflit r_4 et r_5. Les rôles r_2 et r_3 ont un rôle senior commun r_1 ; cependant, ce rôle r_1 n'hérite pas les permissions des rôles r_2 et r_3 comme il est lie à r_2 et à r_3 par la hiérarchie d'activation. Au contraire, dans la figure 3.5(b), qui représente la politique d'interopérabilité de politiques RBAC des domaines A et B notées dans la figure 3.5(a), une contrainte SoD n'est pas ajoutée aux rôles r_2 et r_3, malgré que ces rôles sont liés aux deux rôles inter-domaines contradictoires r_4 et r_5, à cause du rôle r_1 qui est un rôle ancêtre commun de r_2 et r_3 dans la sémantique d'hiérarchie d'héritage (le rôle r_1 hérite les deux rôles r_2 et r_3). La politique d'interopérabilité montrée dans la figure 3.5(b) n'est pas cohérente, et un des mappings des rôles inter-domaines ($r_2 \geq_I r_4$) ou ($r_3 \geq_I r_5$) doit être supprimé.

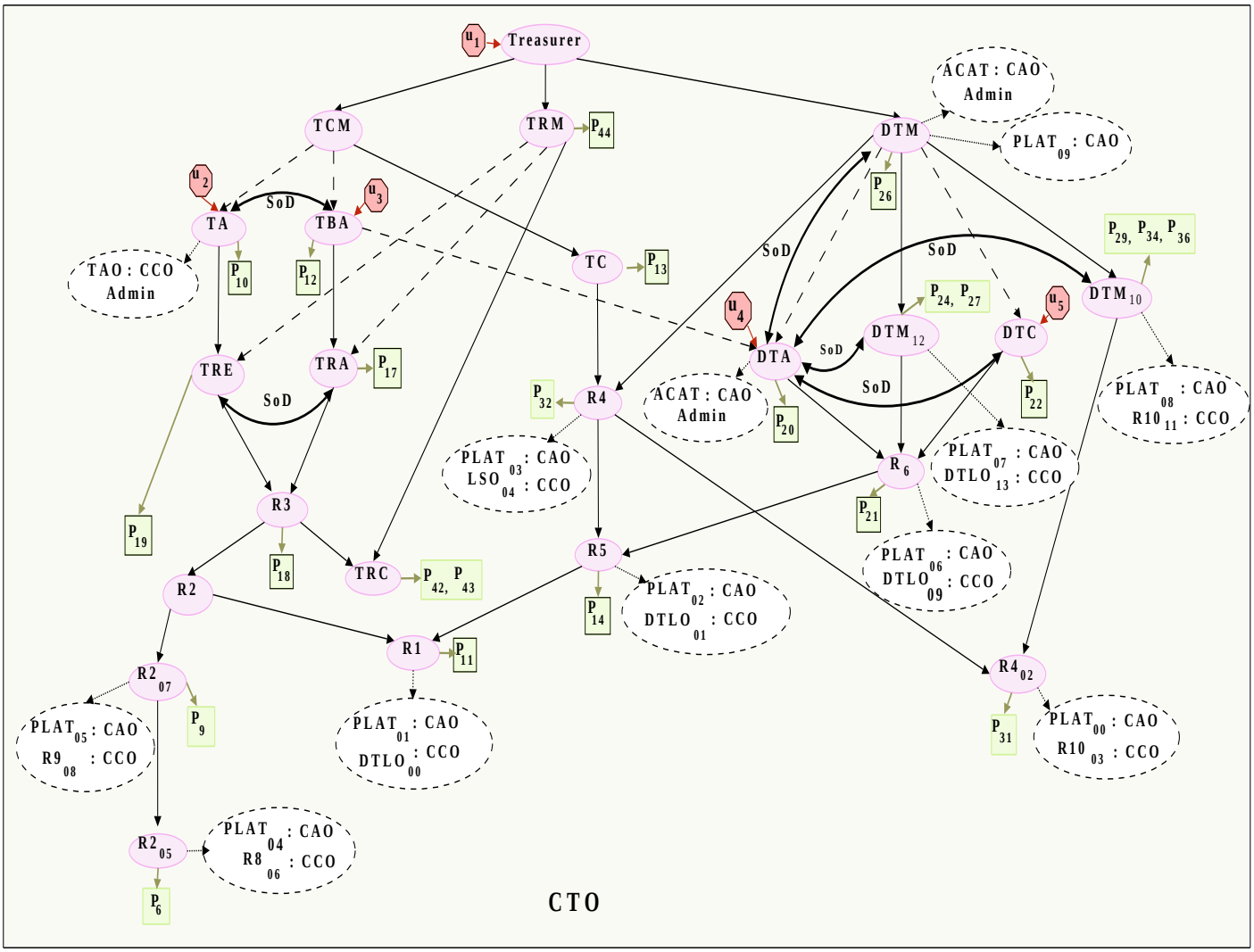
- Le rôle r_1 est contenu dans r_2 si l'ensemble de toutes les permissions directement assignées à r_1 est contenu dans l'ensemble des permissions directement assignées à r_2, et si

tous les rôles juniors de r_1 dans la sémantique I-hiérarchie sont aussi des rôles juniors de r_2 dans la sémantique I-hiérarchie. Si r_2 contient r_1, alors un nouveau rôle junior r_{2j} de r_2 est créé en appelant la fonction *split*, montrée dans l'algorithme 3.4. Dans la fonction *split*, toutes les permissions et les rôles juniors en commun (en sémantique I-hiérarchie) à r_1 et à r_2 sont enlevées de r_2 et sont assignées à r_{2j}, après les rôles r_1 et r_{2j} sont liés par un mapping bidirectionnel.

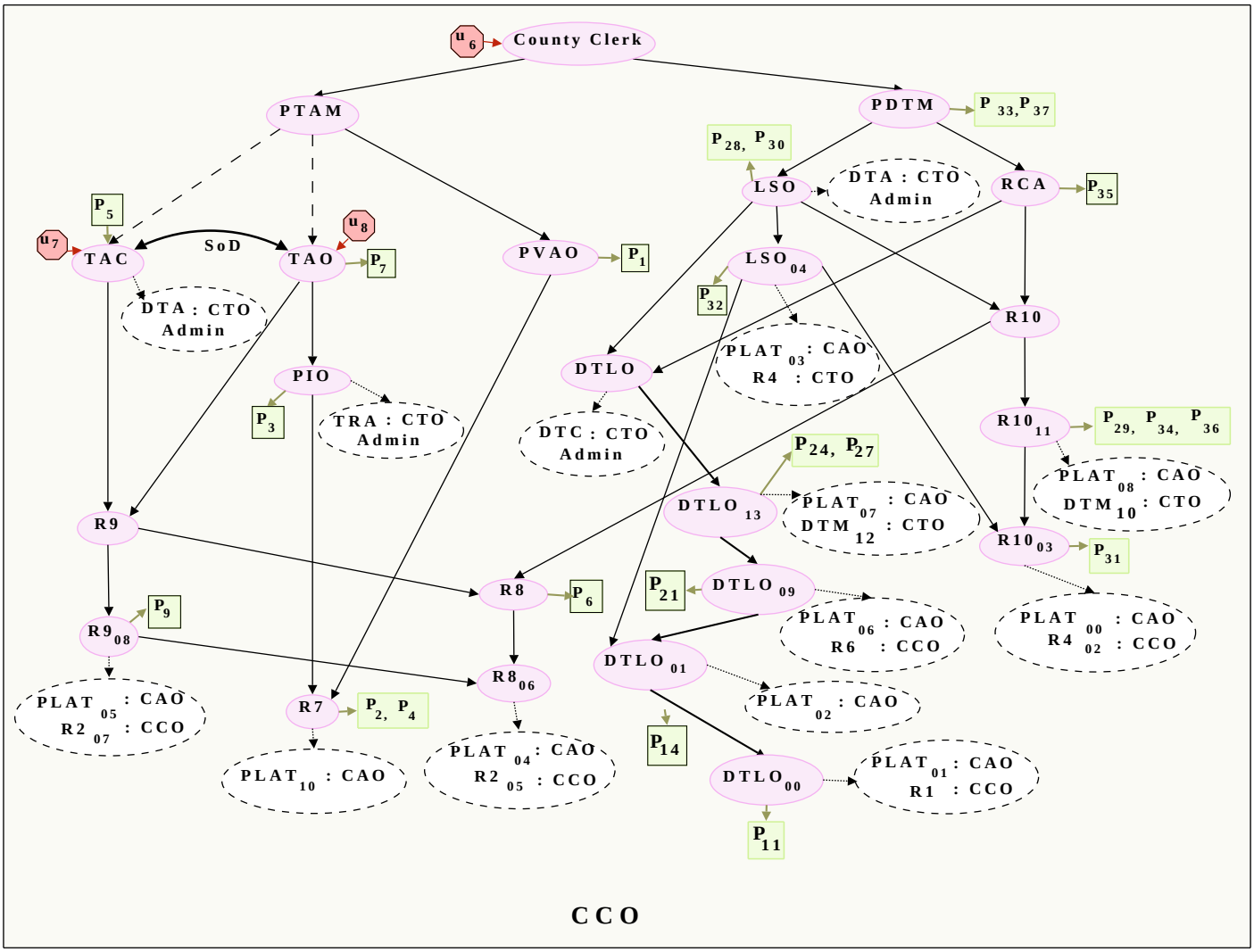
- Si la relation entre les rôles r_1 et r_2 est «l'intersection» mais aucun rôle ne contient l'autre, donc deux nouveaux rôles junior r_{1j} et r_{2j} sont créés, respectivement de r_1 et r_2. Les permissions et les rôles juniors en communs à r_1 et r_2 sont enlevés des rôles senior r_1 et r_2 et sont assignés aux rôles juniors r_{1j} et r_{2j}. Après, un mapping bidirectionnel($r_{1j} \geq_I r_{2j}$ et $r_{2j} \geq_I r_{1j}$) est défini entre r_{1j} et r_{2j}.

3.3.2 Exemple

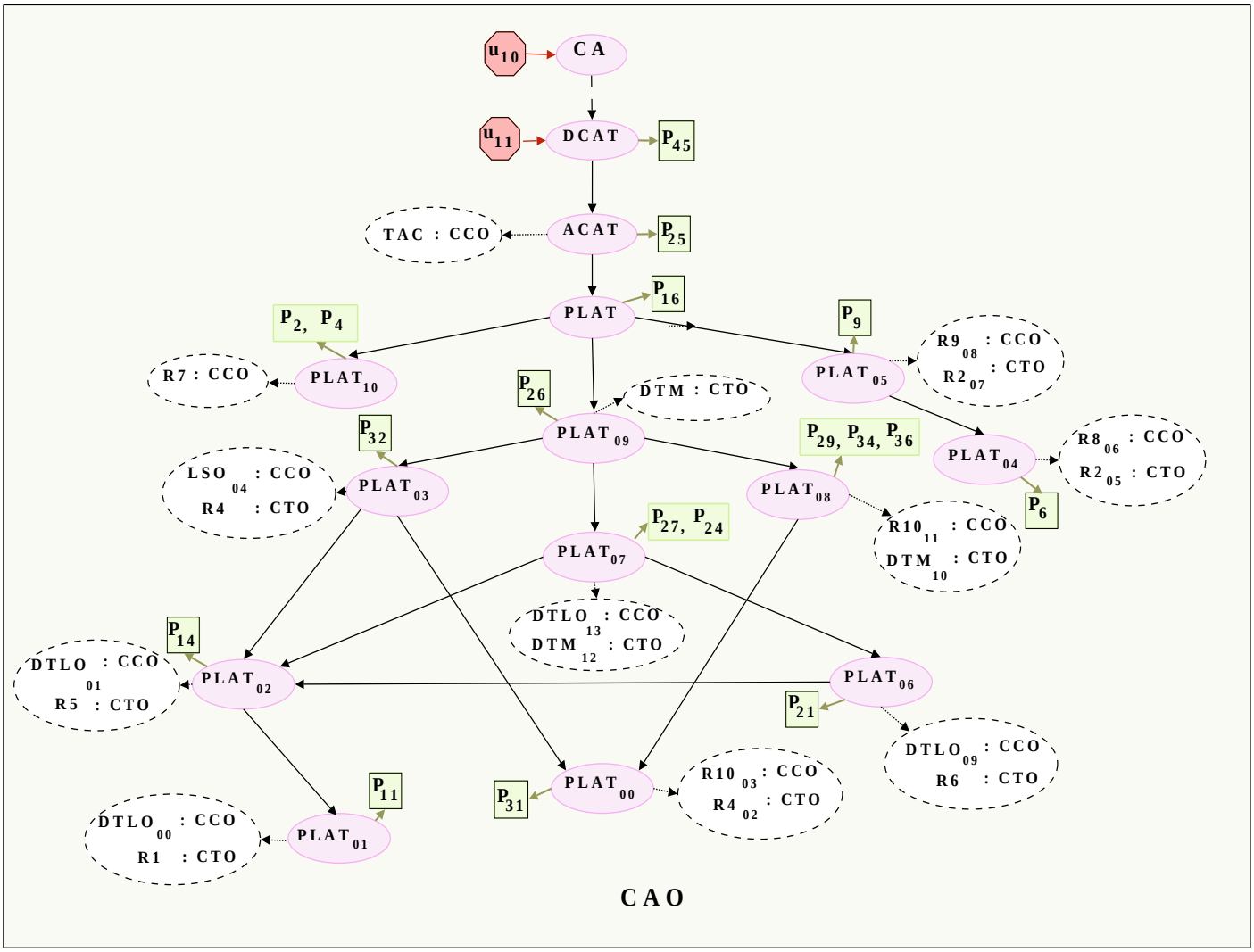
Prenons l'exemple présenté dans la section 2.4 de trois domaines : *CTO* (County Treasure Office), *CCO* (County Clerk Office), et *CAO* (County Attorney Office). Les graphes des politiques RBAC de cet exemple sont montrés dans la figure 2.1. L'intégration des politiques RBAC de contrôle d'accès de ces domaines par l'algorithme *RBAC-Integrate*, donne la politique RBAC multi-domaines notée dans la figure 3.6 [3].


u1
Treasurer
TCM
TRM
P44
ACAT : CAO
Admin
DTM
PLAT09 : CAO
u2
SoD
u3
TA
TBA
P26
P29, P34, P36
TAO : CCO
Admin
P10
P12
TC
P13
SoD
SoD
u4
P24, P27
u5
DTM10
DTM12
DTC
DTA
SoD
TRE
TRA
P17
SoD
SoD
P32
R4
ACAT : CAO
Admin
P20
P22
PLAT08 : CAO
R1011 : CCO
PLAT03 : CAO
LSO04 : CCO
R6
PLAT07 : CAO
DTLO13 : CCO
R3
P19
P21
P18
R5
PLAT06 : CAO
DTLO09 : CCO
TRC
P42, P43
R2
PLAT02 : CAO
DTLO01 : CCO
P14
R1
P11
R202
R207
PLAT05 : CAO
R908 : CCO
P9
PLAT01 : CAO
DTLO00 : CCO
P31
PLAT00 : CAO
R1003 : CCO
R205
PLAT04 : CAO
R806 : CCO
P6
CTO

(a)


u6
County Clerk
PTAM
PDTM
P33, P37
P28, P30
LSO
DTA : CTO
Admin
RCA
P35
P5
u7
TAC
SoD
u8
TAO
P7
PVAO
P1
LSO04
P32
R10
DTLO
PLAT03 : CAO
R4 : CTO
PIO
P3
TRA : CTO
Admin
DTC : CTO
Admin
R1011
P29, P34, P36
P24, P27
PLAT08 : CAO
DTM10 : CTO
DTLO13
PLAT07 : CAO
DTM12 : CTO
R1003
P31
R9
R8
P6
P21
DTLO09
R908
P9
PLAT06 : CAO
R6 : CCO
PLAT00 : CAO
R402 : CCO
DTLO01
R806
PLAT05 : CAO
R207 : CCO
R7
P2, P4
PLAT02 : CAO
P14
PLAT04 : CAO
R205 : CCO
DTLO00
PLAT01 : CAO
R1 : CCO
PLAT10 : CAO
P11
CCO

(b)



(c)

FIG. 3.6 – (a) La politique RBAC du domaine *CTO*, (b) du domaine*CCO*, et (c)du domaine*CAO* après la composition [3].

Observons que certains rôles dans la figure 3.6 de la politique RBAC multi-domaines sont divisés en deux rôles ou plus et que leurs permissions sont redistribuées entre leurs nouveaux rôles juniors crées. Par exemple le rôle *DTM* du domaine *CTO*(voir la figure 2.1(a)), après la composition des politiques RBAC il est divisé en trois rôles : un rôle senior *DTM* et deux rôle junior DTM_{10}, DTM_{12} et Leurs permissions aussi sont redistribuées entre ces rôles (voir la figure 3.6(a)). Ainsi, le rôle *PLAT* du domaine *CAO*(voir la figure 2.1(c)), qui est divisé en douze rôles (voir la figure 3.6(c)).

Les nœud-rôle dans la figure 3.6 sont représentés par deux ellipses différents :

- Les ellipse à ligne continue.
- Les ellipse à ligne discontinue.

Les ellipses à ligne continue représentent les rôles locaux des domaines. Les ellipses à ligne discontinue représentent les rôles inter-domaines, les annotations dans ces ellipses représentent le nom du rôle étranger et le nom du son domaine. Les mappings des rôles inter-domaines sont représentés dans chaque domaine par des arcs des ellipses des rôles locaux vers les ellipses des rôles inter-domaines, par exemple dans le domaine *CTO* l'ellipse avec annotation *PLAT09* :*CAO* représente un rôle étranger *PLAT09* du domaine *CAO*, et l'arc entre le rôle local *DTM* et cet ellipse représente le mapping des rôles inter-domaines entre les rôles inter-domaines *DTM* et $PLAT_{09}$. Les mappings des rôles inter-domaines introduits par l'administrateur sont spécifiés par le mot *Admin*, exemple : l'arc entre le rôle *DTM* du domaine *CTO* et l'ellipse du rôle étranger *ACAT* représente un mapping des rôles inter-domaines introduit par l'administrateur.

3.4 Conclusion

Dans ce chapitre, nous avons détaillé la seule méthode de composition des politiques RBAC proposé par BASIT. Nous avons vu que l'interopérabilité dans le contexte des politiques de contrôle d'accès RBAC consiste à créer des mappings des rôles inter-domaines. En plus de la génération automatisée des mappings des rôles inter-domaines des domaines collaborant, la structure permet aussi aux administrateurs des politiques de sécurité de définir des mappings des rôles inter-domaines selon certains exigences d'interopérabilité des domaines collaborants.

La question qui se pose maintenant : est ce que cette méthode de composition génère une interopérabilité cohérente ? et si c'est non, quels sont les conflits qui peuvent être émergés ? et comment ces conflits peuvent être résolus ? Tous ça est détaillé dans le chapitre suivant .

Chapitre 4

L'optimisation de composition des politiques RBAC

4.1 Introduction

Considérons le système d'information d'une organisation de recherches, où Alice étant une surveillante du projet, est permise d'accéder aux dossiers de Bob, mais pas vice versa. Supposons que cette organisation est juste devenue une filiale d'une société, où Charles est vice-président pour la recherche, et Diana étant sa secrétaire, a l'accès à ses dossiers. Après la fusion, il semble normal de permettre à Charles d'accéder aux dossiers du projet d'Alice. Mais si Bob est permis d'accéder aux dossiers de Diana, il y aurait une violation de sécurité parce que maintenant Bob aurait potentiellement accès (indirectement par l'intermédiaire de Diana et de Charles) aux dossiers d'Alice auxquels il devrait être interdit d'accès.

Bien que la violation de sécurité dans cet exemple ne soit pas trop difficile pour la découvrir et l'enlever, un véritable environnement multi-domaines pourrait avoir des centaines ou des milliers d'accès inter-domaines aux informations et ressources des domaines collaborants. En d'autres termes, l'interopérabilité des systèmes hétérogènes avec des structures de contrôle d'accès pose de nouveaux défis ; la violation des contraintes de sécurité et d'autonomie des domaines collaborant.

Dans ce chapitre, nous citons les différents types de conflits peuvent apparaître dans la politique RBAC d'interopérabilité, et la méthode de résolution des conflits existée et nous expliquons comment on peut formuler le problème de résolution de conflits en un problème d'optimisation, en se basant sur certains critères d'optimisation.

4.2 Les conflits dans l'interopérabilité des politiques RBAC

La politique RBAC d'interopération obtenue, peut ne pas être cohérente et ne pas satisfaire les contraintes de sécurité de domaines collaborant. De plus, les mappings des rôles inter-domaines que l'administrateur introduisent, peuvent être en conflit avec les politiques de contrôle d'accès des domaines collaborant.

Particulièrement dans la politique de composition des politiques de contrôle d'accès RBAC, trois types de violations de sécurité peuvent survenir à cause des mappings incompatibles des rôles inter-domaines :

4.2.1 La violation des contraintes d'assignation des rôles

La politique globale cause une violation des contraintes d'assignation des rôles du domaine k, si elle permet à un utilisateur u_k du domaine k d'accéder à un rôle local r_k, bien que u_k n'était pas directement assigné à r_k ou à aucun de ses rôles seniors du domaine k.

4.2.2 La violation des contraintes SoD spécifiques des rôles

La politique globale cause une violation des contraintes SoD spécifiques des rôles du domaine k, si elle permet à un utilisateur d'accéder aux deux rôles contradictoires simultanément r_i et r_j du domaine k à la même session ou aux sessions concourantes.

4.2.3 La violation des contraintes SoD spécifiques des utilisateurs

Soit ${U^c}_r$ l'ensemble des utilisateurs qui sont en conflit pour le rôle r_K appartenant au domaine k. La politique globale cause une violation des contraintes SoD spécifiques des utilisateurs du domaine k si elle permet aux deux utilisateurs distincts de l'ensemble ${U^c}_r$ d'accéder au rôle r_K à la même session ou aux sessions concourantes.

Un mécanisme de résolution de conflit, est nécessaire pour résoudre les conflits entre les domaines collaborant, d'une manière optimale.

4.3 Détection et résolution des conflits

La politique d'interopérabilité des domaines CTO, CCO et CAO montrée dans la figure 3.6 est incohérente et elle ne satisfait pas les exigences de sécurité des domaines collaborants. Plusieurs conflits des mappings des rôles inter-domaines sont apparus, par exemple : les mappings spécifiés par l'administrateurs : $TA : CTO \geq_I TAO : CCO$ et $PIO : CCO \geq_I TRA : CTO$ causent une violation de contrainte SoD spécifiques des rôles, définie entre les rôles TRE et TRA du domaine CTO, ces mappings permettent à l'utilisateur u_2 assigné au rôle TA, d'accéder au rôle TRA par l'intermédiaire du chemin inter-domaine $TA : CTO \geq_I TAO : CCO \geq_I PIO : CCO \geq_I TRA : CTO$, aussi les mappings des rôles inter-domaines $DTA : CTO \geq ACAT : CAO$, et $PLAT09 : CAO \geq DTM : CTO$ cause une violation des contraintes d'assignation des rôles, ils permettent au utilisateur u_4 assigné au rôle junior DTA, d'accéder au rôle senior DTM, par le cycle d'héritage : $DTA : CTO \geq ACAT : CAO, ACAT : CAO \geq PLAT_{09} : CAO$.

Ces mappings sont en conflit et la solution est d'enlever certains mappings de l'ensemble des conflits. Par exemple pour résoudre le conflit qui provoque la violation des contraintes SoD des rôles TRE et TRA un des mappings suivants doit être supprimé ; soit on supprime le mapping $TA : CTO \geq_I TAO : CCO$ ou le mapping $PIO : CCO \geq_I TRA : CTO$.

La question qui se pose maintenant : quels sont les mappings des rôles inter-domaines de l'ensemble qui sont en conflits qui doivent être enlevés, pour que les contraintes de sécurité et de l'autonomie des domaines collaborants ne soient pas violés ? Aussi, l'enlèvement arbitrairement des mappings des rôles inter-domaines réduit l'interopérabilité.

L'objectif de la phase de résolution des conflits, est de produire une politique d'interopération qui maximise les accès inter-domaines sûrs et garde les pertes d'autonomie des domaines collaborants dans des limites acceptables.

Le problème de résolution des conflits dans une politique RBAC multi-domaine, peut être formulé comme un problème d'optimisation avec l'objectif de maximiser les accès inter-domaines permis selon certains critères d'optimalité. La notion d'optimalité est nouvel, analysé en termes de compromis entre la perte d'autonomie et le degré d'interopération [3].

Définition : L'interopérabilité sûre des politiques RBAC de contrôle d'accès consiste des politiques RBAC individuelles sûres des domaines collaborants et un ensemble F des mappings additionnels et légaux des rôles inter-domaines .

4.3.1 Les critères d'optimisation

Il y a quatre types de critères d'optimisation [10, 3] :

4.3.1.1 Maximiser le partage direct des informations

Maximiser le partage direct des informations implique de chercher à maximiser le nombre des mappings des rôles inter-domaines.
Par exemple, on prend la politique d'interopérabilité de deux domaines A et B, notée dans la figure 4.1 :

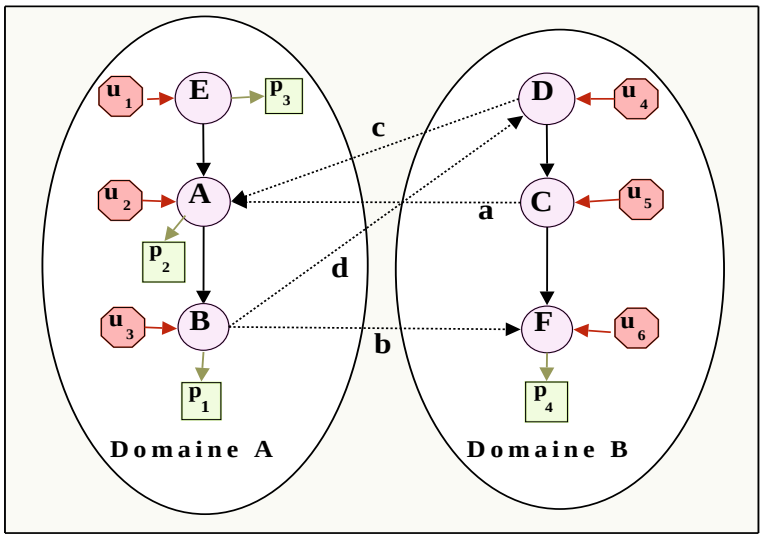


FIG. 4.1 – La politique d'interopérabilité incohérente de deux domaines A et B.

cette politique RBAC d'interopération cause une violation de sécurité ; le rôle B hérite son rôle senior A par l'intermédiaire des mappings des rôles inter-domaines d et c ou d et a. Pour résoudre ce conflit quelques mappings des rôles inter-domaines doivent être supprimés, soit on supprime les mappings des rôles inter-domaines a et c ou le mapping des rôles inter-domaines d.

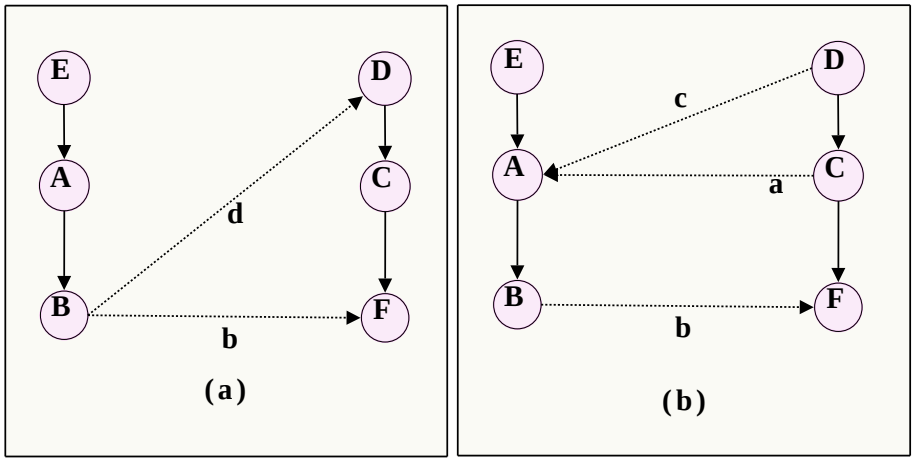


FIG. 4.2 – (a)La politique d'interopération après la suppression des mappings des rôles inter-domaines c et a, (b)La politique d'interoperation après la suppression du mapping des rôles inter-domaines d .

N.B : Pour expliquer l'exemple précédent, on utilise la notation suivante :

- $B \longrightarrow D$ signifie que le rôle B hérite les permissions du rôle D.
- (B, D) représente un mapping des rôles inter-domaines unidirectionnel du rôle B au rôle D.

Si on supprime le mapping des rôles inter-domaines d la taille maximale du F est $|F| = 3$, et $F = \{(B, F), (C, A) , (D, A) \}$. Si on supprime les mappings des rôles inter-domaines a et c la taille maximale du F est $|F| = 2$, et $F = \{(B, F), (B, D)\}$.

Alors pour satisfaire le critère, il est mieux de supprimer le mapping des rôles inter-domaines d et de préserver les mappings des rôles inter-domaines c et a.

4.3.1.2 Maximiser le partage direct et indirect des informations

Une autre mesure naturelle d'optimalité est de maximiser le partage direct et indirect des informations. Le but est de trouver une interopération sûre avec un nombre maximal des accès inter-domaines légaux, au lieu de chercher une solution sûre avec une taille maximale du F.

Prenons la politique d'interopération représentée dans la figure 4.1, où les mappings des rôles inter-domaines a et d (ou c et d) cause une violation de sécurité. Précédemment, pour une solution avec une taille maximale de F, il était meilleur d'enlever le mapping des rôles inter-domaines d pour que les mappings des rôles inter-domaines a et c puissent être préservés. Maintenant pour obtenir l'accès maximal, il est en réalité meilleur, d'enlever les mappings des

rôles inter-domaines a que c et de préserver le mapping des rôles inter-domaines d, parce que ce dernier facilite plus le partage indirect des informations.

- Si on supprime les mappings des rôles inter-domaines c et a, le nombre des accès permis est 9 (voir la figure 4.2(a)) :
  L'ensemble des accès légaux est $\{B \longrightarrow F,\ B \longrightarrow D,\ B \longrightarrow C,\ A \longrightarrow C,\ A \longrightarrow D,\ A \longrightarrow F,\ E \longrightarrow F,\ E \longrightarrow D,\ E \longrightarrow C\}$
- Si on supprime le mapping des rôles inter-domaines d, le nombre des accès permis est 7 (voir la figure 4.2(b)) :
  L'ensemble des accès légaux est $\{B \longrightarrow F,\ A \longrightarrow F,\ E \longrightarrow F,\ C \longrightarrow A,\ C \longrightarrow B,\ D \longrightarrow A,\ D \longrightarrow B\}$

4.3.1.3 La représentation minimale

Quelque soit le nombre entier positif $k \leq |F|$, est-ce qu'il existe un sous-ensemble $F' \subset F$ tel que $|F'| \leq k$ et que l'ensemble des accès légaux reste inchangé quand F est remplacé par F' ?

Par exemple, la politique d'interopérabilité montrée dans la figure 4.3 est cohérente et elle satisfait toutes les contraintes de sécurité.

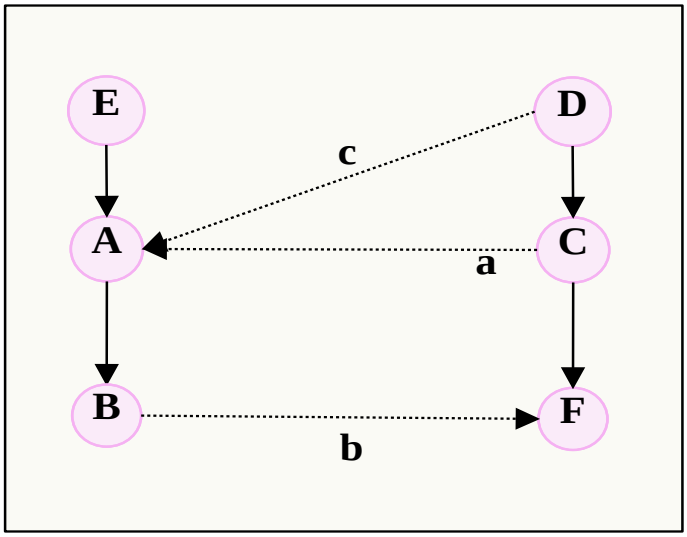


FIG. 4.3 – La politique d'interopérabilité cohérente de deux domaines A et B.

L'ensemble F de cette politique d'interopérabilité est $F = \{ (B, F), (C, A), (D, A)\}$,
et l'ensemble des accès légaux est $\{ B \longrightarrow F, A \longrightarrow F, E \longrightarrow F, C \longrightarrow A, C \longrightarrow B, D \longrightarrow A, D \longrightarrow B\}$.

Si on supprime le mapping des rôles inter-domaines c, l'ensemble des accès légaux reste le

même, donc on peut changer F par $F' = \{ (B, F), (C, A) \}$.

4.3.1.4 Le poids pondéré

Dans certains cas, certains accès inter-domaines ont une priorité plus élevée que les autres. Par conséquent, de tels accès doivent être assignés à un poids plus élevé que des autres pour augmenter leurs chances de préservation dans la politique finale. Le poids d'un accès inter-domaines est défini relativement par rapport aux poids des accès contradictoires qui peuvent être enlevés, en faveur l'accès donné.

4.3.2 La détection et la résolution des conflits

La résolution des conflits d'interopération manuellement est un processus lent, ennuyeux et ad hoc (qui est destiné à répondre à un besoin ponctuel) et ne fournit aucune garantie sur l'optimalité du système résultant d'interopération .

Gong et Al [10] ont examiné l'interopération des systèmes employant la politique de contrôle d'accès multi-niveaux. Ils ont proposé plusieurs techniques d'optimisation pour la résolution de conflits d'interopération. Cependant, ces techniques de résolution sont spécifiques aux conflits d'héritage cyclique et ne considèrent pas d'autres types de conflits d'interopération.

Gavrilla et Al [9] ont défini un jeu de conditions nécessaires et suffisantes pour la composition d'une politique RBAC cohérente. Le critère pour la composition de politique cohérente est défini en termes de cardinalité, de la hiérarchie et des contraintes SoD. En conséquence, une politique d'interopération cohérente peut être composée en vérifiant avec incrémentation de la cohérence des rôles inter-domaines.

La méthodologie de composition de politique proposée par BASIT fournit la seule structure de la détection automatisée et la résolution optimale des conflits d'interopération liés à la politique RBAC multi-domaines. Ces conflits incluent l'héritage cyclique et les violations de contraintes SoD.

4.3.3 L'approche d'optimisation proposée par Basit [3]

L'idée est de formuler le problème d'intégration des politiques de l'environnement multi-domaines en un programme linéaire en nombre entier(IP pour Integer Programming).

Un programme linéaire en nombre entier (IP) est constitué de fonction objective et des contraintes linéaires et des variables astreintes à des valeurs entières. Si certaines variables du modèle sont continues et d'autres en nombres entiers, on parle de programmation mixte entière (MIP pour Mixed Integer Programming). Enfin, un cas particulier très fréquent de programme en nombres entiers est celui où les variables ne peuvent prendre que les valeurs 0 ou 1. Un programme IP est de la forme suivante [30] :

$$\begin{cases} (Maximise) Minimise : C^T x \\ Subject\ to : Ax \leq b \\ x_i : entier. \forall x_i \in x \end{cases}$$

- Typiquement, $x_i = 0$ ou x_i =1.
- A est la matrice de contraintes.
- C est un vecteur définissant les critères d'optimalité en termes du poids des variables de décision correspondant aux autorisations des utilisateurs aux rôles.

4.3.3.1 La Formulation IP de la Politique RBAC multi-domaines

Dans la formulation IP de la politique RBAC, toutes les contraintes comme l'assignation des rôles, les contraintes SoD et les contraintes d'accès permises et interdites sont définies en utilisant des inégalités linéaires. Les variables utilisées dans ces inégalités transmettent des informations sur la possibilité d'accès des utilisateurs aux rôles. Par exemple, les variables ont la forme u_{ir_j} où le premier indice i identifie l'utilisateur et le deuxième indice r_j spécifie le rôle. La formulation IP fonctionne pour les critères d'optimisation suivantes : le partage maximum de données et le poids pondéré. Le changement de mesures d'optimisation dans la formulation exige seulement le changement des poids dans la fonction objective.

- La variable u_{ir_j} est une variable binaire, c'est-à-dire, il peut prendre " 0" ou "1" .
- Si la variable $u_{ir_j} = 1$, donc l'utilisateur u_i est autorisé d'accéder au rôle r_j, sinon u_i n'est pas autorisée de l'accéder par aucun moyen.
- Si l'utilisateur u_i et le rôle r_j sont de domaines différents et $u_{ir_j} = 0$, donc dans le graphe de RBAC, il ne devrait pas y avoir un chemin du nœud d'utilisateur u_i au nœud du rôle r_j.

La politique globale RBAC composée de l'environnement multi-domaines peut être inconséquente, et un chemin peut exister entre un utilisateur u_i d'un domaine et un rôle r_j d'un autre domaine, et dans la solution du problème IP $u_{ir_j} = 0$. Cette inconséquence est résolue en supprimant le mapping des rôles inter-domaines qui se trouve entre le nœud d'utilisateur u_i et le nœud du rôle r_j.

4.3.3.2 Les règles de transformation des contraintes

Ces règles de transformation des contraintes pour générer les équations IP des contraintes de la politique RBAC d'interopération.

- L'ensemble des utilisateurs du domaine k est noté par U_k .
- L'ensemble des rôles du domaine k est noté par R_k.
- L'union de tous les ensembles U_k est noté par U.
- L'union de tous les ensembles R_k est noté par R.

1. Pour chaque domaine k, si un utilisateur $u_i \in U_k$ n'est pas autorisé au rôle $r_j \in R_k$ par la politique de contrôle d'accès du domaine k, alors u_{ir_j}=0.

2. Pour un utilisateur $u_i \in U$ et un rôle $r_j \in R$, si domaine(u_i) $\neq$ domaine(r_j) et u_i ne peut pas hériter les permissions du rôle r_j, alors u_{ir_j}=0.

3. Soit A_u l'ensemble des utilisateurs assignés au rôle r_j . Au moins un utilisateur de l'ensemble A_u doit être capable d'accéder au rôle r_j. Formellement :

$$\sum_{u_i \in\ A_u} u_{ir_j}$$

4. Soit u_{ir_j}=1 et il existe un rôle r_k tel que domaine (r_j) = domaine (r_k) et $r_j \geq_I r_k$, alors u_i est aussi autorisé d'accéder au rôle r_k, c'est-à-dire, u_{ir_k} =1.

5. Soient un utilisateur u_i et un rôle r_k tel que domaine (u_i) $\neq$ domaine (r_k). Et R_m l'ensemble de rôles tel que pour tout $r_m \in R_m$, domaine (r_m) = domaine (r_k). En plus dans le graphe RBAC, il y a un chemin de u_i à r_m et $r_m \geq_I r_k$. Soient deux autres ensembles des rôles R_c et R_{pc} sont définis comme suit :

   R_c =$\{r | r \geq_I r_k \wedge domaine(r_k) \neq domaine(r)\}$.
   R_{pc}=$\{r_p | r \in R_c\ telque (r_p = r \wedge u\_assign(u,r)) \vee (r_p \geq_I r \wedge domaine(r) = domaine(r_p))\}$.

   Les inégalités suivantes déterminent les conditions pour qu'un utilisateur u_i puisse accéder au rôle r_k :

   a. $\forall\ r_m \in R_m, u_{ir_m} - u_{ir_k} \leq 0$

   b.
   $$\sum_{r_m \in R_m} u_{ir_m} + \sum_{r_n \in R_c} u_{ir_n} - u_{ir_k} \geq 0$$

   c.
   $$\sum_{r_m \in R_m} u_{ir_m} + \sum_{r_p \in R_{pc}} u_{ir_p} - u_{ir_k} \geq 0$$

Les contraintes ci-dessus impliquent qu'un utilisateur u_i peut accéder au rôle r_k seulement si une des deux conditions suivantes est vérifiée :

1. u_i est autorisé pour un rôle inter-domaines r_m tel que domaine (r_m)= domaine (r_k) et $r_m \geq_I r_k$.
2. u_i est autorisé pour un rôle r_n et il y a un mapping des rôles inter-domaines de r_n à r_k.

La condition (5c) est nécessaire pour éviter n'importe quelle attribution localisée de 1 aux variables u_{ir_k} et u_{ir_n}, où $r_n \in R_c$.

6. Quelque soient deux utilisateurs u_i , u_j et un rôle r_k, supposons que u_i est autorisé d'accéder au rôle r_k, c'est-à-dire $u_{ir_k} = 1$, et qu'un mapping des rôles inter-domaines existe du rôle r_k au rôle r_l. Si l'utilisateur u_i est capable d'accéder à r_l par le chemin du mapping des rôles inter-domaines (r_k ; r_l), alors l'utilisateur u_j, s'il est autorisé d'accéder au rôle r_k, il peut aussi accéder à r_l par le chemin du mapping des rôles inter-domaines (r_k ; r_l). Formellement :
   si domaine(u_i) = domaine(u_j) = domaine(r_k) alors
   (u_{ir_k} - u_{ir_l})-(u_{jr_k} - u_{jr_l}) = 0 ;
   sinon
   (u_{ir_k} - u_{ir_l})-(u_{jr_k} - u_{jr_l}) $\geq$0.

7. une contrainte SoD spécifique des rôles peut exister entre deux rôles intra-domaines ou inter-domaines. Dans le graphe RBAC, la contrainte SoD entre deux rôles contradictoires r_j et r_k est représentée par une flèche à deux points entre les rôles r_j et r_k. Dans la formulation IP, cette contrainte SoD peut être écrite comme suivant : $u_{ir_j} + u_{ir_k} \leq 1$ pour tous les utilisateurs u_i tel que u_i peut accéder à r_j ou à r_k.

8. Supposons qu'une contrainte SoD existe entre deux rôles intra-domaines r_m et r_n est causée par les rôles inter-domaines $r_k$et r_l. Cette contrainte SoD induite peut être écrite en forme d'équation comme suivant : $u_{ir_m} + u_{ir_n} + u_{ir_k} + u_{ir_l} \leq 3$ pour tous les utilisateurs u_i tel que u_i peut accéder à r_m ou r_n.

9. soit U_{kc} l'ensemble des utilisateurs contradictoires pour le rôle r_k. Au maximum, un utilisateur dans l'ensemble U_{kc} est permis d'accéder ou d'activer le rôle r_k à n'importe quel moment donné. Formellement :

$$\sum_{u_i \in\ U_{kc}} u_{ir_k} \leq 1$$

4.3.3.3 La préservation de l'autonomie

Une exigence principale de l'interopérabilité sûre est la préservation de l'autonomie de tous les domaines collaborant. Cependant, la préservation de l'autonomie de différents domaines peut réduire l'interopération et dans certains cas peut éliminer totalement l'interopération . En d'autres termes, il y a une contradiction entre l'interopérabilité cherchante et la préservation de l'autonomie. Dans le cadre d'intégration des politiques RBAC, la violation de l'autonomie du domaine se produit à cause des deux raisons suivantes :

4.3.3.3.1 Les contraintes SoD induite

Une contrainte SoD induite est une contrainte SoD entre deux rôles intra-domaines, r_a et r_b, qui ne sont pas en conflit dans la politique RBAC de leur domaine original(initial). Une telle contrainte SoD est causée par d'autres rôles inter-domaines contradictoires r_c et r_d :

a. Domaine(r_c)$\neq$ Domaine(r_a) = Domaine(r_b).
b. Domaine(r_d)$\neq$ Domaine(r_a) = Domaine(r_b).
c. Conf-rset(r_c, r_d) $\wedge$ [($r_a \geq_I r_c \wedge r_b \geq_I r_d$) $\vee$ ($r_b \geq_I r_c \wedge r_a \geq_I r_d$)].

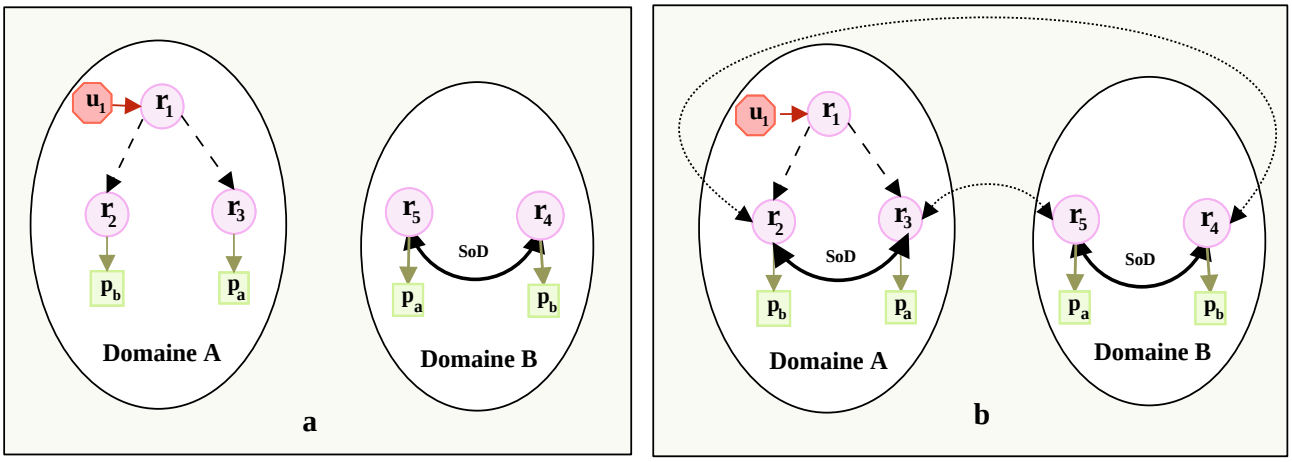


FIG. 4.4 – Exemple de SoD induite.

La figure 4.4(b) illustre une contrainte SoD induite entre deux rôles r_2 et r_3 du domaine A causée par les rôles r_4 et r_5 du domaine B. Notez que dans la politique RBAC originale du domaine A, montrée dans la figure 4.4 (a), r_2 et r_3 ne sont pas contradictoires dans leur domaine original. Suite à cette contrainte SoD induite, l'utilisateur u_1 qui est dans la politique RBAC originale du domaine A, autorisé d'accéder aux deux rôles r_2 et r_3 simultanément, n'est pas autorisé de les accéder dans la politique RBAC multi-domaines.

4.3.3.3.2 La cardinalité asymétrique des rôles

Il y a divers types de cardinalité liés aux rôles par exemple la cardinalité d'assignation du rôle, cardinalité d'activation du rôle,etc. Pour une politique RBAC cohérente, la cardinalité d'un rôle senior ne devrait pas être plus grande que la cardinalité de n'importe quel rôle de ses rôles juniors.

La composition d'une politique globale multi-domaines pour établir l'interopérabilité entre les différents domaines sans aucune violation de la sécurité et de l'autonomie des domaines collaborant, n'est pas une tâche faisable. Cependant, les domaines peuvent être disposés à compromettre leurs autonomies pour établir plus d'interopérabilité mais ces pertes d'autonomie restent dans des limites acceptables. Dans le suivant, un état de relaxation d'autonomie peut être incorporé comme une contrainte dans le problème IP.

Soit L_A représente l'ensemble de tous les rôles inter-domaines qui réduisent les cardinalités des rôles du domaine A ou ajoutent des contraintes SoD induites entre les rôles du domaine A. La perte globale de l'autonomie du domaine A causée par L_A est donnée par :

$$\mathrm{AL}(L_A) = \frac{\begin{pmatrix} \textit{Le nombre total des accès locaux sans} \\ \textit{considérer les liens des mappings} \\ \textit{des rôles inter} - \textit{domaines} \end{pmatrix} - \begin{pmatrix} \textit{Le nombre total des accès locaux en} \\ \textit{présence des liens des mappings} \\ \textit{des rôles inter} - \textit{domaines} \end{pmatrix}}{\begin{pmatrix} \textit{Le nombre total des accès locaux sans considérer} \\ \textit{les liens des mappings des rôles inter} - \textit{domaines} \end{pmatrix}}$$

On peut calculer la perte de l'autonomie d'un domaine, provoquée par un mapping des rôles, l par la formule $\mathrm{AL}(\{l\})$ tel que $l \in L_A$

$$AL(L_A) \leq \sum_{l \in L_A} AL(\{l\})$$

La raison de cette inégalité est que quelques accès locaux peuvent être réduits ensemble par des multiples liens des mappings des rôles inter-domaines.

Soit $S_i, (S_i \subseteq L_A)$ l'ensemble de tous les rôles inter-domaines qui réduisent les mêmes accès locaux du domaine A. Afin de garder la perte d'autonomie d'un domaine dans une certaine limite α, la contrainte d'autonomie suivante peut être ajoutée dans le formalisme IP :

$$\sum_{l_i \in L_A} \Big(\prod_{l_k \in S_i, i \neq k} (1 - u_{rk}) \Big) AL(\{l_i\}) u_{ri} + \sum_{(l_p, l_q \in L_A) \wedge ind\_Sod(l_p, l_q)} AL(\{l_p, l_q\}) u_{rp} u_{rq} \leq \alpha$$

La première somme dans la contrainte ci-dessus capture les pertes d'autonomie dues à la réduction des cardinalités des rôles. La variable de décision $u_{ri}(u_{rk})$ représente la préservation des liens des mappings des rôles inter-domaines l_i (l_k), c-à-d le lien $l_i(l_k)$ est préservé dans la politique finale si $u_{ri} = 1$ $(u_{rk} = 1)$. Le terme $[\prod(1 - u_{rk})]AL(l_i)u_{ri}$ impliquent que le lien du mapping des rôles l_i de l'ensemble AL (l_i) cause une perte d'autonomie si aucun autre lien du mapping des rôles de l'ensemble S_i n'est pas préservé dans la politique finale. Si un autre lien du mapping des rôles l_k est préservé, alors la perte d'autonomie due de l_i n'est pas prise en considération dans le calcul de perte de l'autonomie globale, parce que tous les accès locaux réduits par l_i sont aussi réduits par l_k, ce que signifie que la perte d'autonomie due à l_i est couverte par la perte d'autonomie due à l_k.

La deuxième somme $\sum AL(l_p, l_q)u_{rp}u_{rq}$ dans la contrainte ci dessus capture la perte d'autonomie causée par tous les paires des mappings des rôles qui provoquent l'ajout des contraintes SoD induites dans le domaine A. Le prédicat binaire ind-sod se tient pour n'importe quels deux mappings inter-domaines l_p et l_q si leur préservation dans la politique finale exige l'ajout d'une contrainte SoD induite.

4.3.3.4 L'algorithme de résolution de conflit

L'algorithme *ConfRes* pour résoudre les conflits du graphe RBAC **G** qui représente la politique globale de l'environnement multi-domaines. Avant de transformer les contraintes de politique RBAC en contraintes IP selon les règles présentées précédemment. Des utilisateurs factices sont affectés à deux classes de rôles qui n'ont pas été assignés à aucun utilisateur. La première classe inclut les rôles qui n'ont aucun rôle senior. L'affectation des utilisateurs factices aux rôles de cette classe s'assure que tous les rôles apparaissant dans les équations de contraintes IP. La deuxième classe inclut les rôles qui ont un ensemble non vide d'utilisateurs contradictoires (des utilisateurs qui sont en conflits sur ce rôle). L'utilisateur factice u_{dj} qui s'est assigné au rôle r_j, est également ajouté dans l'ensemble d'utilisateurs contradictoires pour le rôle r_j. Puisque u_{dj} est le seul utilisateur affecté à r_j donc $u_{djrj} = 1$. Ceci interdit n'importe quel utilisateur u_k qui est en conflit avec u_{dj} pour le rôle r_j, d'hériter des permissions de r_j par un rôle senior r_s sans activer r_j.

Une fois que toutes les contraintes IP sont définies, le problème IP est résolu en utilisant les critères d'optimalité dans la fonction objective. Basé sur la solution du problème IP, le graphe **G** est modifié en enlevant les arcs inter-domaines contradictoires avec la solution trouvée et les contraintes SoD induites correspondantes. Le graphe final présente la politique de contrôle d'accès multi-domaines qui répond aux exigences de sécurité et d'autonomie de tous les domaines collaborant.

1: Affecter un utilisateur factice u_{di} à tous les rôles r_i satisfont les deux conditions suivantes :
   a. Aucun utilisateur u n'est affecté à r_i.
   b. Il n'existe aucun rôle r_k pour lequel $r_k \geq_I r_i$.
2: Affecter un utilisateur factice u_{dj} à tous les rôles r_j qui ont un ensemble non vide d'utilisateurs contradictoires.
3: Pour chaque rôle r_j assigné à un utilisateur factice u_{dj} dans l'étape 2, met u_{djri}=1 et met à jour l'ensemble contradictoire d'utilisateurs en faisant ce qui suit :
   a. Ajouter une contrainte SoD spécifique d'utilisateur entre u_{dj} et tous les utilisateurs contradictoires pour le rôle r_j.
   b. Ajouter les nouveaux ensembles d'utilisateurs contradictoires pour le rôle r_j contenant l'utilisateur factice u_{dj} et un utilisateur u_k satisfaisant la condition suivante :

$$\exists u_i \in U, r_l \in R \; telque \; [(r_l \geq_A{}^* r_j) \wedge conf\_user(u_i, u_k, r_l) \wedge (u_i \in conf\_user(r_j))]$$

4: Ecrire les contraintes de politique RBAC d'interopération en forme algébrique, en employant les règles de transformation des contraintes.
5: Définir la fonction objective.
6: Trouver une solution faisable, optimale pour le problème IP.
7: Dans le graphe G de politique RBAC multi-domaines, supprimer les arcs inter-domaines (r_i, r_j) pour lequel il existe un utilisateur u_k tel que $u_{kri} = 1$ et $u_{krj} = 0$ dans la solution faisable et optimale.

$$G = G - \{(r_i, r_j) / \; \exists \; u_k \in U \; telque \; u_{kri} = 1 \; et \; u_{krj} = 0 \; et \; domain(r_i) \neq \; domain(r_j)\}$$

8: Pour chaque arc (r_i, r_j) supprimé de G, si r_j induit une contrainte SoD entre r_i et n'importe quel rôle r_k, cette contrainte SoD induite doit être enlevée du graphe G de politique RBAC d'interopération.
9: Dans le graphe G, supprimer les ensembles d'utilisateurs contradictoires ajoutés dans l'étape 3b.

FIG. 4.5 – L'algorithme *Confres*.

4.3.3.5 Exemples

4.3.3.5.1 Exemple1

Prenons l'exemple suivant : Soient A et B deux domaines, le domaine A contient quatre rôles r_1, r_2, r_3 et r_6. Les relations d'hiérarchies entre les rôles sont définies comme suit :

- Le rôle r_1 hérite le rôle r_6 ($r_1 \geq_I r_6$).
- Une relation d'hiérarchie d'activation entre r_1 et r_2 ($r_1 \geq_A r_2$).
- Une relation d'hiérarchie d'activation entre r_1 et r_3 ($r_1 \geq_A r_3$).

Le domaine B contient deux rôles r_4 et r_5, qui sont des rôles contradictoires,(une contrainte SoD est défini entre eux) ; aucun utilisateur ne peut posséder les deux rôles ensemble. Les politiques de contrôle d'accès RBAC des domaines A et B sont notées dans la figure 4.6(a).

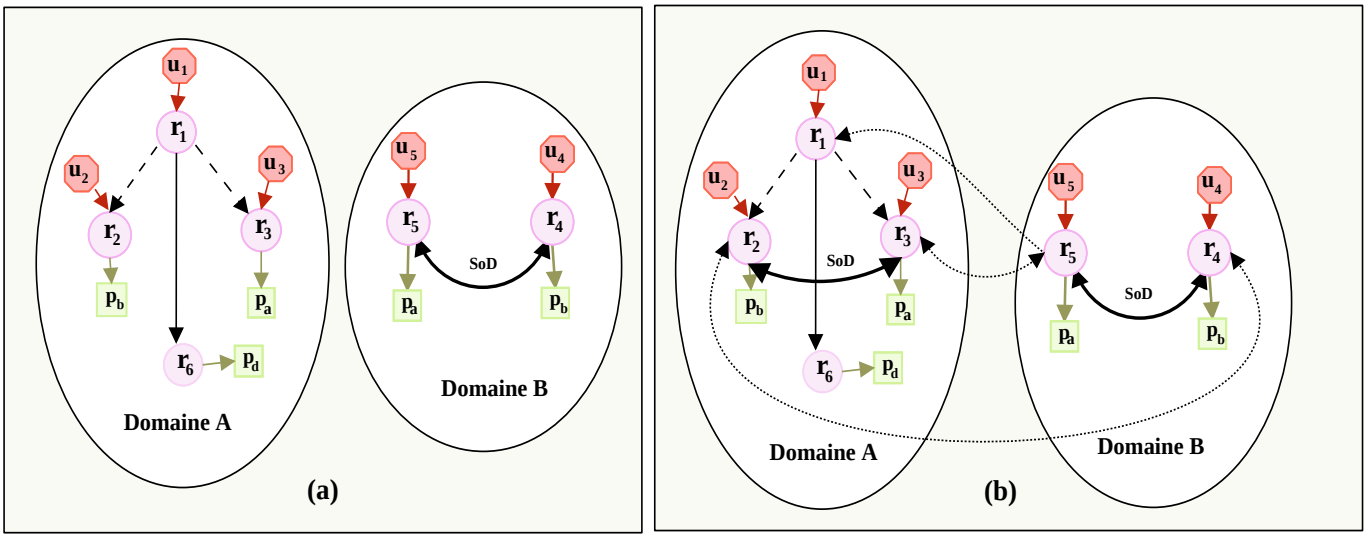


FIG. 4.6 – (a) les politiques RBAC des domaines A et B. (b)L'interopérabilité des politiques RBAC des domaines A et B.

L'interopérabilité des politiques RBAC de deux domaines A et B, est montrée dans la figure 4.6(b). l'interopérabilité est établie par la définition de deux mappings bidirectionnels des rôles inter-domaines ($r_2 : A$, $r_4 : B$), ($r_3 : A$, $r_5 : B$) et un mapping unidirectionnel ($r_5 : B \geq_I r_1 : A$) ajouté par l'administrateur. Le mappings bidirectionnel ($r_2 : A$, $r_4 : B$) implique que chaque utilisateur assume le rôle r_2, peut accéder au rôle inter-domaines r_4 ($r_2 : A \geq_I r_4 : B$) et chaque utilisateur assume le rôle r_4 peut accéder le rôle inter-domaines r_2 ($r_4 : A \geq_I r_2 : B$) et c'est la même chose pour le mapping bidirectionnel ($r_3 : A$, $r_5 : B$). Le mapping unidirectionnel ($r_5 : B \geq_I r_1 : A$) signifie que les utilisateurs assignés au rôle r_5 du domaine B peuvent assumer le rôle inter-domaines r_1 du domaine A.

une contrainte SoD induite, est ajoutée entre les rôles r_2, r_3 du domaine A, à cause des mappings des rôles inter-domaines entre les rôles r_2, r_3 et les rôles contradictoires du domaine B : r_4, r_5.

Cette interopérabilité n'est pas cohérente et elle cause des violations des contraintes de sécurité ; le mapping bidirectionnel des rôles inter-domaines (r_3 , r_5), et le mapping spécifie par l'administrateur(r_5 :B $\geq_I$ r_1 :A) causent une violation de contraintes d'assignation des rôles, ils permettent à l'utilisateur u_3 assigné au rôle junior r_3, d'accéder au rôle senior r_1. Pour résoudre ce conflit, un des deux mappings (r_3 :A $\geq_I$ r_5 :B) ou (r_5 :B $\geq_I$ r_1 :A) doit être

supprimé, dans les deux cas, le nombre des accès inter-domaines reste le même.

La contrainte SoD induite cause une perte d'autonomie égale à 16.67% selon la formule de la perte de l'autonomie définie dans le paragraphe 4.3.3.3, le nombre des accès permis dans le domaine A avant l'interopérabilité est six ; l'utilisateur u_1 peut accéder aux rôles r_2, r_3 et r_6, l'utilisateur u_2 peut accéder au rôle r_2, et l'utilisateur u_3 peut accéder au rôle r_3.

Après la composition des politiques RBAC des domaines A et B, l'utilisateur u_1 devient interdit d'activer en même temps les deux rôles r_2 et r_3 ; soit il active le rôle r_2 ou le rôle r_3, alors le nombre total des accès dans le domaine A devient cinq, donc la perte de l'autonomie dans le domaine A est AL ($\{(r_2, r_4),(r_3, r_5)\}$) = 6-5/6 $\simeq$ 16.67%. Supposant que la perte d'autonomie permise par le domaine A est 10 pourcent. Une contrainte de relaxation d'autonomie est ajoutée au système linéaire, définie comme suivant :

$$16.67 * (u_{3r_5} * u_{2r_4}) \leq 10$$

où $u_{3r_5} = 1$ (u_{2r_4}=1) implique que le mapping des rôles inter-domaines $r_3 : A \geq_I r_5 : B(r_2 : A \geq_I r_4 : B)$ est retenu dans la politique finale d'interopérabilité.

la formulation IP de la politique RBAC multi-domaines de la figure 4.6(b) est montrée dans La figure 4.7.

La fonction objectif : $u_{5r6} + u_{5r3} + u_{5r2} + u_{5r1} + u_{4r2} + u_{3r5} + u_{3r4} + u_{2r4} + u_{1r5} + u_{1r4}$

Les contraintes dérivées des règles 1, 2, 3, et 4 :

c1) $u_{1r1} = 1$ c2) $u_{1r6} = 1$ c3) $u_{2r1} = 0$ c4) $u_{2r2} = 1$ 5) $u_{2r3} = 0$ c6) $u_{2r6} = 0$
c7) $u_{3r1} = 0$ c8) $u_{3r2} = 0$ c9) $u_{3r3} = 1$ c10) $u_{3r6} = 0$ c11) $u_{2r5} = 0$ c12) $u_{4r4} = 1$
c13) $u_{4r5} = 0$ c14) $u_{5r4} = 0$ c15) $u_{5r5} = 1$ c16) $u_{4r1} = 0$ c17) $u_{4r3} = 0$ c18) $u_{4r6} = 0$

Les contraintes dérivées de la règle 5 :

c19) $u_{4r4} - u_{4r2} \geq 0$ c20) $u_{5r5} - u_{5r1} \geq 0$ c21) $u_{5r5} - u_{5r3} \geq 0$ c22) $u_{5r1} - u_{5r6} \leq 0$
c23) $u_{5r1} - u_{5r6} \geq 0$ c24) $u_{1r2} - u_{1r4} \geq 0$ c25) $u_{1r3} - u_{1r5} \geq 0$ c26) $u_{2r2} - u_{2r4} \geq 0$
c27) $u_{3r3} - u_{3r5} \geq 0$

Les contraintes dérivées de règle 6 :

c27) $(u_{2r2} - u_{2r4}) - (u_{1r2} - u_{1r4}) = 0$ c28) $(u_{2r2} - u_{2r4}) - (u_{5r2} - u_{5r4}) \geq 0$
c29) $(u_{3r3} - u_{3r5}) - (u_{1r3} - u_{1r5}) = 0$ c30) $(u_{4r4} - u_{4r2}) - (u_{3r4} - u_{3r2}) \geq 0$
c31) $(u_{5r5} - u_{5r1}) - (u_{3r5} - u_{3r1}) \geq 0$

Les contraintes dérivées de règle 7 :

c32) $u_{1r_2} + u_{1r_3} \leq 1$ c33) $u_{2r_2} + u_{2r_3} \leq 1$ c34) $u_{3r_2} + u_{3r_3} \leq 1$ c35) $u_{4r_2} + u_{4r_3} \leq 1$
c36) $u_{5r_2} + u_{5r_3} \leq 1$ c37) $u_{1r_4} + u_{1r_5} \leq 1$ c38) $u_{2r_4} + u_{2r_5} \leq 1$ c39) $u_{3r_4} + u_{3r_5} \leq 1$
c40) $u_{4r_4} + u_{4r_5} \leq 1$ c41) $u_{5r_4} + u_{5r_5} \leq 1$

Les contraintes dérivées de règle 8 :

c42) $u_{1r_2} + u_{1r_3} + u_{1r_4} + u_{1r_5} \leq 3$

La contrainte de relaxion de l'autonomie :

c43) $16.67 * (u_{3r_5} * u_{2r_4}) \leq 10$

FIG. 4.7 – La formulation IP de la politique RBAC multi-domaines montrée dans la figure.

Notons que dans la fonction objective, toutes les variables de décision représentant des accès inter-domaines sont assignées un poids égale à un, signifiant que le critère d'optimalité est de maximiser tous les accès inter-domaines. Une solution optimale du problème IP notée dans la figure 5.5 est donnée comme suivant [3] :

$$u_{1r_4} = 0, u_{1r_5} = 0, u_{2r_4} = 1, u_{3r_5} = 0, u_{4r_2} = 1, u_{5r_1} = 1, u_{5r_3} = 1, u_{5r_6} = 1, et\ u_{5r_2} = 0.$$

Puisque u_{3r_3} =1 (contrainte c9 dans la la figure 5.5), et $u_{3r_5} = 0$, le mapping des rôles inter-domaines $r_3 \geq_I r_5$ doit être supprimé du graphe de la politique RBAC multi-domaines notée dans la figure. La suppression du mapping des rôles inter-domaines $r_3 \geq_I r_5$ implique que la contrainte SoD induite entre les rôles r_2 et r_3 devient invalide et elle doit être supprimée. Donc la politique RBAC d'interopération ne cause aucune perte d'autonomie du domaine A.

4.3.3.5.2 Exemple2 Prenons la politique d'interopérabilité de trois domaines collaborant : *CTO*(County Treasure Office), *CCO* (County Clerk Office) et *CAO* (County Attorney Office) présentée dans la section 3.3.2 et notée dans la figure 3.6.

L'interopérabilité des politiques RBAC de ces domaines n'est pas sûre et contient plusieurs conflits. Dans le tableau 5.1 quelques conflits de cette politique RBAC d'interopération sont cités.

Mapping des rôles	**Violation de sécurité**	**Type de violation**	**Domaine affecté**
$TA : CTO \geq_I TAO : CCO$ $PIO : CCO \geq_I TRA : CTO$	permet au u_2 d'hériter les permissions des rôles contradictoire *TRE* et *TRA*.	SoD spécifique des rôles	*CTO*
$TA : CTO \geq_I TAO : CCO$ $DTM : CTO \geq_I ACAT : CAO$ $ACAT : CAO \geq_I TAC : CCO$	permet au u_1 d'hériter les permissions des rôles contradictoires *TAC* et *TAO*	SoD spécifique des rôles	*CCO*
$DTA : CTO \geq_I ACAT : CAO$ $PLAT_{05} : CAO \geq_I R2_{07} : CTO$	permet au u_4 d'accéder au rôle $R2_{07}$	Assignation des rôles	*CTO*
$TAC : CCO \geq_I DTA : CTO$ $PLAT_{06} : CAO \geq_I DTLO_{09} : CCO$	permet au u_8 d'accéder au rôle $DTLO_{09}$	Assignation des rôles	*CCO*
$DTLO : CCO \geq_I DTC : CTO$ $LSO : CCO \geq_I DTA : CTO$ $R10_{11} : CCO \geq_I DTM_{10} : CTO$	permet au u_6 d'accéder aux rôles contradictoire *DTA* et *DTC*, *DTA* et DTM_{10}	SoD spécifique des rôles	*CTO*

TAB. 4.1 – Les violations de sécurité dans la politique RBAC multi-domaines des domaines *CTO*, *CCO*, et *CAO*

Ces conflits sont résolus en appliquant l'algorithme de résolution de conflit *ConfRes*. *ConfRes* transforme d'abord les contraintes de politique RBAC en des contraintes *IP*, on obtient environ 1300 contraintes. Le problème IP est résolut avec l'objectif de maximiser les accès inter-domaines. la solution obtenue est présentée dans le tableau avec un maximun de 149 accès inter-domaines.

Le tableau présente les mappings des rôles inter-domaines qui doivent être supprimés pour que la politique RBAC multi-domaines notée dans la figure soit sûre.

Domaine CTO→Domaine CCO	**Domaine CTO→Domaine CAO**
	$(DTM, ACAT)$.
Domaine CCO→Domaine CTO	**Domaine CCO→Domaine CAO**
$(R9_{08}, R2_{07})$, $(R8_{06}, R2_{05})$, $(LSO_{04}, R4)$, (LSO, DTA), $(DTLO_{13}, DTM_{12})$, $(R10_{11}, DTM_{10})$, $(R10_{03}, R4_{02})$, (TAC, DTA), (PIO, TRA).	
Domaine CAO→Domaine CTO	**Domaine CAO→Domaine CCO**
$(PLAT_{05}, R2_{07})$, $(PLAT_{04}, R2_{05})$, $(PLAT_{03}, R4)$, $(PLAT_{07}, DTM_{12})$, $(PLAT_{08}, DTM_{10})$, $(PLAT_{00}, R4_{02})$, $(PLAT_{09}, DTM)$.	

TAB. 4.2 – La solution du problème *IP* de la politique globale RBAC des domaines : *CTO*, *CCO* et *CAO*, obtenue par l'algorithme *ConfRes*

4.4 Conclusion

Générer une politique RBAC multi-domaines, qui préserve les contraintes de sécurité et d'autonomie n'est pas une tâche facile ; maximiser le degrés d'interopérabilité implique une perte d'autonomie des domaines collaborant, et préserver l'autonomie implique la réduction de l'interopérabilité et dans certains cas l'élimination complètement de l'interopérabilité. BASIT a ajouté une contrainte de relaxion d'autonomie au système de contraintes IP généré de politique RBAC multi-domaines, ce qui permet d'obtenir une solution qui maximiser l'interopérabilité et garder la perte de l'autonomie des domaines collaborant dans des limites acceptables. Mais est ce que la méthode d'optimisation proposée par BASIT, peut détecter et résoudre tous les conflits qui peuvent apparaître dans une politique RBAC multi-domaines ? et est ce que la représentation des politiques multi-domaines obtenues est minimale ?

Chapitre 5

Proposition d'une amélioration d'optimisation des politiques d'interopérabilité RBAC

5.1 Introduction

Dans ce présente chapitre, nous allons montrer que la méthode d'optimisation proposée par BASIT [3] peut ne pas détecter et résoudre tous les conflits qui peuvent apparaître dans une politique globale RBAC des environnements multi-domaines, et la politique RBAC multi-domaines optimisée reste incohérente. Ainsi, nous proposons une amélioration de la méthode d'optimisation de BASIT pour détecter et résoudre tous les conflits probablement apparaissant dans une politique RBAC multi-domaines. Aussi, nous proposons une extension du processus de composition des politiques RBAC à fin qu'il assure une représentation minimale de la politique d'interopérabilité sûre de ces politiques de contrôle d'accès RBAC.

5.2 Les limitations de la méthode d'optimisation de BASIT

Il y a deux limitations dans la formulation IP(Integer programming) de la méthode d'optimisation de BASIT :

- La mauvaise formulation de la hiérarchie d'activation.
- Une contrainte de la règle(6) de la formulation IP est incorrecte.

5.2.1 La mauvaise formulation de la hiérarchie d'activation

Prenons l'exemple défini dans le chapitre précédent(cf.§ 4.3.3.5.1) de deux domaines A et B. La figure 5.1 représente la politique d'interopération RBAC optimisée par la méthode de BASIT de cet exemple.

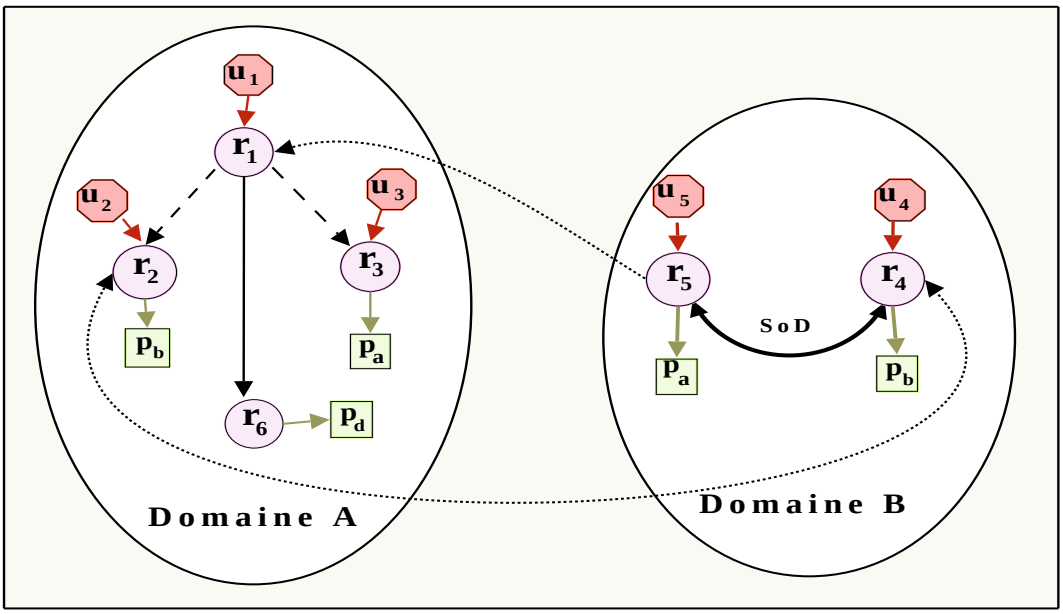


FIG. 5.1 – La politique optimisée incohérente de deux domaine A et B.

La politique d'interopérabilité obtenue n'est pas cohérente, comme le montre dans la figure 5.1, il y a une violation des contraintes d'assignation des rôles ; l'utilisateur u_5 assigné au rôle r_5 du domaine B peut accéder au rôle r_4 s'il active le rôle r_2 du domaine A. La méthode de BASIT n'a pas pu détecter ce conflit parce que dans le problème IP et selon ses contraintes(voir la figure 4.7), l'utilisateur u_5 n'assume pas le rôle r_2. Mais dans le graphe de la politique RBAC (noté dans la figure 4.6(b)), l'utilisateur u_5 accède au rôle r_1 à travers le mapping $r_5 : DomaineB \geq_I r_1 : DomaineA$, donc l'utilisateur u_5 peut activer le rôle r_2. La cause de cette contradiction est la mauvaise formulation des relations d'hiérarchie d'activation dans la formulation IP proposée par BASIT, il n'y a aucune règle qui dit que : si l'utilisateur u_5 puisse accéder au rôle r_1 alors u_5 peut activer le rôle r_2 à n'importe quel moment. Si en se basant sur la solution obtenue par BASIT et pour que l'interopérabilité optimisée soit sûre, la relation d'hiérarchie d'activation entre les rôles r_1 et r_2 doit être supprimée, et ça c'est une violation des contraintes d'autonomie (voir la section 3.2). Pour optimiser la politique RBAC d'interopérabilité, nous avons le droit de supprimer que les accès inter-domaines.

Dans la réalité, si l'utilisateur u_5 assume le rôle r_1 alors u_5 a la possibilité d'activer le rôle r_2 et donc d'accéder au rôle r_4, dans notre proposition, nous allons nous baser sur cette hypothèse.

5.2.2 Une contrainte de la règle (6) de la formulation IP est incorrecte

Selon la règle(6) de la formulation IP proposée par BASIT, si un utilisateur u_j peut accéder au rôle local r_k(du même domaine) et il existe un mapping des rôles inter-domaines du rôle r_k à un rôle inter-domaines r_l, alors si l'utilisateur u_i, tel que domaine(u_i) = domaine (r_k) et u_{ir_k} = 1, peut accéder au rôle inter-domaines r_l à travers le mapping (r_k, r_l) donc u_j peut aussi l'accéder : $(u_{ir_k} - u_{ir_l}) - (u_{jr_k} - u_{jr_l}) = 0$. Cette contrainte force que si l'utilisateur u_i ne peut pas accéder au rôle r_l à travers le mapping des rôles inter-domaines (r_k, r_l)(le mapping (r_k, r_l) peut être supprimé dans la solution du problème IP) alors u_j aussi ne peut pas l'accéder, et ça c'est faut parce que peut être il existe un autre chemin de u_j vers r_l, comme montre dans la figure 5.2, si le mapping des rôles inter-domaines (r_k, r_l)est supprimé, l'utilisateur u_i ne peut pas accéder au rôle r_l, mais l'utilisateur u_j peut l'accéder à travers le chemin du mapping des rôles inter-domaines (r_h, r_l).

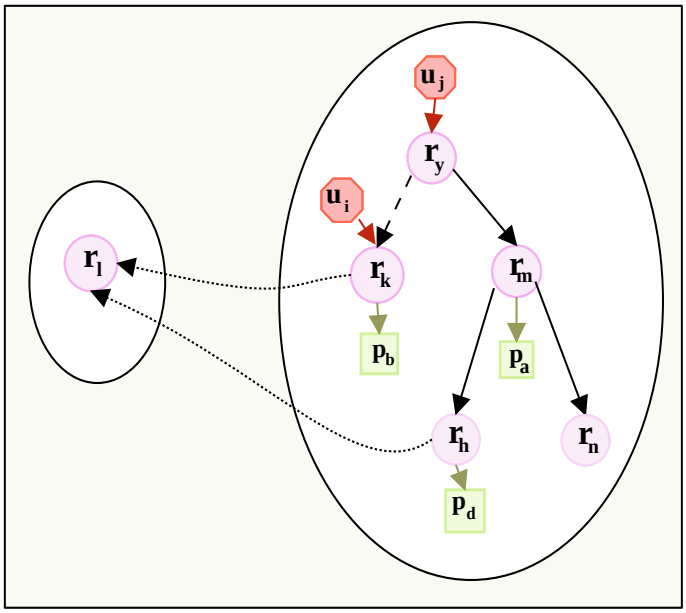


FIG. 5.2 – Exemple contradictoire de la justesse d'une contrainte de la règle(6).

5.3 La proposition d'une amélioration de la méthode d'optimisation de BASIT

Dans notre proposition, nous allons encapsuler les hiérarchies d'activation d'un domaine, aux utilisateurs des autres domaines, par exemple si l'utilisateur u_5 dans la figure 5.1 peut accéder au rôle r_1, alors il y a une possibilité que l'utilisateur u_5 peut accéder au rôle r_4 sans prendre en considération si u_5 active ou non le rôle r_2.
La hiérarchie d'activation est male formulée dans la méthode d'optimisation de BASIT, en ce qui concerne la détermination des accès des utilisateurs aux rôles inter-domaines. L'objectif des règles (5) et (6) dans la formulation IP(cf.§ 4.3.3.1) est de décrire les contraintes des accès des utilisateurs aux rôles inter-domaines. Pour palier les inconvénients de la méthode d'optimisation de BASIT, nous allons changer les deux règles (5) et (6) de la formulation IP par les règles suivantes *R5* et *R6*.

5.3.1 La règle *R5*

Dans la règle (5) de la formulation IP(cf.§ 4.3.3.1), BASIT a déterminé deux cas possibles pour qu'un utilisateur puisse accéder à un rôle inter-domaines :
Soient un utilisateur u_i et un rôle r_k de domaines différents, u_i peut accéder à r_k si un des deux cas est vrai :

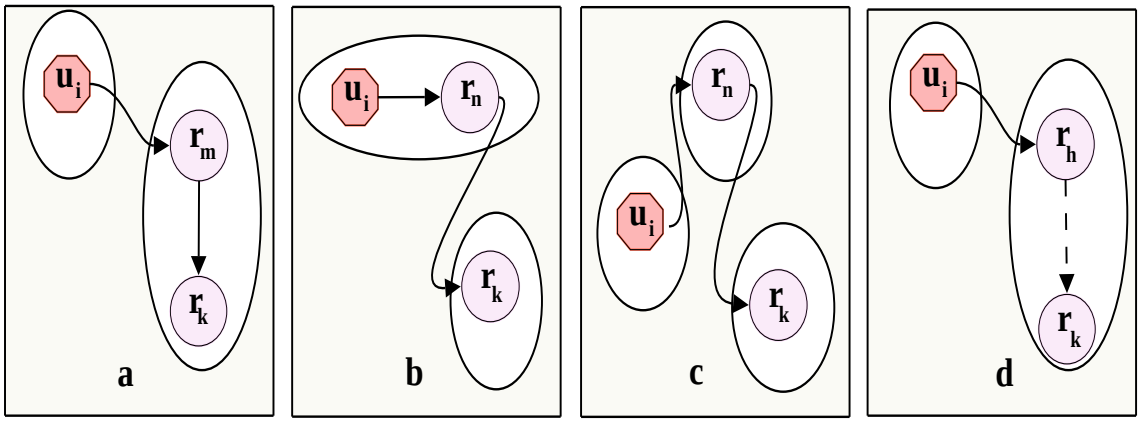


FIG. 5.3 – Les cas possibles pour qu'un utilisateur puisse aaccéder à un rôle inter-domaines.

a. Il existe un rôle r_m, tel que domaine(r_m)= domaine(r_k), $r_m \geq_I r_k$, et l'utilisateur u_i peut accéder à r_m, comme montre dans la figure 5.3(a).
b. Il existe un rôle r_n, tel que domaine(r_n)$\neq$ domaine(r_k), et $r_n \geq_I r_k$, et l'utilisateur u_i peut accéder r_n, comme montre dans la figure 5.3(b,c).

Il y a une troisième possibilité qui était ignorée dans la formulation IP de BASIT :

c. Il existe un rôle r_h, tel que domaine(r_h) = domaine(r_k), et $r_h \geq_A r_k$, et l'utilisateur u_i peut accéder au rôle r_h, comme montre dans la figure 5.3(d).

Nous définissons La règle *R5* comme suit :
Soient un utilisateur u_i et un rôle r_k de domaines différents, et quatre ensembles R_m, R_c, R_{pc}, et R_h sont définis comme suit :

R_m=$\{r_m|$ domaine(r_m) = domaine(r_k) et $r_m \geq_I r_k$ et dans le graphe RBAC, il y a un chemin de u_i à $r_m\}$.

R_c =$\{r | r \geq_I r_k \wedge domaine(r_k) \neq domaine(r)\}$.

R_{pc}=$\{r_p | \exists r \in R_c$ *tel que* $(r_p = r \wedge u\_assign(u, r)) \vee (r_p \geq_I r \wedge domaine(r) = domaine(r_p))\}$.

R_h =$\{r_h \mid domaine(r_k) = domaine(r_h) \wedge r_h \geq_A r_k$ *et dans le graphe RBAC il y a un chemin de* u_i r_h $\}$.

Les inégalités suivantes déterminent les conditions pour que l'utilisateur u_i puisse accéder au rôle r_k :

$$a. \quad \forall\, r_m \in R_m, u_{ir_m} - u_{ir_k} \leq 0$$

$$b. \quad \sum_{r_m \in R_m} u_{ir_m} + \sum_{r_h \in R_h} u_{ir_h} + \sum_{r_n \in R_c} u_{ir_n} - u_{ir_k} \geq 0$$

$$c. \quad \sum_{r_m \in R_m} u_{ir_m} + \sum_{r_h \in R_h} u_{ir_h} + \sum_{r_p \in R_{pc}} u_{ir_p} - u_{ir_k} \geq 0$$

5.3.2 La règle *R6*

Soient deux rôles r_k et r_l de deux domaines différents (domaine(r_k) $\neq$ domaine(r_l)), se sont liés par un mapping des rôles inter-domaines du rôle r_k au rôle r_l ($r_k \geq_I r_l$), et un utilisateur u_i de même domaine du rôle r_k autorisé de l'accéder, c'est-à-dire $u_{ir_k} = 1$.

1. S'il existe un rôle r_h, tel que domaine (r_h) = domaine (r_k) et $r_h \geq_A^* r_k$, ou il existe un

autre rôle r_i tel que $r_h \geq_A^* r_i \geq_I^* r_k$, alors :

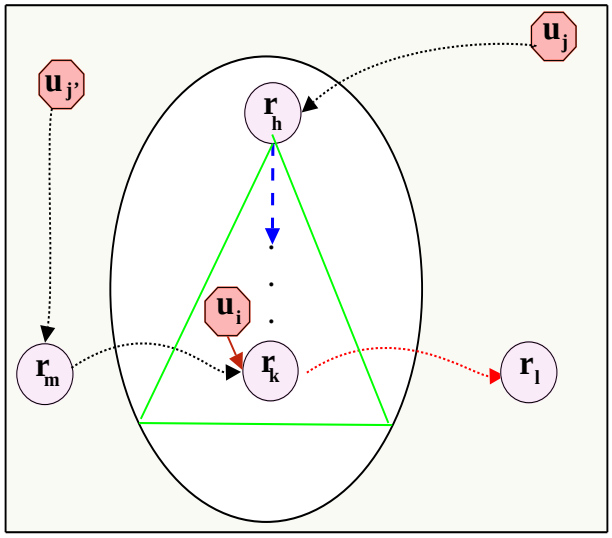


FIG. 5.4 – Encapsulation des relations d'hiérarchie d'activation des rôles aux utilisateurs inter-domaines.

a. Pour chaque utilisateur u_j, appartient au domaine différent du rôle $r_k(domaine(u_j) \neq domaine(r_k))$, peut accéder au rôle r_h, si l'utilisateur u_i est capable d'accéder à r_l à travers le chemin du mapping des rôles inter-domaines $(r_k\,;\, r_l)$, alors l'utilisateur u_j peut aussi l'accéder à travers le chemin du mapping des rôles inter-domaines ($r_k\,;\, r_l$)(voir la figure 5.4). Formellement :

$$(u_{jr_h} * u_{ir_l}) - u_{jr_l} \leq 0.$$

b. S'il existe un rôle r_m du domaine différent du rôle r_k (domaine $(r_m)\neq$ domaine (r_k)), et il y a un mapping des rôles inter-domaines du rôle r_m au rôle r_k ;$(r_m \geq_I r_k)$. Pour chaque utilisateur $u_{j'}$ appartient au domaine différent du rôle r_k $(domaine(u_{j'}) \neq domaine(r_k))$, peut assumer le rôle r_m, si l'utilisateur u_i est capable d'accéder à r_l à travers le chemin du mapping des rôles inter-domaines $(r_k\,;\, r_l)$, alors l'utilisateur $u_{j'}$ peut aussi l'accéder à travers le chemin du mapping des rôles inter-domaines ($r_k\,;\, r_l$), (voir la figure 5.4) :

$$(u_{ir_k} - u_{ir_l}) - (u_{j'r_k} - u_{j'r_l}) \geq 0.$$

2. Sinon, quelque soit l'utilisateurs u_j tel que Domaine$(u_j) \neq$ Domaine(r_k) :

a. Si l'utilisateur u_i est capable d'accéder au rôle r_l par le chemin du mapping des rôles inter-domaines$(r_k\,;\, r_l)$, alors l'utilisateur u_j s'il est autorisé d'accéder au rôle r_k, il peut aussi accéder à r_l à travers le chemin du mapping des rôles inter-domaines ($r_k\,;\, r_l$)(voir la figure 5.5(a, b)) :

$$(u_{ir_k} - u_{ir_l}) - (u_{jr_k} - u_{jr_l}) \geq 0.$$

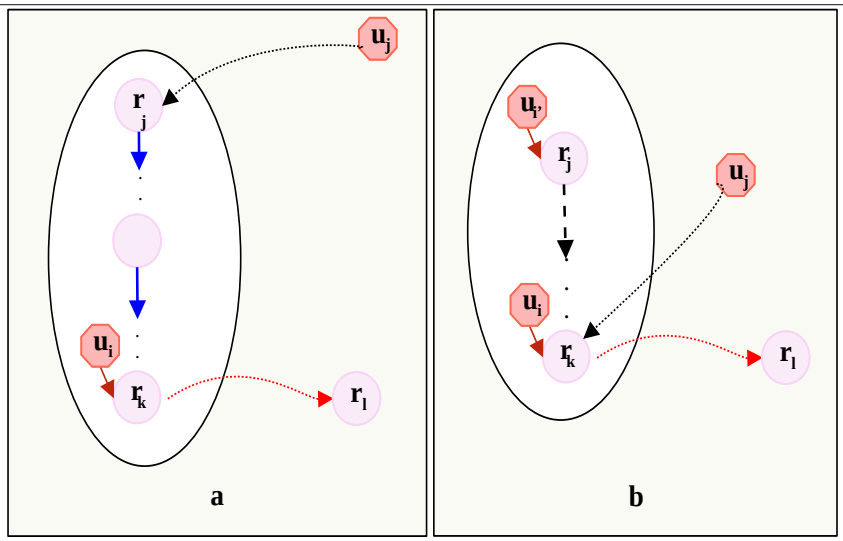


FIG. 5.5 – Les accès des utilisateurs aux rôles inter-domaines au présence que des relations d'hiérarchie d'héritage.

b. Aussi, s'il existe un utilisateur $u_{i'}$, tel que Domaine($u_{i'}$) = Domaine(r_k), et il existe un rôle r_j tel que $u_{i'r_j}$=1, et $r_j \geq_A^* r_k$ ou il existe un rôle r_i tel que $r_j \geq_A^* r_i \geq_I^* r_k$, pour chaque utilisateur u_j du domaine différent du rôle r_k, peut assumer le rôle r_k(voir la figure 5.5(b)), la contrainte suivante est ajoutée au système d'équations non linéaires :

$$(u_{i'r_j} - u_{i'r_l}) - (u_{jr_k} - u_{jr_l}) \geq 0.$$

5.4 L'implémentation de l'algorithme d'optimisation

Pour tester et évaluer notre amélioration de la méthode d'optimisation de BASIT et de la comparer avec la sienne, nous avons implémenté deux application, en utilisant le langage JAVA, une est basée sur la formulation IP de BASIT et l'autre sur notre formulation IP modifiée. Les données de ces applications est une politique d'interopérabilité RBAC. Ces applications génèrent des systèmes non linéaires correspondant à la politique. On parle de modèle d'optimisation non linéaire (NLP) lorsque l'on doit optimiser une fonction sous contraintes et que soit la fonction objectif, soit au moins une contrainte est non linéaire ; pour la méthode d'optimisation de BASIT, la contrainte de relaxion est non linéaire, donc le système généré est non linéaire, pour notre méthode d'optimisation la règle R5 génére quelques contraintes non linéaires en plus de la contrainte de relaxion alors le système généré est aussi non linéaire, en plus les variables prennent des valeurs entiers alors on parle d'un problème MINLP (Mixed-Integer Non Linear Programming). Pour résoudre les systèmes non linéaires générés, on utilise le logiciel *GAMS*[1](General Algebraic Modelling System).

[1] www.gams.com

5.4.1 La présentation du logiciel *GAMS*

GAMS, initialement développé par un groupe d'économistes de la Banque Mondiale [12] suite à la complexité des problèmes de modélisation économique, *GAMS* dispose d'un langage d'entrée qui permet d'écrire des modèles avec une formulation algébrique concise et facilement lisible. *GAMS* est capable de résoudre des problèmes aussi volumineux que complexes ; Il peut être utilisé pour traiter une grande variété de problèmes (NLP, MIP, MINLP, etc). La simplicité du formalisme *GAMS* est un avantage pour la description des problèmes. La particularité de cet environnement d'optimisation réside aussi dans le fait qu'il permet une description des modèles indépendantes des algorithmes de résolution, c'est-à-dire, que si le choix d'un changement au niveau de la méthode de résolution est effectué, aucune modification particulière de la formulation n'est nécessaire. La conception de *GAMS* prend en considération des concepts issus des techniques de manipulation des données théoriques et de la programmation mathématique. La fusion des deux approches permet la mise en place d'une stratégie globale facilitant la formulation du problème pour l'utilisateur. Les principes généraux utilisés pour la définition du système *GAMS* sont les suivants :

- La syntaxe proposée à l'utilisateur pour définir un problème est générique, c'est-à-dire indépendante des méthodes de résolution. Cette syntaxe s'applique à tout type de problème, linéaire, non linéaire, mixte linéaire et mixte non linéaire.
- La description du problème d'optimisation est indépendante des données nécessaires. Cette description est effectuée par un ensemble de relations algébriques et de déclarations de types, l'introduction des données se faisant par des affectations, comme dans tous les langages évolués de programmation.
- L'allocation des ressources est effectuée automatiquement suivant la complexité du problème. L'utilisateur n'aura pas à se soucier de certains détails comme les tailles allouées et la conservation de mémoire.

Deux logiciels ont été développés pour la résolution des problèmes MINLP : *SBB* (Simply Branch & Bound) et *DICOPT* (Discrete Continuous Optimization Package). Le premier est basé sur une procédure de résolution Branch & Bound (Gupta et Ravindram, 1985) [12]. Le second repose sur le principe de l'approximation externe, égalité relaxation et pénalité augmentée (Viswanathan et Grossmann, 1990) [12]. Utilisant des stratégies de résolution différentes, ils ont été appliqués à des cas d'exemples très complexes et se sont montrés très performants. Nous avons appliqué ces deux logiciels sur nous exemples, et nous avons obtenu les mêmes résultats, donc dans ce présente travail, nous allons expliquer le principe de travail du module *SBB*.

5.4.2 le module *SBB*

Le module de résolution *SBB* (Simple Branch & Bound) est plus récent que *DICOPT*, puisque sa première implémentation date d'octobre 2000. Son principe repose sur la combinaison d'un algorithme Branch & Bound classiquement utilisé pour la résolution de problèmes MILP et d'un solveur de type NLP.

5.4.2.1 La méthode Branch and Bound

Initialement développée par Gupta (1980) et Gupta et AL (1981) [12], le principe général de la méthode d'énumération implicite est le suivant : toutes les solutions possibles du problème peuvent être énumérées mais l'analyse des propriétés du problème permet d'éviter l'énumération de larges classes de mauvaises solutions. Dans un bon algorithme par séparation et évaluation successives, seules les solutions potentiellement intéressantes sont donc énumérées.

L'approche de type Branch-and-Bound communément appelée méthode de séparation et d'évaluation est une méthode générique de résolution de problèmes d'optimisation, et plus particulièrement d'optimisation combinatoire ou discrète. Dans ces méthodes, la séparation permet de pouvoir énumérer toutes les solutions, tandis que l'évaluation évite l'énumération systématique de toutes les solutions. La phase de séparation consiste à diviser le problème en un certain nombre de sous problèmes qui ont chacun leur ensemble de solutions réalisables de telle sorte que tous ces ensembles recouvrent le problème initial [22, 23]. Dans le cas d'un problème MINLP, les variables discrètes sont considérées continues, d'où des sommets où chacun implique un problème de programmation non linéaire en variables continues (NLP). Par la suite des contraintes de bornes sont ajoutées sur certaines de ces variables de façon à obtenir des valeurs entières situées sur les bornes. En comparant les solutions des divers problèmes continus, on obtient une borne inférieure sur la valeur optimale du critère. On détermine également une borne supérieure sur cette valeur, qui correspond à une solution d'un problème NLP pour laquelle toutes les valeurs des variables discrètes du problème initial sont entières.

5.4.2.2 L'algorithme de *SBB*

L'idée directrice de l'algorithme de *SBB* est une reproduction du fonctionnement d'un algorithme Branch & Bound, habituellement dédié au traitement de problèmes de type MILP. La différence repose sur la résolution, à chaque sommet du Branch & Bound, d'un sous-problème continu non-linéaire au moyen d'un solveur NLP de GAMS.

Concernant la première itération, l'algorithme *SBB* débute en résolvant le problème MINLP relaxé (RMINLP). Celui-ci, défini comme étant la relaxation continue du problème initial, est donc de type NLP. Si elle est pas faisable, *SBB* s'arrête. Si toutes les variables discrètes prennent des valeurs entières, *SBB* retourne le résultat trouvé comme étant la solution optimale. Dans les autres cas, la mise en oeuvre de la procédure arborescente est amorcée.

Pour chaque traitement de la relaxation continue d'un sommet, les sommets enfants sont placés dans une liste de sommets à traiter. Cette liste est actualisée à chaque itération. C'est le cas, par exemple, lorsqu'une nouvelle borne supérieure est générée : les sommets de la liste présentant une borne inférieure plus grande que cette nouvelle valeur sont éliminés. L'algorithme s'arrête alors lorsque la liste de sommets est vide. La représentation schématique de l'algorithme mis en oeuvre dans *SBB* apparaît sur la figure 5.11 :

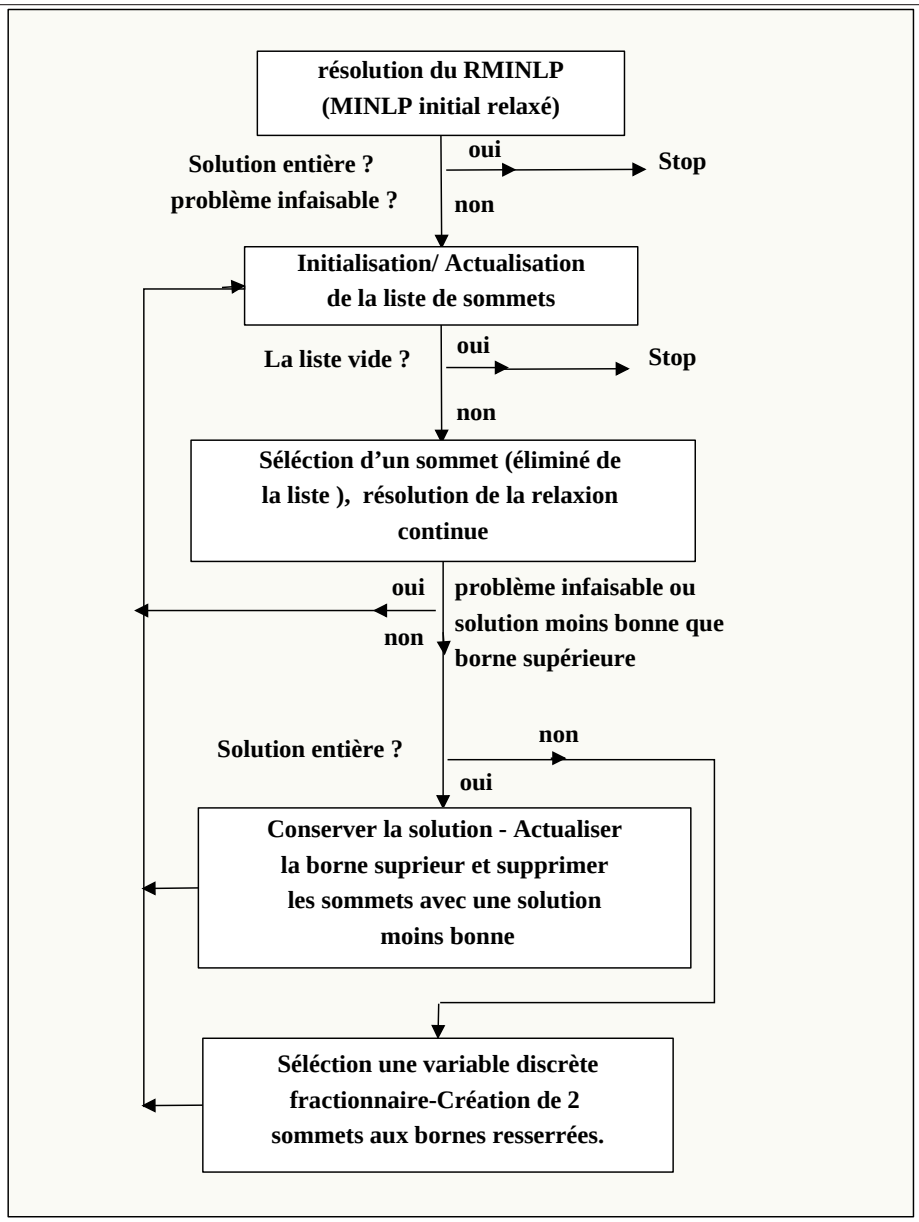


FIG. 5.6 – Organigramme de l'algorithme SBB.

5.5 Test et discussion

Pour faire la comparaison entre la formulation IP de BASIT, et notre formulation IP modifiée, on prend les exemples suivants :

5.5.1 Exemple1

prenons l'exemple de la politique d'interopérabilité de deux domaines A et B présenté dans cf.§ 4.3.3.5.1, Le système d'équations non linéaire obtenu de la politique d'interopérabilité RBAC de cet exemple (notée dans la figure 4.6(b)) par l'application de notre formulation IP modifiée est noté dans la figure 5.7

La fonction objectif : $u_{5r6} + u_{5r3} + u_{5r2} + u_{5r1} + u_{4r2} + u_{3r5} + u_{3r4} + u_{2r4} + u_{1r5} + u_{1r4}$

Les contraintes dérivées des règles 1, 2, 3, et 4 :

c1) $u_{1r1} = 1$ c2) $u_{1r6} = 1$ c3) $u_{2r1} = 0$ c4) $u_{2r2} = 1$ 5) $u_{2r3} = 0$ c6) $u_{2r6} = 0$
c7) $u_{3r1} = 0$ c8) $u_{3r2} = 0$ c9) $u_{3r3} = 1$ c10) $u_{3r6} = 0$ c11) $u_{2r5} = 0$ c12) $u_{4r4} = 1$
c13) $u_{4r5} = 0$ c14) $u_{5r4} = 0$ c15) $u_{5r5} = 1$ c16) $u_{4r1} = 0$ c17) $u_{4r3} = 0$ c18) $u_{4r6} = 0$

Les contraintes dérivées de règle R5 :

c19) $u_{2r_4} - u_{1r_4} \geq 0$ c20) $u_{3r_5} - u_{1r_5} \geq 0$ c21) $u_{3r_2} - u_{3r_4} \geq 0$ c22) $u_{2r_2} - u_{2r_4} \geq 0$
c23) $u_{3r_3} - u_{3r_5} \geq 0$ c24) $u_{4r_4} - u_{4r_2} \geq 0$ c25) $u_{5r_5} - u_{5r_1} \geq 0$ c26) $u_{5r_1} - u_{5r_6} \leq 0$
c27) $u_{5r_1} - u_{5r_6} \geq 0$ c28) $(u_{5r_1} * u_{2r_4}) - u_{5r_4} \geq 0$ c29) $u_{5r_1} + u_{5r_5} - u_{5r_3} \geq 0$
c30) $u_{5r_1} + u_{5r_4} - u_{5r_2} \geq 0$

Les contraintes dérivées de règle R6 :

c31) $(u_{4r_4} - u_{4r_2}) - (u_{3r_4} - u_{3r_2}) \geq 0$ c32) $(u_{5r_5} - u_{5r_1}) - (u_{3r_5} - u_{3r_1}) \geq 0$

Les contraintes dérivées de règle 7 :

c33) $u_{1r_2} + u_{1r_3} \leq 1$ c34) $u_{2r_2} + u_{2r_3} \leq 1$ c35) $u_{3r_2} + u_{3r_3} \leq 1$ c36) $u_{4r_2} + u_{4r_3} \leq 1$
c37) $u_{5r_2} + u_{5r_3} \leq 1$ c38) $u_{1r_4} + u_{1r_5} \leq 1$ c39) $u_{2r_4} + u_{2r_5} \leq 1$ c40) $u_{3r_4} + u_{3r_5} \leq 1$
c41) $u_{4r_4} + u_{4r_5} \leq 1$ c42) $u_{5r_4} + u_{5r_5} \leq 1$

Les contraintes dérivées de règle 8 :

c43) $u_{1r_2} + u_{1r_3} + u_{1r_4} + u_{1r_5} \leq 3$

La contrainte de relaxion de l'autonomie :

c44) $16.67 * (u_{3r_5} * u_{2r_4}) \leq 10$

FIG. 5.7 – Le système d'équations non linéaires de la politique RBAC multi-domaines généré par notre formulation IP.

La solution de système est :

$$u_{1r_4} = 0, u_{1r_5} = 1, u_{2r_4} = 0, u_{3r_5} = 1, u_{4r_2} = 1, u_{5r_1} = 0, u_{5r_3} = 1, u_{5r_6} = 0, et\ u_{5r_2} = 0.$$

Puisque u_{2r_2} =1(contrainte *c4* dans la figure 5.7), et $u_{2r_4} = 0$, alors le mapping des rôles inter-domaines $r_2 \geq_I r_4$ doit être supprimé du graphe de la politique RBAC multi-domaines noté dans la figure 4.6, et u_{5r_5} =1 (contrainte *c15* dans la figure 5.7), et u_{5r_1} =0 signifie aussi que le mapping des rôles inter-domaines $r_5 \geq_I r_1$ doit être supprimé du graphe de politique RBAC multi-domaines. La suppression du mapping des rôles inter-domaines $r_2 \geq_I r_4$ implique que la contrainte SoD induite entre les rôles r_2 et r_3 devient invalide et elle doit être supprimée. Donc la politique RBAC d'interopération ne cause aucune perte d'autonomie du domaine A et aucune violation de sécurité, la politique d'interopération RBAC optimisée est sûre (voir la figure 5.8).

Si on veut forcer la préservation d'un mapping des rôles inter-domaines dans la politique

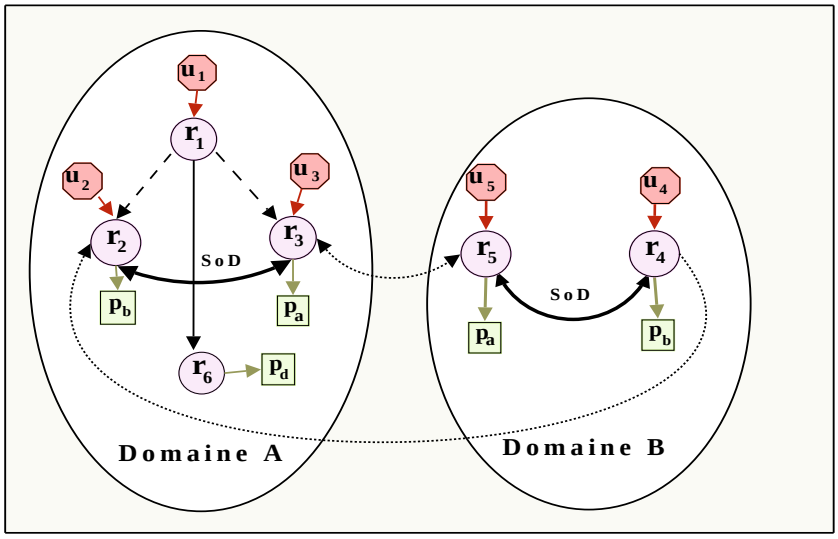


FIG. 5.8 – La politique optimisée cohérente de deux domaine A et B.

RBAC multi-domaines optimisée, on utilise les poids. Par exemple dans la politique d'inter-opérabilité ci-dessus si on veut préserver le mapping des rôles inter-domaines u_{2r_4} on lui associé dans la fonction objective un poids égale à deux, et la solution sera :

$$u_{1r_4} = 1, u_{1r_5} = 0, u_{2r_4} = 1, u_{3r_5} = 0, u_{4r_2} = 1, u_{5r_1} = 0, u_{5r_3} = 1, u_{5r_6} = 0, etu_{5r_2} = 0.$$

5.5.2 Exemple2

Prenons l'exemple de la politique d'interopérabilité de trois domaine *CTO*, *CCO* et *CAO* présenté dans la section 3.3.2, cette politique d'interopérabilité n'est pas cohérente et plusierus types de conflits sont apparus(cf.§ 4.3.3.5.2). L'application de la formulation IP modifiée génère environ 1500 contraintes. La solution de ce système non linéaire est notée dans le tableau 5.1. La solution obtenue est presque la même solution avec la formulation IP de BASIT sauf que

Domaine CTO→Domaine CCO	**Domaine CTO→Domaine CAO**
	(TA,TAO).
Domaine CCO→Domaine CTO	**Domaine CCO→Domaine CAO**
($R9_{08}$, $R2_{07}$), ($R8_{06}$, $R2_{05}$), (LSO_{04}, $R4$), ($DTLO$, DTC), ($DTLO_{13}$, DTM_{12}), ($R10_{11}$, DTM_{10}), ($R10_{03}$, $R4_{02}$), (TAC, DTA), (PIO, TRA).	
Domaine CAO→Domaine CTO	**Domaine CAO→Domaine CCO**
($PLAT_{05}$, $R2_{07}$), ($PLAT_{04}$, $R2_{05}$), ($PLAT_{03}$, $R4$), ($PLAT_{07}$, DTM_{12}), ($PLAT_{08}$, DTM_{10}), ($PLAT_{00}$, $R4_{02}$), ($PLAT_{09}$,DTM).	

TAB. 5.1 – La solution obtenue du problème *IP* de la politique globale RBAC des domaines : *CTO*, *CCO* et *CAO*, par l'algorithme *ConfRes*avec la formulation IP modifiée

les mappings des rôles inter-domaines (LSO, DTA) et (DTM, $ACAT$) sont préservés et les mappings (TA,TAO) et ($DTLO$, DTC) sont supprimés dans la politique RBAC finale, cette solution accroît le nombre maximal des accès inter-domaines de 149 à 153. Dans la solution précédente (voir le tableau5.1) les mappings (LSO, DTA) et (DTM,$ACAT$) sont supprimés à cause de la contrainte incorrecte de la règle (6)(voir la section 5.2.2).

5.6 La représentation minimale de l'interopérabilité des politiques RBAC

Les méthodes d'optimisation existantes peuvent fournir des politiques d'interopération cohérentes, mais la représentation des politiques n'est pas forcément minimale ; des mappings des rôles inter-domaines redondants peuvent apparaître. Sachant qu'un des critères importants d'optimisation est la représentation minimale de la politique RBAC.

5.6.1 Mapping redondant des rôles inter-domaines

Définition : On dit qu'un mapping des rôles inter-domaines unidirectionnel (r_i, r_k) du rôle r_i au rôle r_k des domaines respectivement A et B est redondant s'il offre, pour chaque utilisateur du domaine A qui assume le rôle r_i, les mêmes permissions ou un sous ensemble de permissions du domaine B, fournis par un autre mapping des rôles inter-domaines du domaine A au domaine B.

La composition des politiques RBAC consiste à établir des mappings entre les rôles des domaines collaborants. Il y a deux types de mappings des rôles inter-domaines définis dans l'algorithme de composition proposé par BASIT :

a. Les mappings créés entre les rôles inter-domaines équivalents, selon leurs ensembles de permissions assignées et leurs ordres hiérarchiques.
   - Si les rôles inter-domaines par exemple r_1 et r_2 sont équivalents alors ils sont liés ensemble, en définissant un mapping des rôles bidirectionnels entre eux, c'est-à-dire, $r_1 \geq_I r_2$ et $r_2 \geq_I r_1$.
   - Si la relation entre les rôles est contient ; r_1 contient r_2 ($r_1 \supset r_2$) selon leurs attributions de permissions et leurs ordres hiérarchiques alors un nouveau rôle junior r_{1j} du rôle r_1 équivalent au rôle r_2 sera créer et il est lié avec r_2 par un mapping des rôles unidirectionnel $r_{1j} \geq_I r_2$.

2) Les mappings ajoutés par l'administrateur de sécurité. L'administrateur ajoute un mapping des rôles unidirectionnel (r_{A_i}, r_{B_j}) entre deux rôles inter-domaines r_{A_i} du domaine A et r_{B_j} du domaine B, pour satisfaire certaines exigences de l'interopérabilité et il n'existe aucun rôle r_{A_k} tel que r_{A_k} appartient au domaine A, et r_{A_k} et le rôle r_{B_j} du domaine B se sont liés par un mapping des rôles inter-domaines (r_{A_k}, r_{B_j}). Donc on ne peut pas trouver un mapping des rôles inter-domaines, introduit par l'administrateur, redondant.

Les nouveaux rôles créés pendant la phase de composition ne sont pas associés directement à aucun utilisateur, et ils sont créés pour qu'on puisse faire les mappings des rôles inter-domaines dans des cas où c'est impossibles, de le faire avec les rôles initials des domaines collaborants. Donc tous ces nouveaux rôles sont liés avec d'autres rôles inter-domaines équivalents avec eux selon leurs structures hiérarchiques, en plus il y a des relations d'héritages entre les rôles du même domaine, ce qui peut créer plusieurs chemins aux utilisateurs d'un domaine source pour accéder aux permissions d'un domaine destinataire. Exemples :

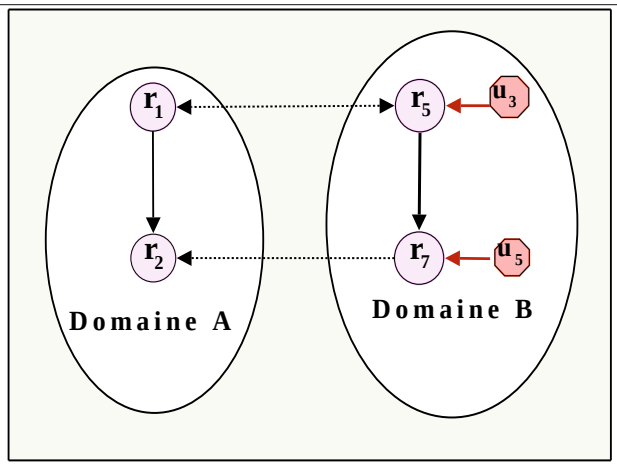


FIG. 5.9 – Exemple des mappings des rôles inter-domaines non redondants.

Dans la figure 5.9, on observe que l'utilisateur u_3 peut accéder aux permissions du rôle r_2 à travers deux chemins différents ; soit par le mapping des rôles inter-domaines bidirectionnel (r_5, r_1) ou le mapping des rôles inter-domaines unidirectionnel (r_7, r_2), mais on ne peut pas supprimer le mapping (r_7, r_2) ; parce que l'utilisateur u_5 ne peut pas accéder aux permissions inter-domaines du rôle r_2.

Prenons maintenant l'exemple noté dans la figure 5.10, le rôle r_7 est un nouveau rôle junior du rôle r_5 créé pendant la phase de composition, à ce rôle n'associe aucun utilisateur, l'utilisateur u_3 peut l'accéder à travers la relation d'hiérarchie d'héritage.

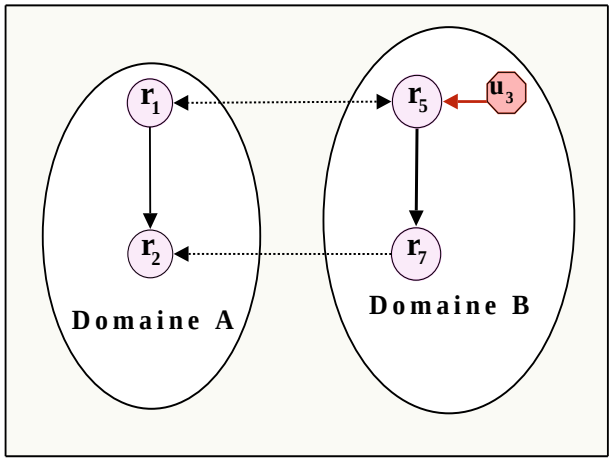


FIG. 5.10 – Exemple d'un mapping redondant des rôles inter-domaines .

Dans ce cas le mapping des rôles inter-domaines unidirectionnel (r_7, r_2) est redondant et sa suppression n'influe pas sur l'ensemble F des accès permis ; u_3 peut accéder aux permissions du rôle inter-domaine r_2 par l'héritage.

5.6.2 La proposition

Notre idée est d'ajouter, après la phase de détection et de résolution des conflits, une phase qui permet de donner la représentation minimale de la politique globale RBAC des environnements multi-domaine.

5.6.2.1 L'algorithme de la représentation minimale

On combine tous les domaines entre eux deux à deux.

1. A chaque itération, on prend deux domaines, par exemple un domaine A et un domaine B, et on s'intéresse aux mappings des rôles inter-domaines du domaine source A vers domaine destinataire B.

2. Pour chaque mapping des rôles inter-domaines (r_A, r_B) on calcule l'ensemble des couples (u_A, p_B) des utilisateurs et des permissions des domaines A et B respectivement , et on détermine pour chaque utilisateur du domaine A qui peut accéder au rôle r_A, les permissions qu'il peut assigner du domaine B à travers le mapping des rôles inter-domaines (r_A, r_B).

3. On compare tous les ensembles de paires utilisateur-permission des mappings des rôles interdomaines du domaine A au domaine B, s'il existe un ensemble inclut dans un autre alors le mapping des rôles inter-domaines correspondant à cet ensemble est supprimé.

5.6.2.2 L'évaluation

Pour évaluer la proposition, on prend les deux exemples suivants :

5.6.2.2.1 Exemple1

Soit l'exemple suivant(noté dans la figure 5.11) qui représente une politique d'interopérabilité sûre de deux domaines A et B : On applique l'algorithme de la représentation minimale, on a deux itérations :

- **Itération 1 :** Domaine $A \rightarrow$ Domaine B.
- **Itération 2 :** Domaine $B \rightarrow$Domaine A.

On prend par exemple l'itération 2 (Domaine $B\rightarrow$ Domaine A), les ensembles de paires des utilisateurs et des permissions de chaque mapping des rôles inter-domaines du domaine B au

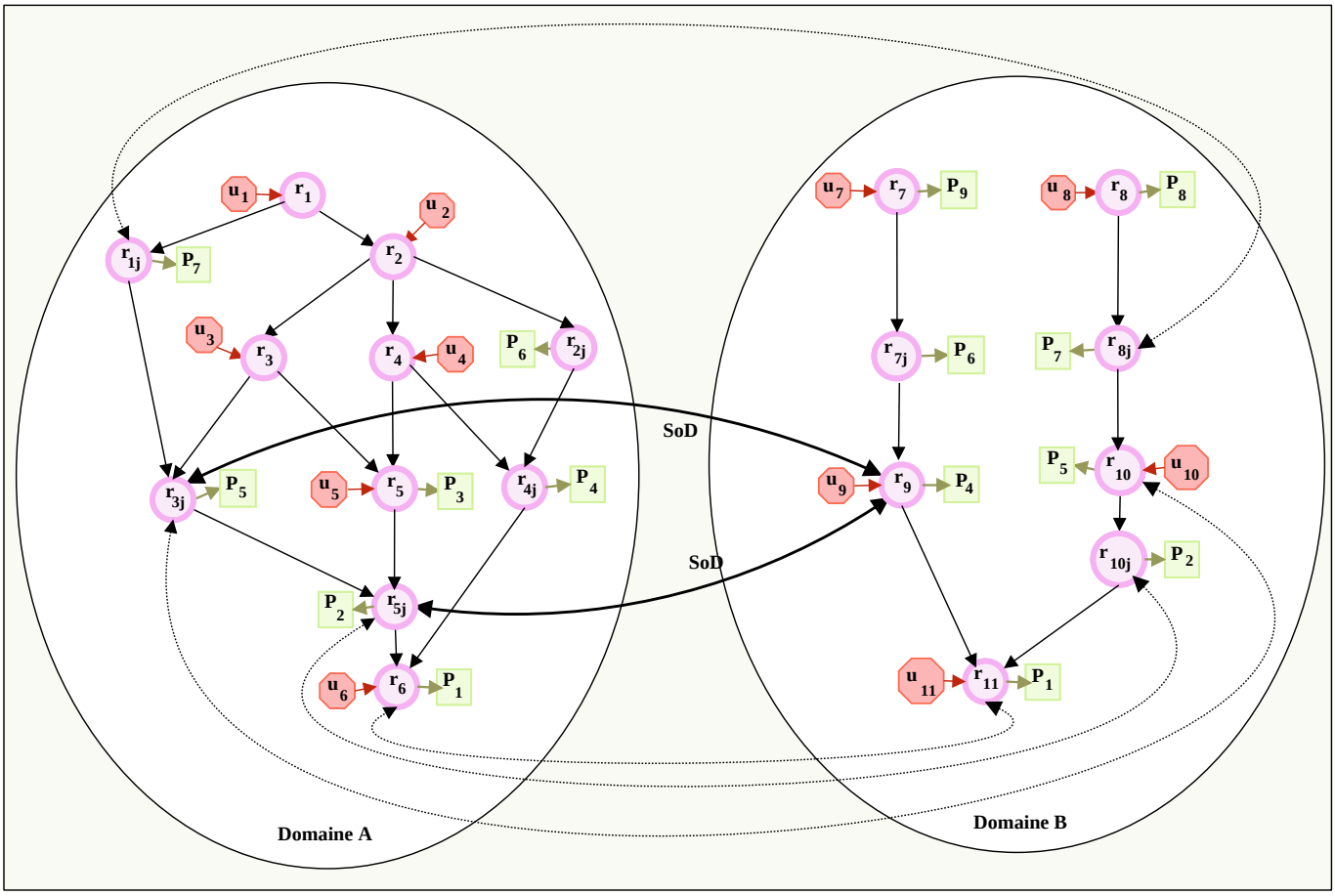


FIG. 5.11 – La politique d'interopérabilité sûre des deux domaines A et B.

domaine A sont :

- (r_{10}, r_{3j}) : $\{(u_8, p_5), (u_8, p_2), (u_8, p_1), (u_{10}, p_5), (u_{10}, p_2), (u_{10}, p_1)\}$.
- (r_{10j}, r_{5j}) : $\{(u_8, p_2), (u_8, p_1), (u_{10}, p_2), (u_{10}, p_1)\}$.
- (r_{11}, r_6) : $\{(u_8, p_1), (u_7, p_1), (u_{10}, p_1)\}$.

On trouve que le mapping des rôles inter-domaines (r_{10j}, r_{5j}) fournit aux utilisateurs du domaine B un sous ensemble de permissions du domaine A obtenues à travers le mapping des rôles inter-domaines (r_{10}, r_{3j}). Alors le mapping des rôles inter-domaines (r_{10j}, r_{5j}) est redondant et sa suppression n'influe pas sur l'ensemble F des accès légaux.

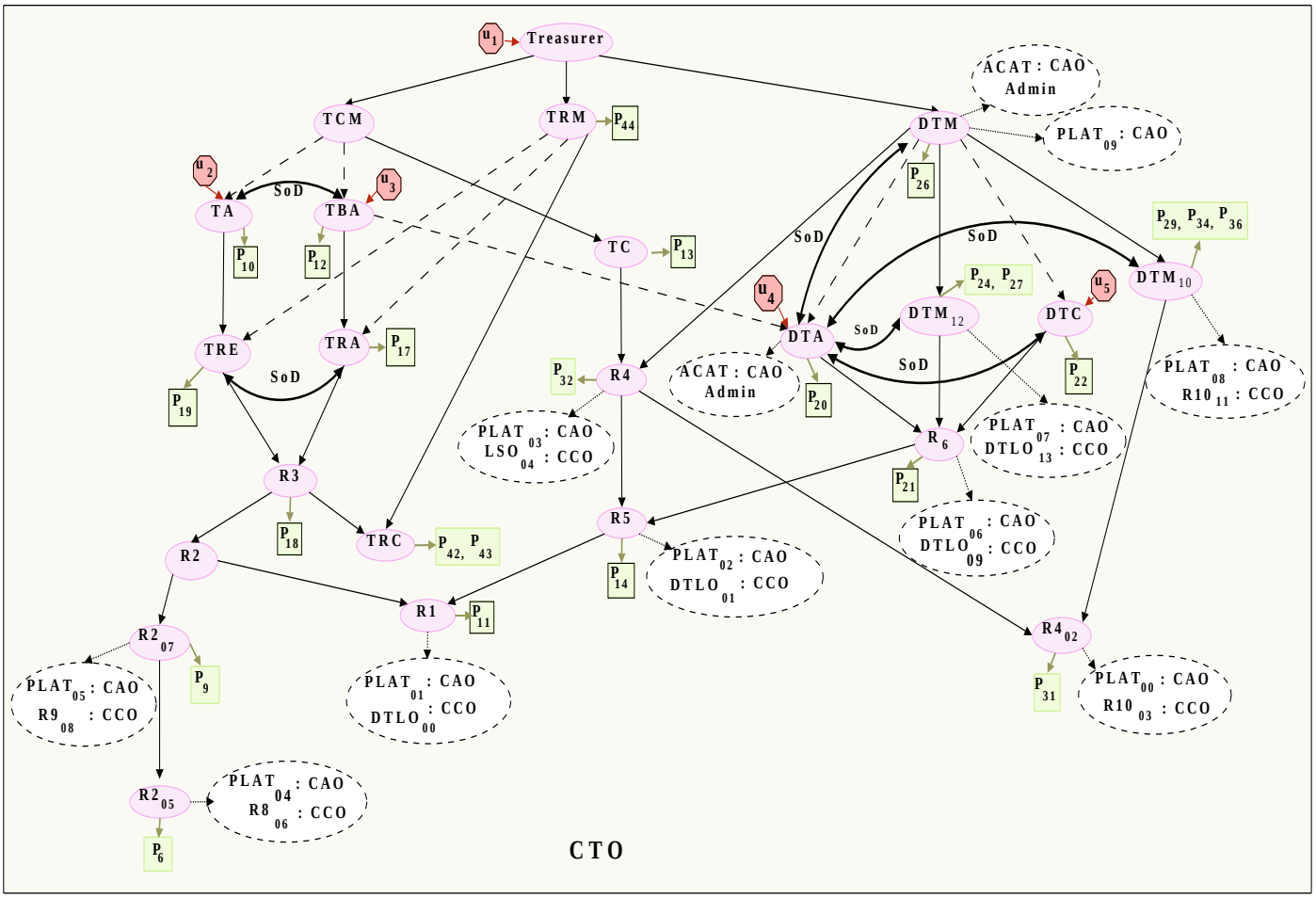
5.6.2.2.2 Exemple2

Prenant maintenant l'exemple des trois domaines *CTO*, *CCO* et *CAO*, la politique d'interopération sûre est donnée dans la figure 5.12. Après l'application de l'algorithme de la représentation minimale, les mappings montrés dans le tableau 5.2 peuvent être supprimés sans influence sur l'ensemble des accès permis de l'ensemble F.

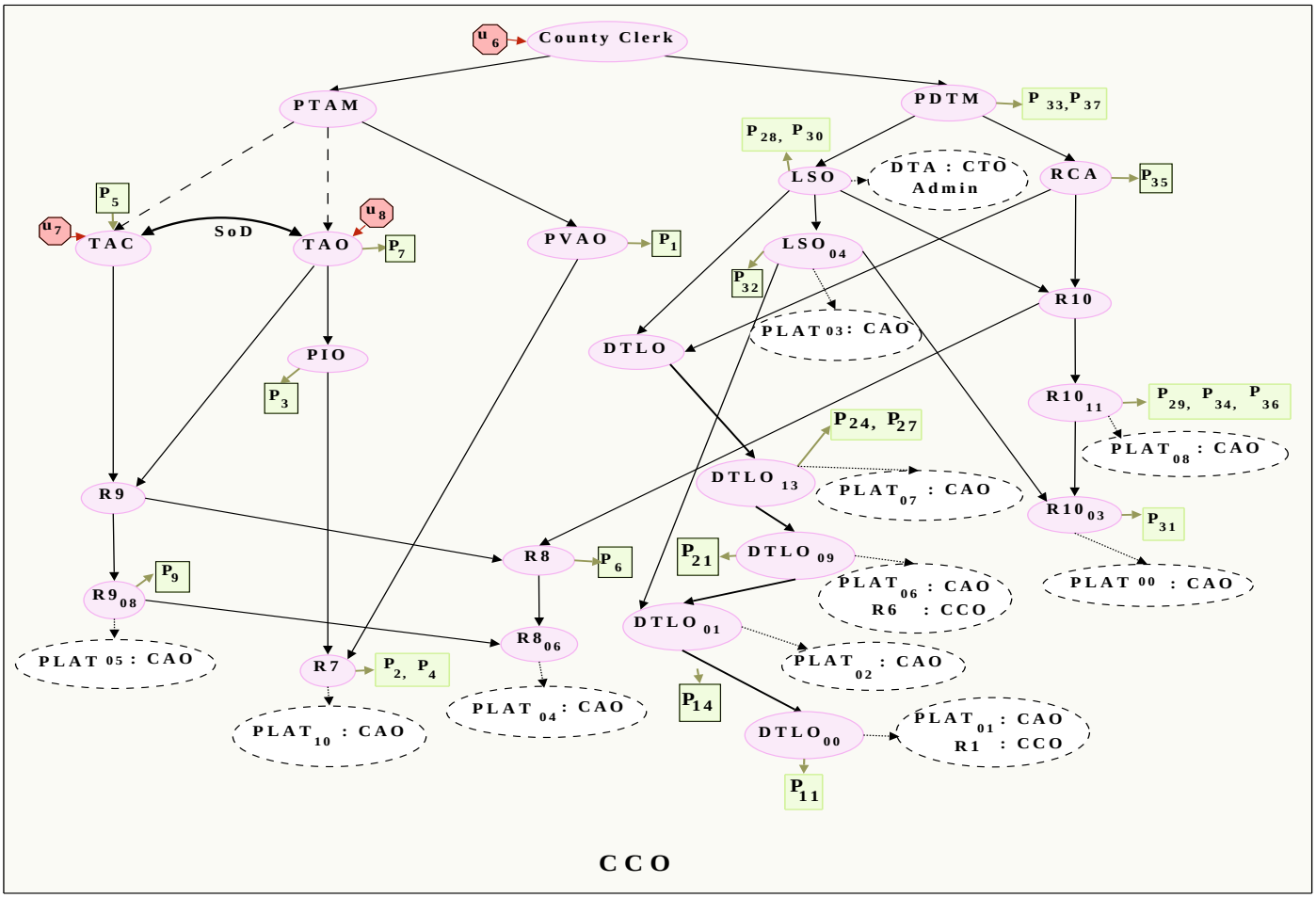
Domaine CTO→Domaine CCO	**Domaine CTO→Domaine CAO**
($R4_{02}$, $R10_{03}$), ($R2_{05}$, $R8_{06}$).	(DTM_{10},$PLAT_{08}$), (DTM_{12}, $PLAT_{07}$), (DTM, $PLAT_{09}$), ($R2_{05}$, $PLAT_{04}$), ($R4_{02}$, $PLAT_{00}$).
Domaine CCO→Domaine CTO	**Domaine CCO→Domaine CAO**
($DTLO_{00}$, R1).	($R10_{03}$, $PLAT_{00}$), ($DTLO_{01}$, $PLAT_{02}$), ($DTLO_{00}$, $PLAT_{01}$), ($DTLO_{09}$, $PLAT_{06}$).
Domaine CAO→Domaine CTO	**Domaine CAO→Domaine CCO**
($PLAT_{01}$, R1), ($PLAT_{02}$, R5).	($PLAT_{04}$, $R8_{06}$), ($PLAT_{00}$, $R10_{03}$), ($PLAT_{06}$, $DTLO_{09}$), ($PLAT_{02}$, $DTLO_{01}$), ($PLAT_{01}$,$DTLO_{00}$).

TAB. 5.2 – Les mappings redondants des rôles inter-domaines dans la politique globale RBAC des domaines collaborant : *CTO*, *CCO* et *CAO*.

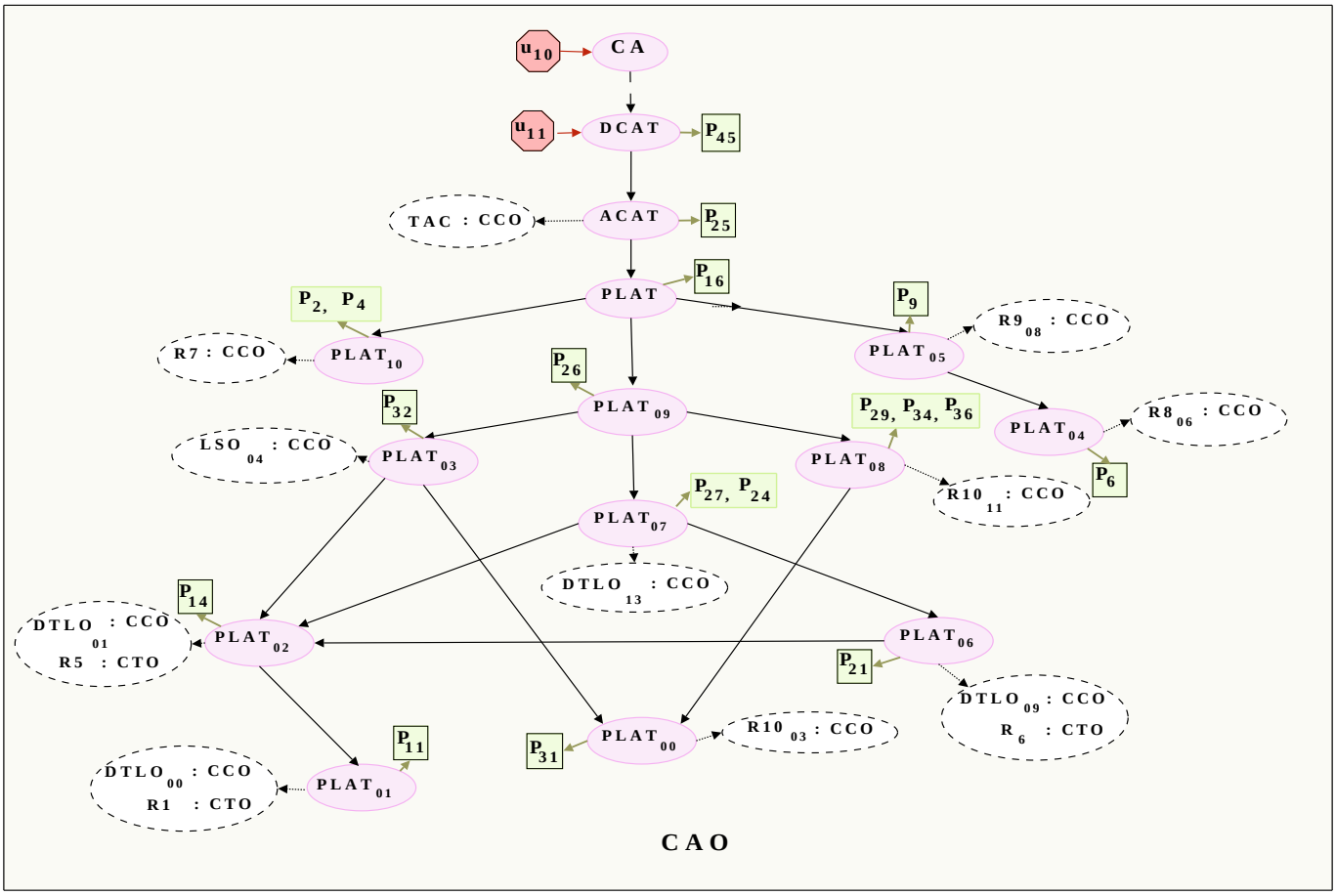
On observe dans le tableau 5.2, qu'il y a un tas de mappings des rôles inter-domaines redondants dans la politique sûre d'interopération des domaines *CTO*, *CCO* et *CAO*, qui ne donnent davantage aucun partage des données ou ressources inter-domaines, d'où il est préférable et nécessaire de les éliminer de la politique globale de l'environnement multi-domaines.


u1
Treasurer
TCM
TRM
P44
ACAT : CAO
Admin
DTM
PLAT09 : CAO
u2
SoD
u3
TA
TBA
P26
P29, P34, P36
P10
P12
TC
P13
SoD
SoD
u4
P24, P27
u5
DTM12
DTC
DTM10
TRE
TRA
P17
DTA
SoD
SoD
P32
R4
ACAT : CAO
Admin
P20
PLAT08 : CAO
R1011 : CCO
P19
SoD
PLAT03 : CAO
LSO04 : CCO
R6
PLAT07 : CAO
DTLO13 : CCO
P22
R3
P21
P18
TRC
P42, P43
R5
PLAT06 : CAO
DTLO09 : CCO
R2
PLAT02 : CAO
DTLO01 : CCO
P14
R1
P11
R202
R207
P9
P31
PLAT00 : CAO
R1003 : CCO
PLAT05 : CAO
R908 : CCO
PLAT01 : CAO
DTLO00 : CCO
R205
PLAT04 : CAO
R806 : CCO
P6
CTO

(a)


u6
County Clerk
PTAM
PDTM
P33, P37
P28, P30
LSO
DTA : CTO
Admin
RCA
P35
P5
u7
TAC
SoD
TAO
u8
P7
PVAO
P1
LSO04
P32
R10
PLAT03: CAO
DTLO
PIO
P3
R1011
P29, P34, P36
PLAT08 : CAO
P24, P27
DTLO13
PLAT07 : CAO
R9
R1003
P31
R8
P6
P21
DTLO09
PLAT00 : CAO
R9 08
P9
PLAT06 : CAO
R6 : CCO
DTLO01
R806
PLAT05 : CAO
R7
P2, P4
PLAT02 : CAO
P14
PLAT04 : CAO
DTLO00
PLAT01 : CAO
R1 : CCO
PLAT10 : CAO
P11
CCO

(b)



(c)

FIG. 5.12 – La politique d'interopérabilité sûre des trois domaines *CTO*, *CCO* et *CAO*.

5.7 Conclusion

Maximiser l'interopérabilité sûre entre les politiques de sécurité est en général est un problème NP-complet [10], et la complexité de composition et d'optimisation des politiques de contrôle d'accès RBAC augmente avec l'incrémentation des domaines collaborants. La méthode d'optimisation proposé par BASIT peut ne pas détecter et résoudre tous les conflits qui peuvent apparaître dans une politique RBAC multi domaines. Nous avons amélioré cette méthode et nous avons obtenu de bons résultats, aussi la représentation de la politique RBAC multi-domaines optimisée ne peut pas être minimale. Pendant la phase de composition des politiques RBAC, des mappings des rôles inter-domaines redondants peuvent être générés. L'application d'un algorithme qui détecte les mappings des rôles inter-domaines redondants après l'étape d'optimisation est nécessaire pour réduire un peu la complexité et simplifier la gestion des accès inter-domaines .

Conclusion générale et Perspectives

Dans ce travail, nous avons traités le problème de gestion de contrôle d'accès dans les environnement multi-domaines ; il s'agit de permettre à des utilisateurs de domaines de sécurité indépendants d'accéder à des ressources d'un domaine à un autre avec la garantie de disposer de suffisamment de droits pour travailler et seulement des droits minimums requis.

Particulièrement notre projet consiste à optimiser les politiques multi-domaines basées sur le modèle de contrôle d'accès RBAC. Nous avons vu que dans la littérature il n' y a pas beaucoup de travaux sur la composition des politiques RBAC, et que nous sommes basés sur la méthode de composition de BASIT, qui est constituée de deux étapes la première consiste à composer la politique RBAC multi-domaines à partir des politiques RBAC des domaines collaborants, par la définition des mappings entre les rôles des domaines collaborant, la deuxième étape consiste à détecter et résoudre les conflits peuvent apparaître dans la politique multi-domaines générée, d'une manière optimale.

La violation de sécurité des politiques des domaines collaborant n'est pas permise, et une autre exigence d'intégration de politique est de préserver l'autonomie de tous les domaines collaborants. Cependant, il y a un compromis entre l'interopérabilité recherchée et la préservation de l'autonomie. Toutefois, les domaines peuvent tolérer une certaine perte d'autonomie pour établir plus d'interopérabilité. BASIT a formulé le problème d'interopération sûre comme un problème d'optimisation avec l'objectif de maximiser l'interopérabilité sans causer aucune violation de sécurité de domaines collaborants et garder les pertes d'autonomie des domaines collaborants dans des limites acceptables. Cependant la méthode d'optimisation de BASIT peut ne pas détecter et résoudre tous les conflits de politique RBAC multi-domaines. Nous avons proposé une amélioration de cette méthode à fin d'obtenir une politique RBAC multi-domaines sûre, aussi nous avons ajouté une étape après l'étape d'optimisation pour obtenir une politique RBAC d'interopérabilité sûre avec une représentation minimale.

Par ailleurs, il est envisagé à plus long terme d'utiliser des méthodes formelles pour vérifier la consistance et la conformité aussi bien de la politique de sécurité intra-domaines que de la politique multi-domaines générée.

La cohérence de politique multi-domaines générés par la structure proposée d'intégration des politiques, dépend de la cohérence des politiques de contrôle d'accès des domaines collaborants. Si la politique de contrôle d'accès de n'importe quel domaine de collaboration est en conflit alors la politique multi-domaines résultante sera incohérente. Donc, les politiques de contrôle d'accès des domaines doivent être vérifiées avant que l'interopérabilité soit établie. La vérification des politiques de sécurité de différents domaines doit précéder l'étape d'intégration de politique.

La vérification de la politique du contrôle d'accès d'un domaine entraîne des divers défis, y compris :

i) La spécification de la politique de contrôle d'accès en utilisant un modèle formel.
ii) L'identification des exigences de sécurité.
iii) La détermination si une politique donnée rapporte des accès non autorisés.

Le modèle de spécification de politique devrait être générique et assez flexible pour exprimer une vaste gamme de politiques de sécurité et de contrôle d'accès.

Ainsi, les exigences de sécurité et de contrôle d'accès dans un environnement multi-domaines, peuvent changer à cause des raisons suivantes : l'évolution des politiques de contrôle d'accès des domaines collaborants avec le temps, l'ajout de nouveaux domaines au système de collaboration, la suppression des domaines de collaboration, et aussi la politique de sécurité multi-domaines elle-même peut changer ; les administrateurs responsables de politique globale d'interopération peuvent définir de nouvelles règles ou des contraintes pour des accès inter-domaines. Par conséquent, la politique d'interopération doit être redéfinie pour intégrer les nouvelles exigences de sécurité et de contrôle d'accès.

La définition d'une nouvelle politique d'interopérabilité en réintégrant les politiques de contrôle d'accès est un processus long et peut ne pas être viable dans les environnements où les politiques des domaines changent fréquemment. Alors, un mécanisme d'ajustement de politique est nécessaire pour capturer toutes les modifications des politiques, et reconfigurer la politique d'interopérabilité en temps utile.

Bibliographie

[1] V. ATLURI and W-K. HUANG. An authorization model for workflows. In *European symposium on research in computer security*, Rome, Italy, 1996. Springer Verlag.

[2] V. ATLURI, W-K. HUANG, and E. BERTINO. A semantic based execution model for multilevel secure workflows. In *Journal of Computer Security*, pages 8(1) :3–41, 2000.

[3] S. BASIT, J. JAMES, E. BERTINO, and A. GHAFOOR. Secure interoperation in a multidomain environment employing rbac policies. *IEEE transactions on knowledge and data engineering*, 17(11), 2005.

[4] E. BERTINO, E. FERRARI, and V. ATLURI. Specification and enforcement of authorization constraints in workflow management systems. In *ACM Transactions on Information and System Security*, Février 1999.

[5] S. DAWSON, S. QIAN, and P. SAMARATI. Providing security and interoperation of heterogeneous systems. In *Distributed and Parallel Databases*, volume 8, pages 119–145, Aug 2000.

[6] E. DISSON and D. BOULANGER. Utilisation d'xacml pour le contrôle d'accès dans les systèmes d'information coopératifs. In *INFORSID*, pages 17–31, 2006.

[7] T. FERNANDES. Les politiques de sécurité. juin 2001.

[8] D. FERRAILO, J. CUGINI, and D. RICHARD KUHN. Role based access control (rbac) : Features and motivations. 1995.

[9] S. GAVRILA and J. BARKLEY. Formal specification for role based access control user/role and role/role relationship management. In S.I. Gavrila and J.F. Barkley, editors, *In Proceedings of Third ACM Workshop on Role-Based Access Control*, pages 81–90, Octobre 1998.

[10] L. GONG and X. Q IAN. Computational issues in secure interoperation. *IEEE transactions on software engineering*, 22(1), 1996.

[11] A. HADDAD and T. MOUTET. Modélisation et vérification de contrôle d'accès selon une démarche critères communs. *International conference on risks and security of internet and systems (CRISIS 2007), Marrakech, Maroc*, pages 117–124, Juillet 2007.

[12] S. HOCINE. *Identification de Modèles de Procedes par Programmation Mixte Déterministe.* PhD thesis, L'institut national polythecnique de Toulouse, Décembre 2006.

[13] J. JOSHI, A. GHAFOOR, W. AREF, and E.SPAFFORD. Digital government security infrastructure design challenges. *IEEE Computer Society Press*, 34(2) :66 – 72, Février 2001.

[14] J.D. JOSHI, E. BERTINO, U. LATIF, and A. GHAFOOR. A generalized temporal role-based access control model. *IEEE transactions on knowledge and data engineering*, 17(1) :4–23, Janvier 2005.

[15] A. ABOU EL KALAM. *Modèles et Politiques de Sécurité pour les Domaines de la Santé et des Affaires Sociales.* PhD thesis, 2003.

[16] A. ABOU EL KALAM, R. EL BAIDA, P. BALBIANI, S. BENFERHAT, F. CUPPENS, Y. DESWARTE, A. MIEGE, C. SAUREL, and G. TROUESSIN. Orbac : un modèle de contrôle d'accès basé sur les organisations. *Cahiers francophones de la recherche en sécurité de l'information*, 2 :30–40, 2003.

[17] M. KOCH, L. MANCINI, and F. PARISI-PRESICCE. A graph-based formalism for rbac. *ACM Transactions on Information and System Security*, 5(3) :332–365, 2002.

[18] R. LABORDE and T. DESPRATS. Gestion de conditions stables dans xacml : Intérêt d'une approche par notification. In Sami Tabbane and Choukair Zied, editors, *Gestion de REseaux et de Services (GRES), Hammamet,Tunisie, 06/11/07-09/11/07*, http ://www.editions-hermes.fr/, Décembre 2008. Hermès Science Publications.

[19] W.S. LI and C. CLIFTON. Semantic integration in heterogeneous databases using neural networks. In *Proc. of the 20th Int. Conf. on Very Large Data Bases (VLDB-94)*, pages 1–12, 1994.

[20] E. LUPU and M. SLOMAN. Conflicts in policy-based distributed systems. *IEEE Transactions on Software Engineering*, 25 :852–869, 1999.

[21] T. MOSES. This specification defines version 2.0 of the extensible access-control markup language. (1), Février 2005.

[22] M.TAWARMALANI. Convexification and global optimization of nonlinear programs. *Workshop on Integer Programming and Continuous Optimization, Chemnitz*, Novembre 2004.

[23] M.TAWARMALANI. Mixed integer nonlinear programs theory, algorithms and applications. *Department of Mechanical and Industrial Engineering University Of IllinoisUrbana-Champaign*, 2004.

[24] R. POTTINGER and P. BERNSTEIN. Merging models based on given correspondences. In *In VLDB*, pages 826–873, 2003.

[25] R.THOMAS and R.SANDHU. Task-based authorization controls (tbac) : A family of models for active and enterprise-oriented authorization management. In *11 th IFIP Working Conference on Database Security*, Lake Tahoe, California, USA, 1997.

[26] P. SAMARATI and S. DE CAPITANI DI VIMERCATI. Access control : Policies, models, and mechanisms. *Foundations of Security Analysis and Design*, pages 137–196, 2001.

[27] R. SANDHU, E. COYNEK, H. FEINSTEINK, and C. YOUMANK. Role-based access control models. *IEEE Computer*, 29(2) :38–47, Février 1996.

[28] R. SANDHU, D. FERRAIOLO, and R. KUHN. The nist model for role based access control : Towards a unified standard. In *In Proceedings of the fifth ACM workshop on Role-based access control*, pages 47–63, 2000.

[29] R. THION and S. COULONDRE. Les dépendances généralisées comme support à l'expression et à la vérification de politiques de sécurité. In *XXIIIème Congrès INFORSID, Atelier "Sécurité des Systèmes d'Information", SSI'05*, pages 89–100, 2005.

[30] L. WOLSEY. *Integer Programming*. New York, 1998.

[31] G. YAN, W.K. NG, and E. L IM. Product schema integration for electronic commerce a synonym comparison approach. *IEEE Trans. Knowledge and Data Eng*, 14(03) :583–598, Mai-Juin 2002.